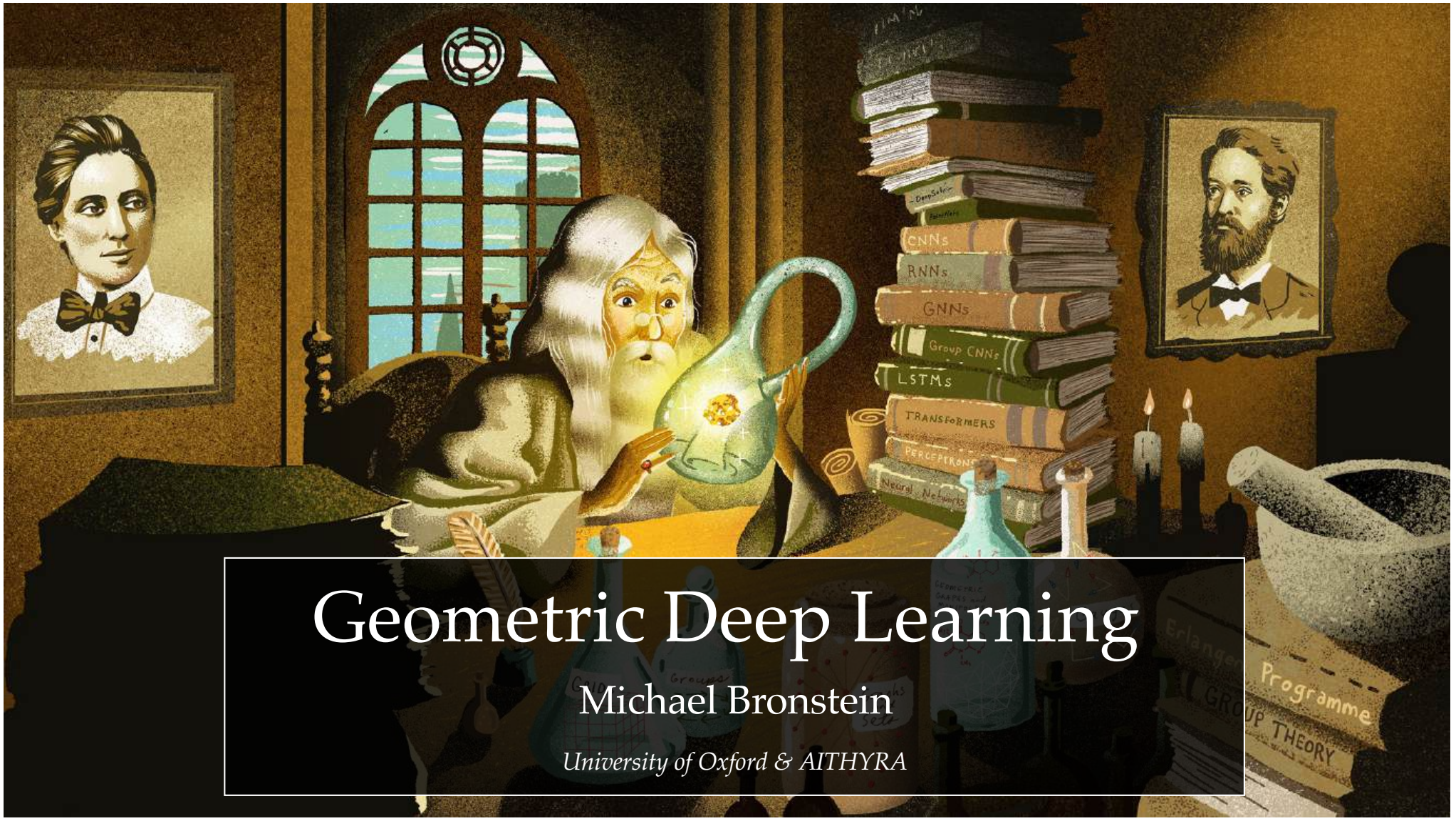


Geometric Deep Learning

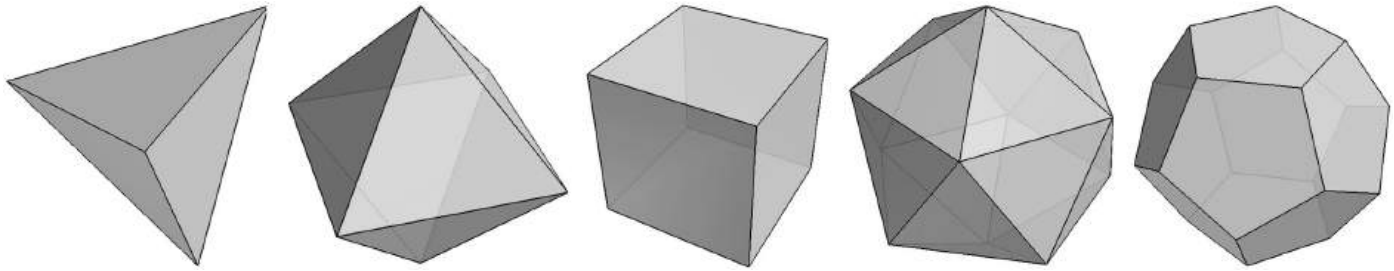
Michael Bronstein

University of Oxford & AITHYRA

"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

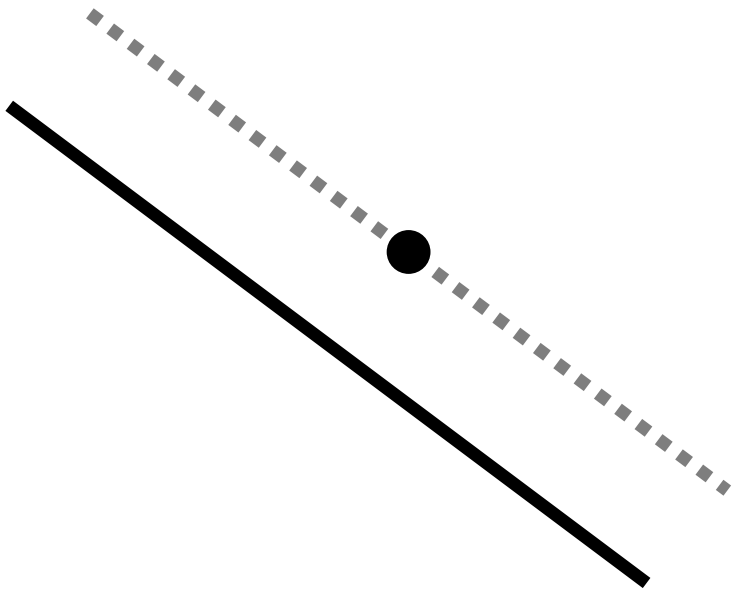


Plato

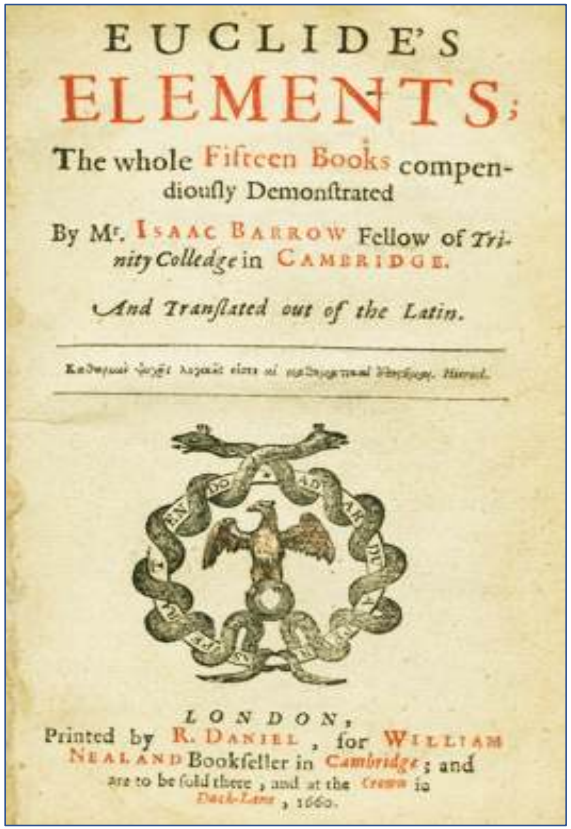
~370 BC

Portrait: Ihor Gorskyi


ΑΓΕΩΜΕΤΡΗΤΟΣ
ΜΗΔΕΙΣ ΕΙΣΙΤΩ
Oxford Math Institute



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 NEAL AND 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.

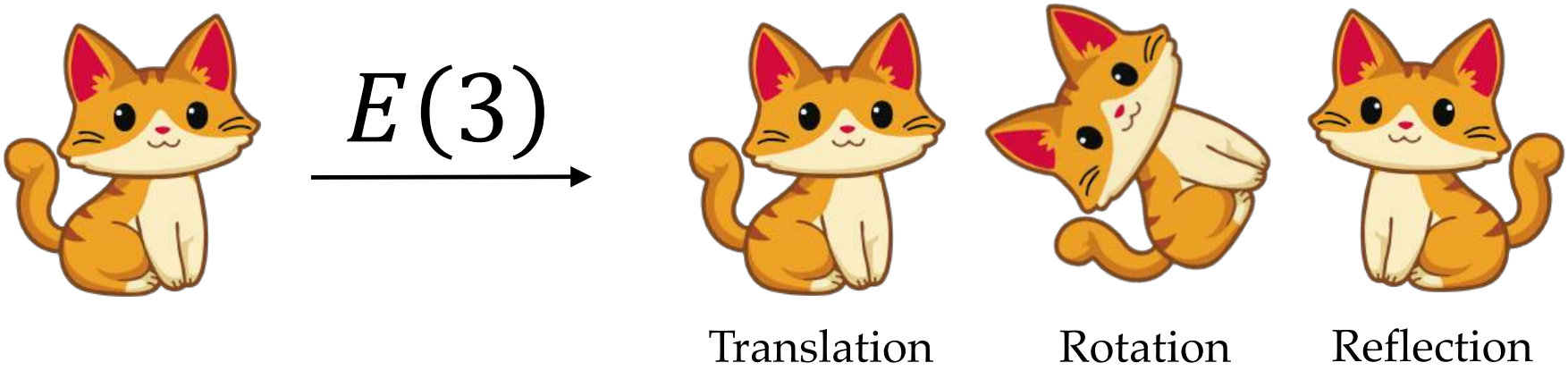
Erlangen.
Verlag von Andreas Deichert.
1872.

F. Klein

1872

Klein 1872

Euclidean geometry

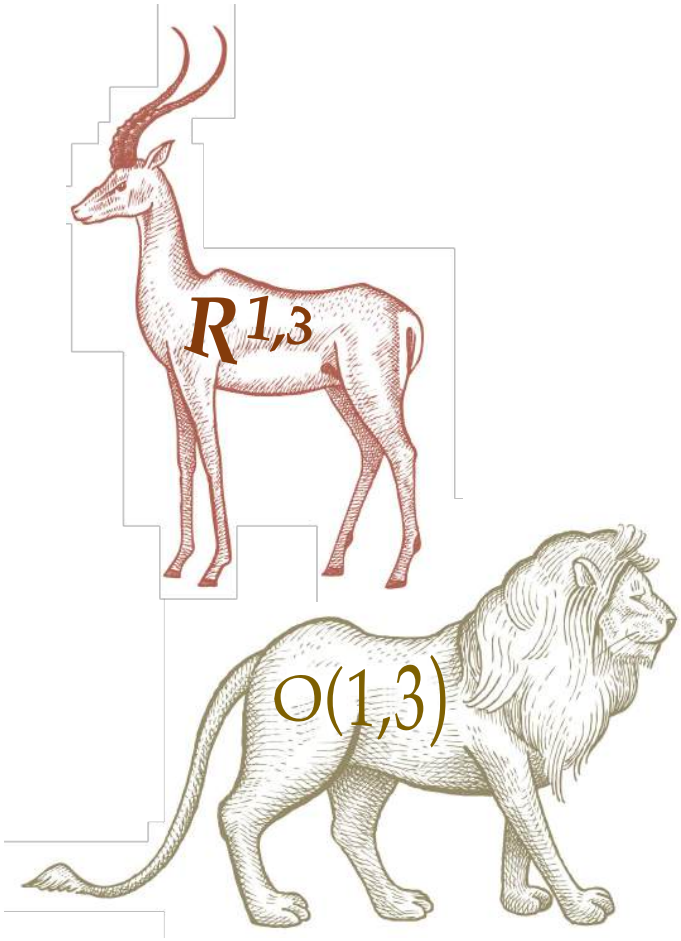


Klein 1872

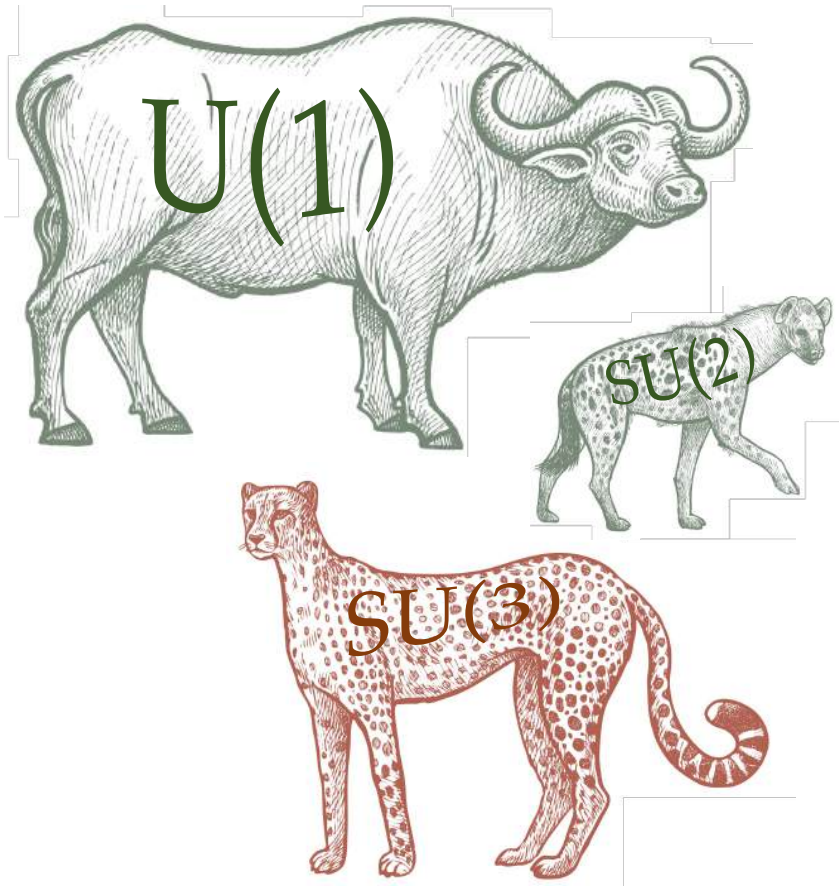
H. Poincaré **H. Minkowski** **E. Noether** **H. Weyl** **C. N. Yang** **R. L. Mills**

1904 1907 1918 1929 1954

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



External symmetry



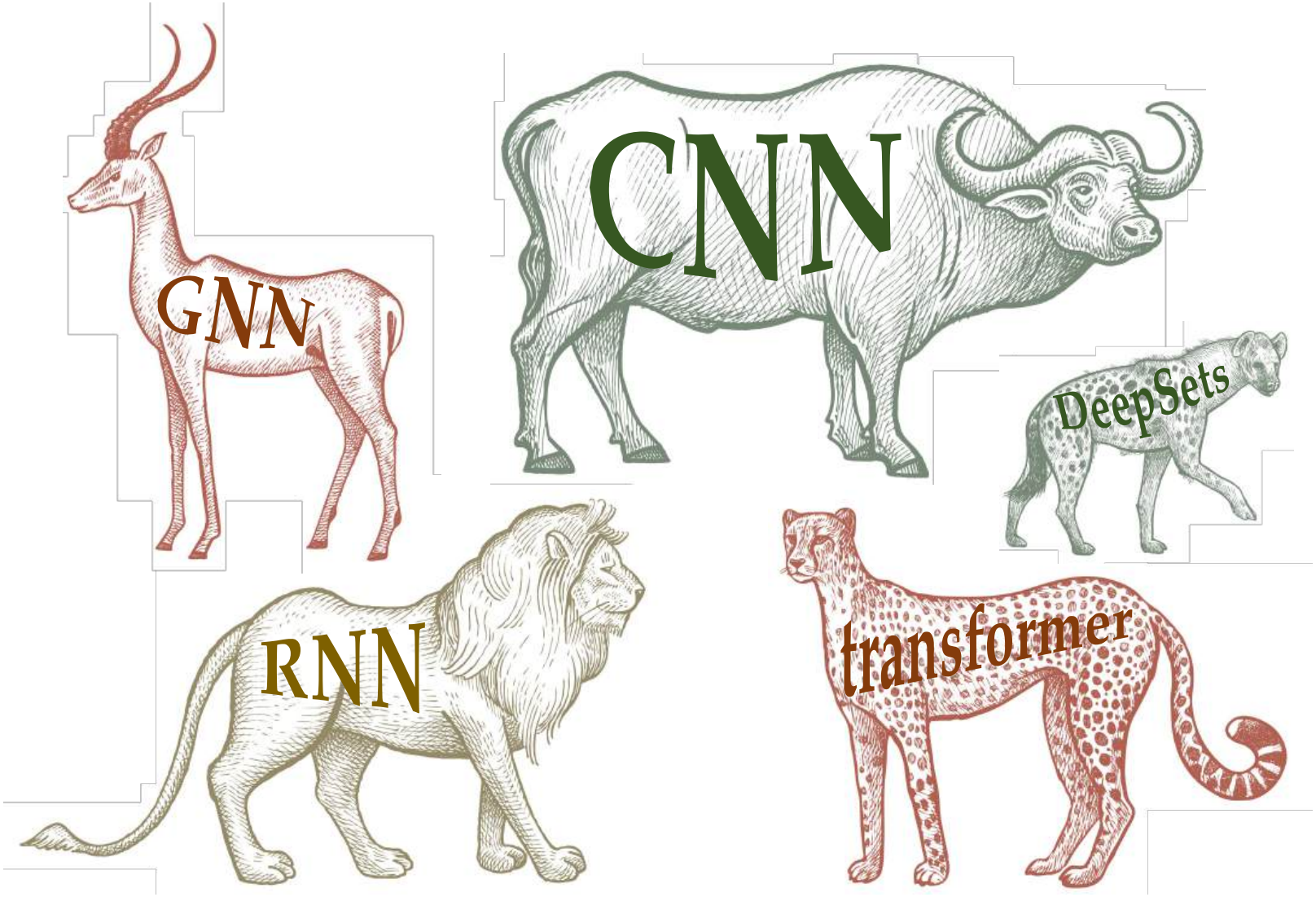
Internal symmetry

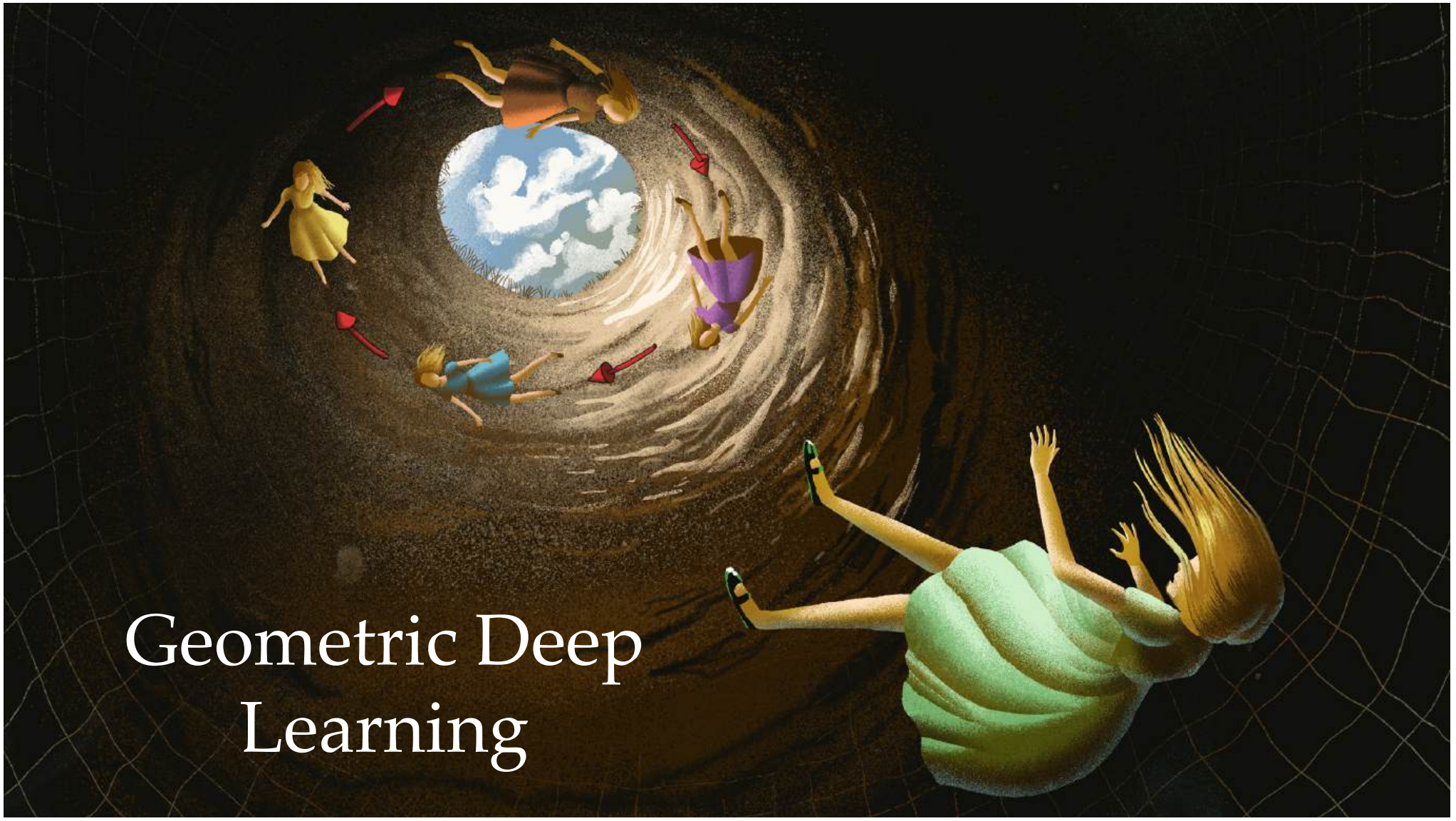
"It is only slightly overstating the case to say that Physics is the study of symmetry"

— *More is different*

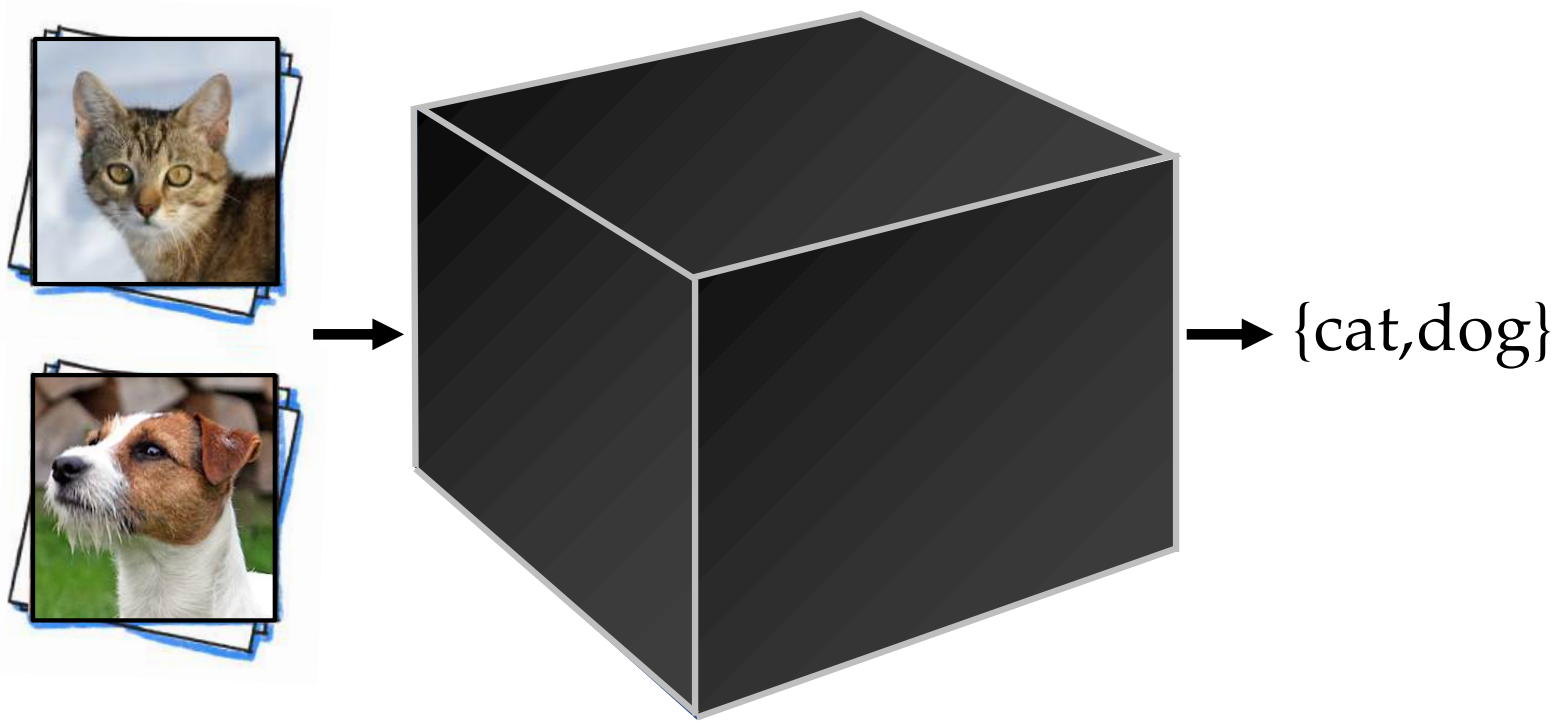
P. Anderson

Anderson 1972

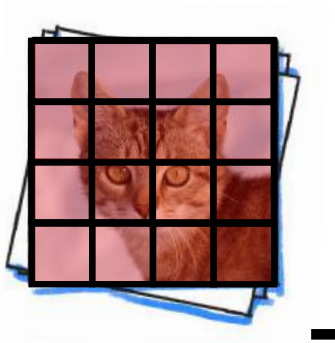

GNN
CNN
DeepSets
RNN
transformer

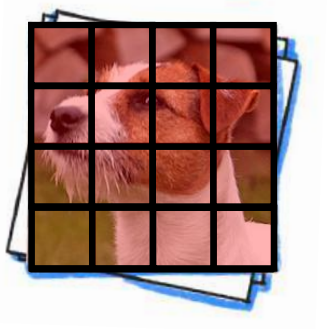

Geometric Deep
Learning

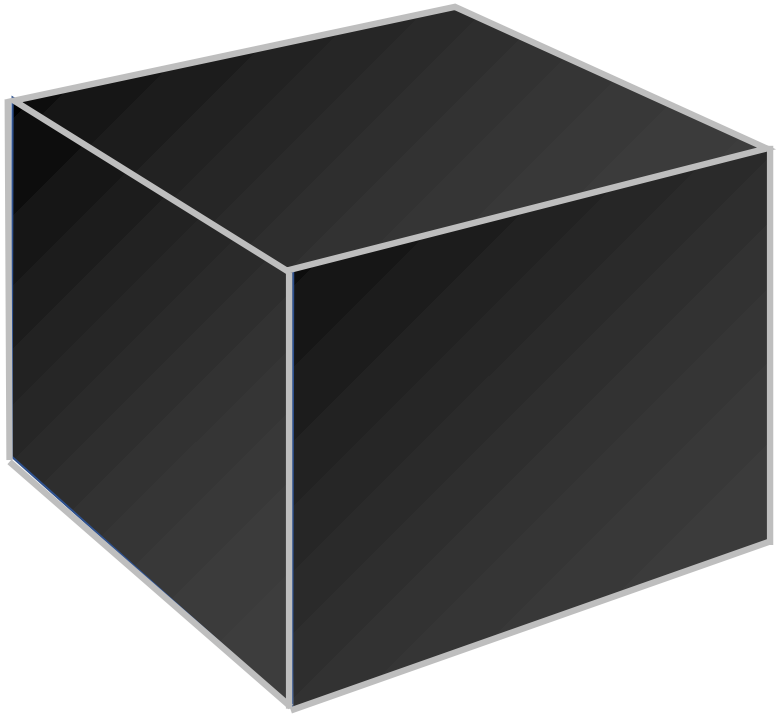
Supervised ML = Function Approximation



Supervised ML = Function Approximation







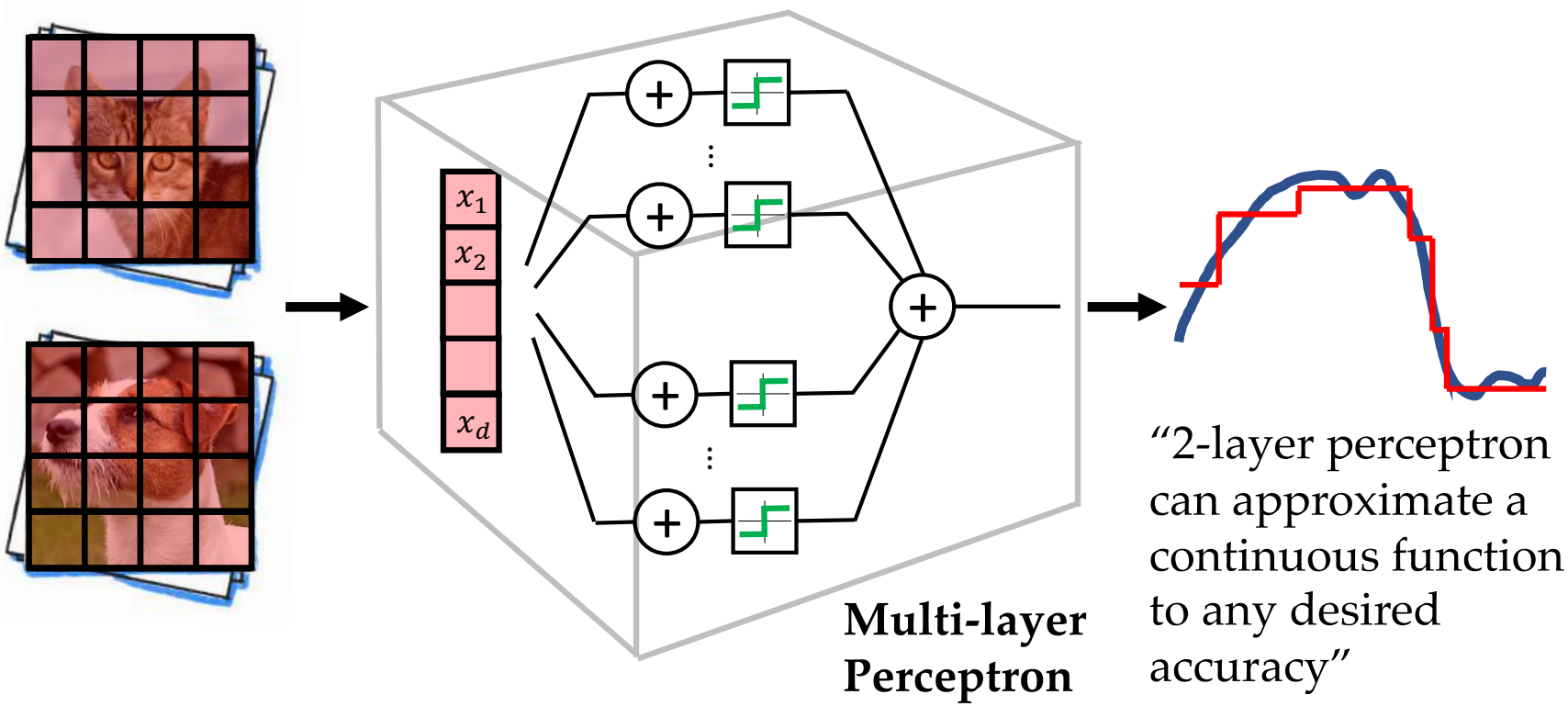
{cat,dog}

F. Rosenblatt

1957

Rosenblatt 1957; Portrait: Ihor Gorskyi

Universal Approximation

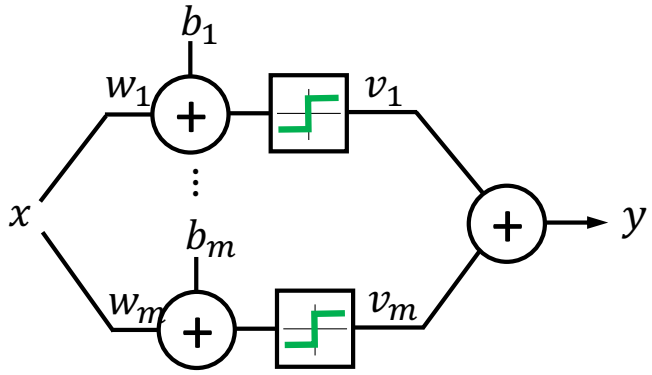


Universal Approximation: Hilbert's 13th problem 1900; Kolmogorov 1956; Arnold 1957; Cybenko 1989; Hornik 1991; Barron 1993; Leshno et al 1993; Maiorov 1999; Pinkus 1999

Shallow Perceptrons are universal approximators

Universal Approximation Theorem: σ is not polynomial iff for every continuous function $f: K \subset \mathbb{R}^d \to \mathbb{R}$ defined on a *compact set* K and $\varepsilon > 0$, there exists a two-layer Perceptron with m neurons and weights $\mathbf{W}, \mathbf{b}, \mathbf{v}$ s.t.

$$\max_{x \in K} \left| f(x) - \sum_{j \leq m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right| < \varepsilon$$



Cybenko 1989; Hornik 1991; Pinkus 1999

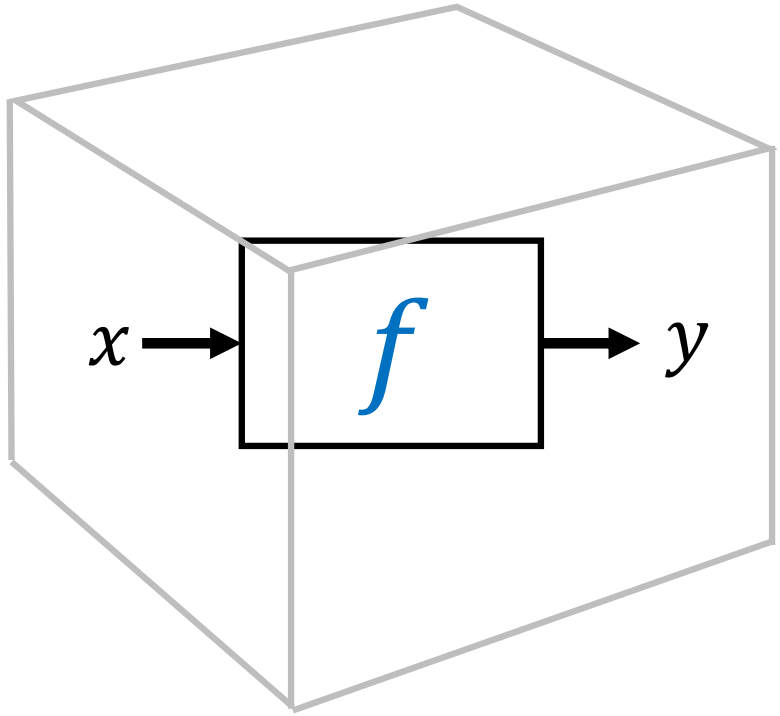
Shallow Perceptrons are universal approximators

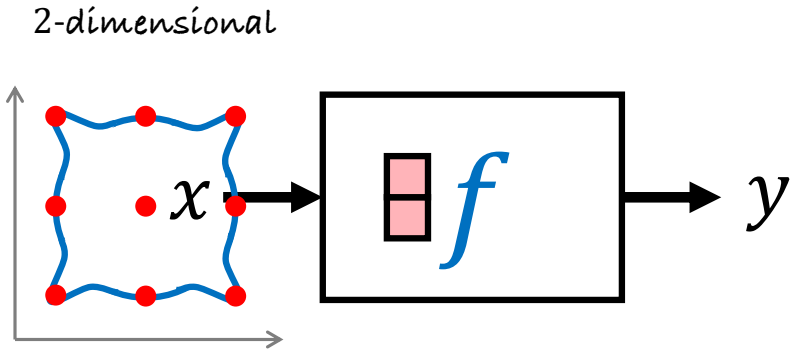
Universal Approximation Theorem: σ is not polynomial iff for every continuous function $f: K \subset \mathbb{R}^d \to \mathbb{R}$ defined on a *compact set* K and $\varepsilon > 0$, there exists a two-layer Perceptron with m neurons and weights $\mathbf{W}, \mathbf{b}, \mathbf{v}$ s.t.

$$\max_{x \in K} \left| f(x) - \sum_{j \leq m} v_j \sigma(w_j^{\mathrm{T}} x + b_j) \right| < \varepsilon$$

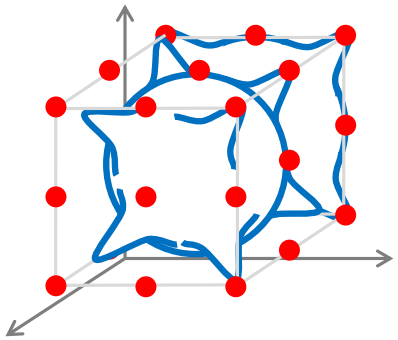
What is the relation between dimension d, number of neurons m, and the error ε?

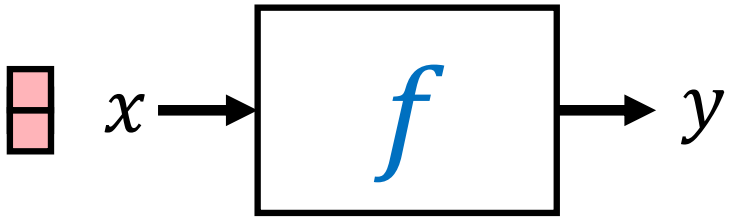
Cybenko 1989; Hornik 1991

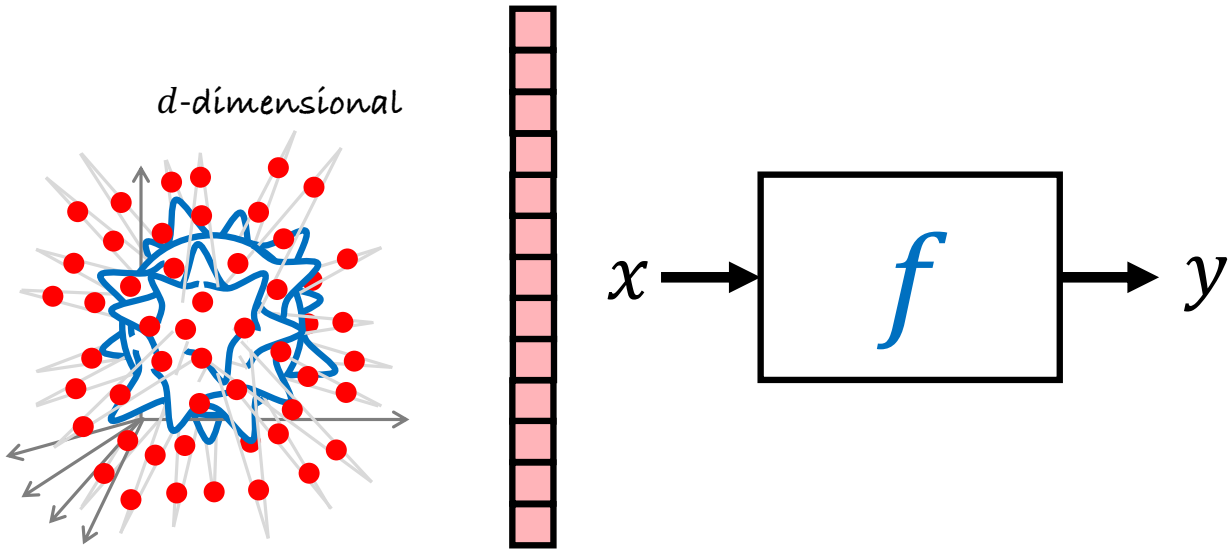

x
f
y


2-dimensional
x
f
y

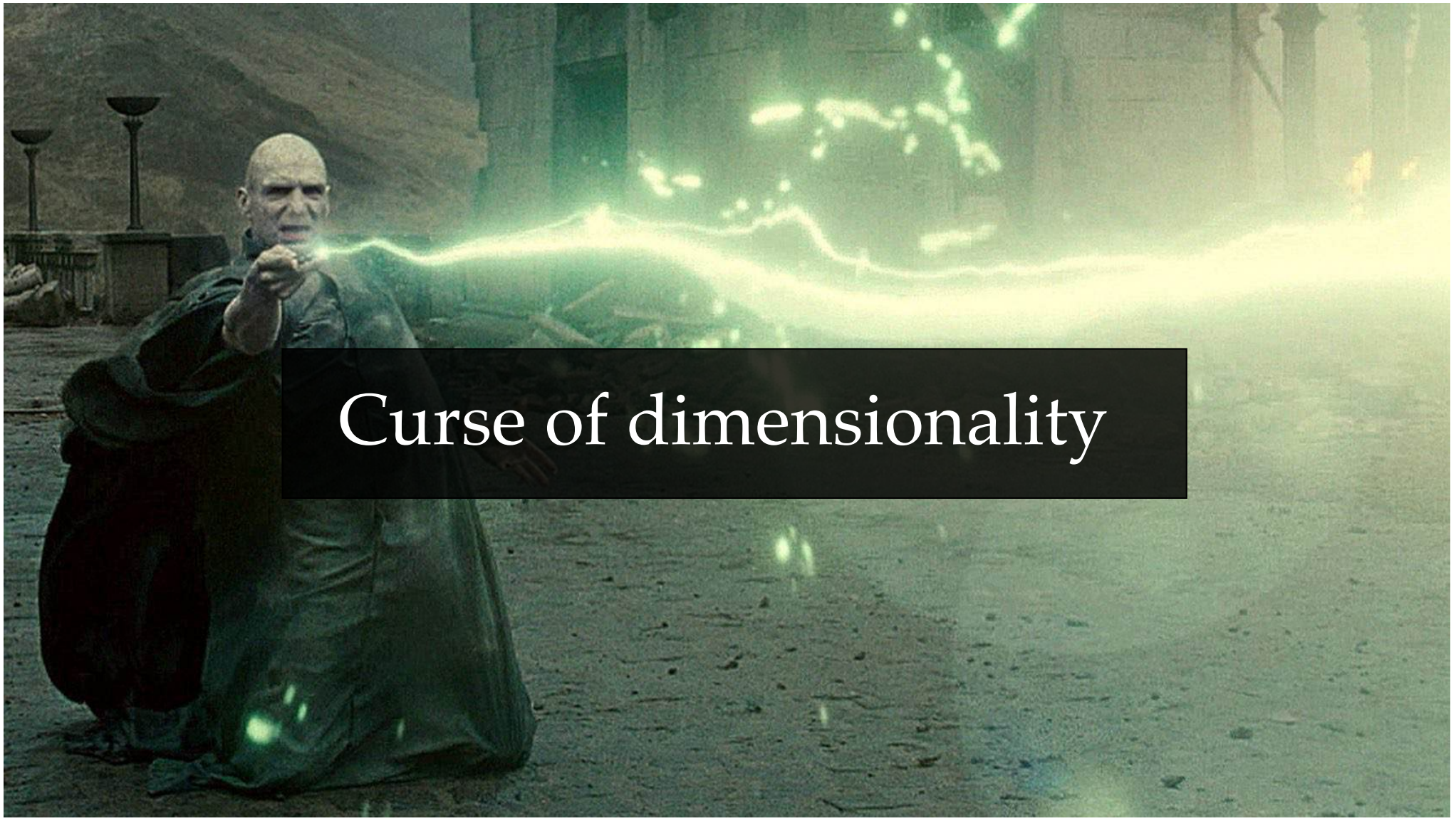
3-dimensional







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


Curse of dimensionality

Approximation rates

- Bound on the approximation error

$$\varepsilon = \inf_{g \in \mathcal{F}} \sup_{x \in K \subset \mathbb{R}^d} |f(x) - g(x)|$$

w.r.t. d (input dimension) and m (model complexity) for different classes of functions

- *Sobolev class* $f \in H^s(\mathbb{R}^d) = \left\{ f \in L_2(\mathbb{R}^d) : \int_{\mathbb{R}^d} (1 + \|\omega\|^2)^s \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty \right\}$

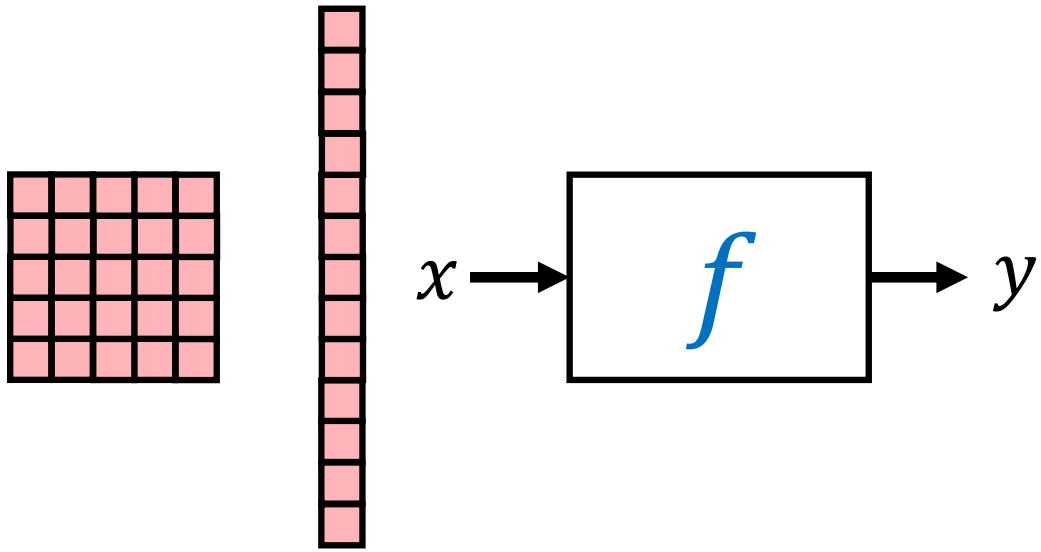
  error is exponential $\varepsilon = \mathcal{O}(m^{-s/d})$ **dimensionality-cursed!**

- *Barron class* $f \in \left\{ f \in L_2(\mathbb{R}^d) : \int_{\mathbb{R}^d} \|\omega\|^2 \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty \right\}$

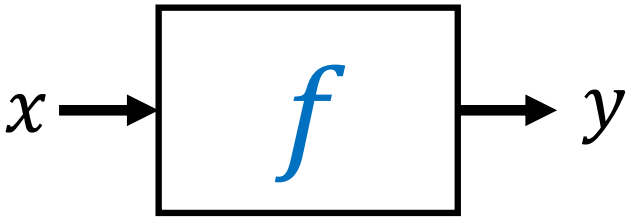
  error is $\varepsilon = \mathcal{O}(m^{-1})$ **too strong assumption in practice!**

Maiorov 1999; Barron 1993

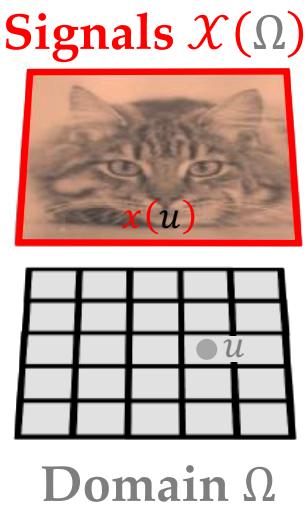
Geometric priors

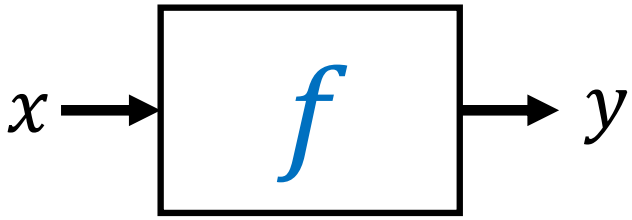


Geometric priors

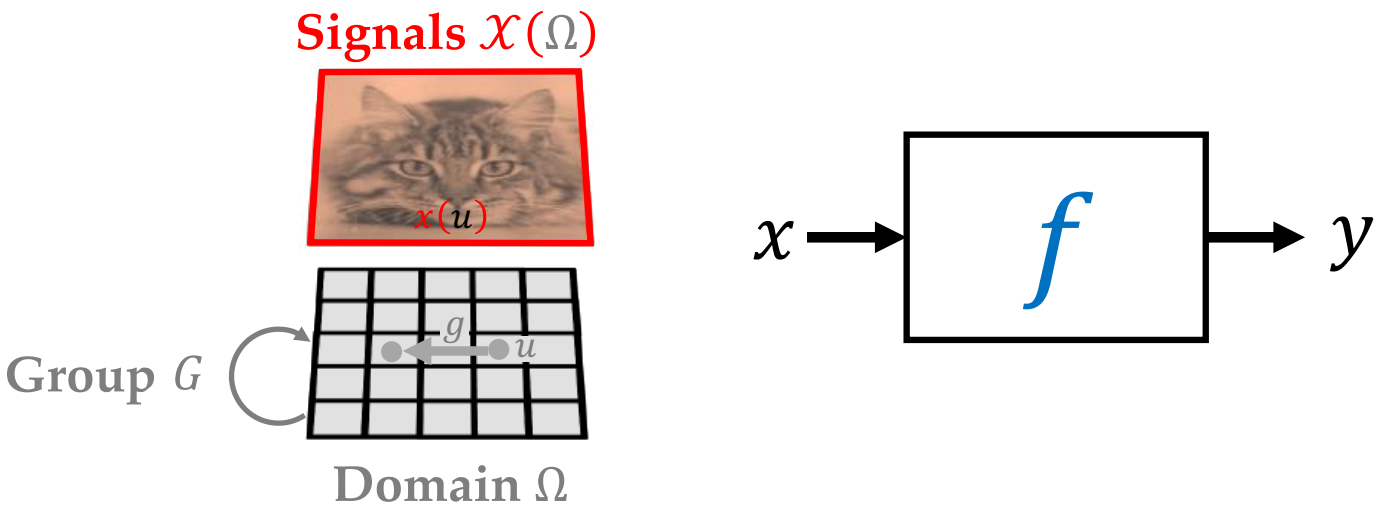


Geometric priors

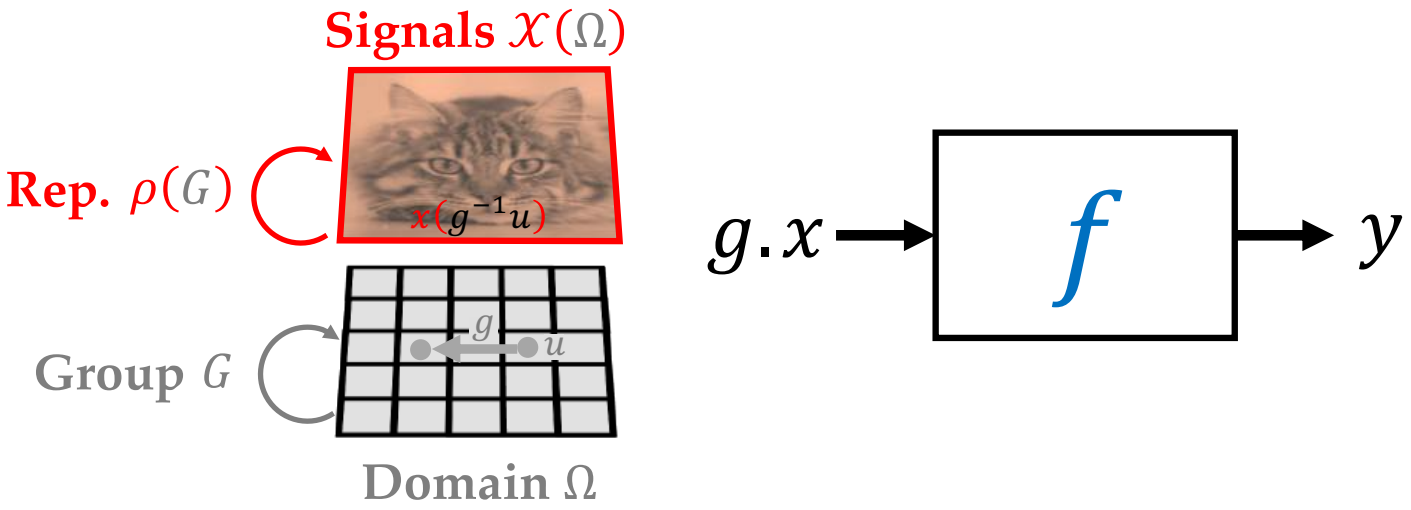




Geometric priors

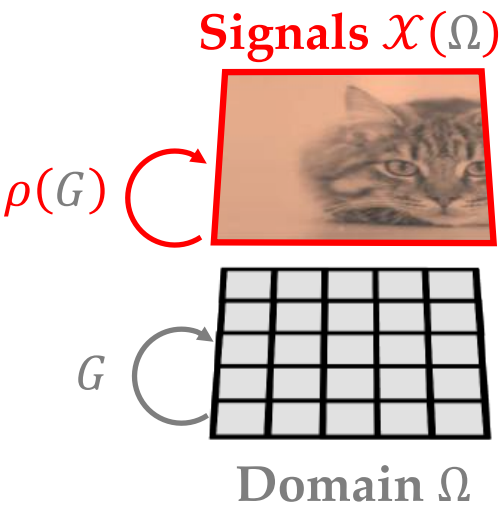


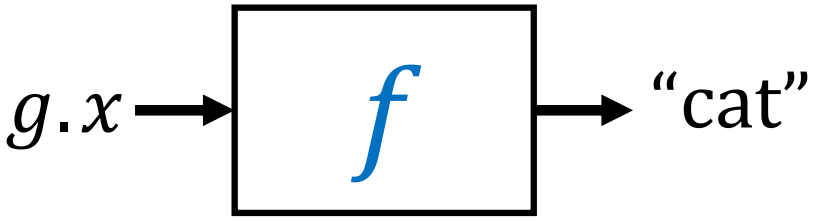
Geometric priors



"How f interacts with the group G?"

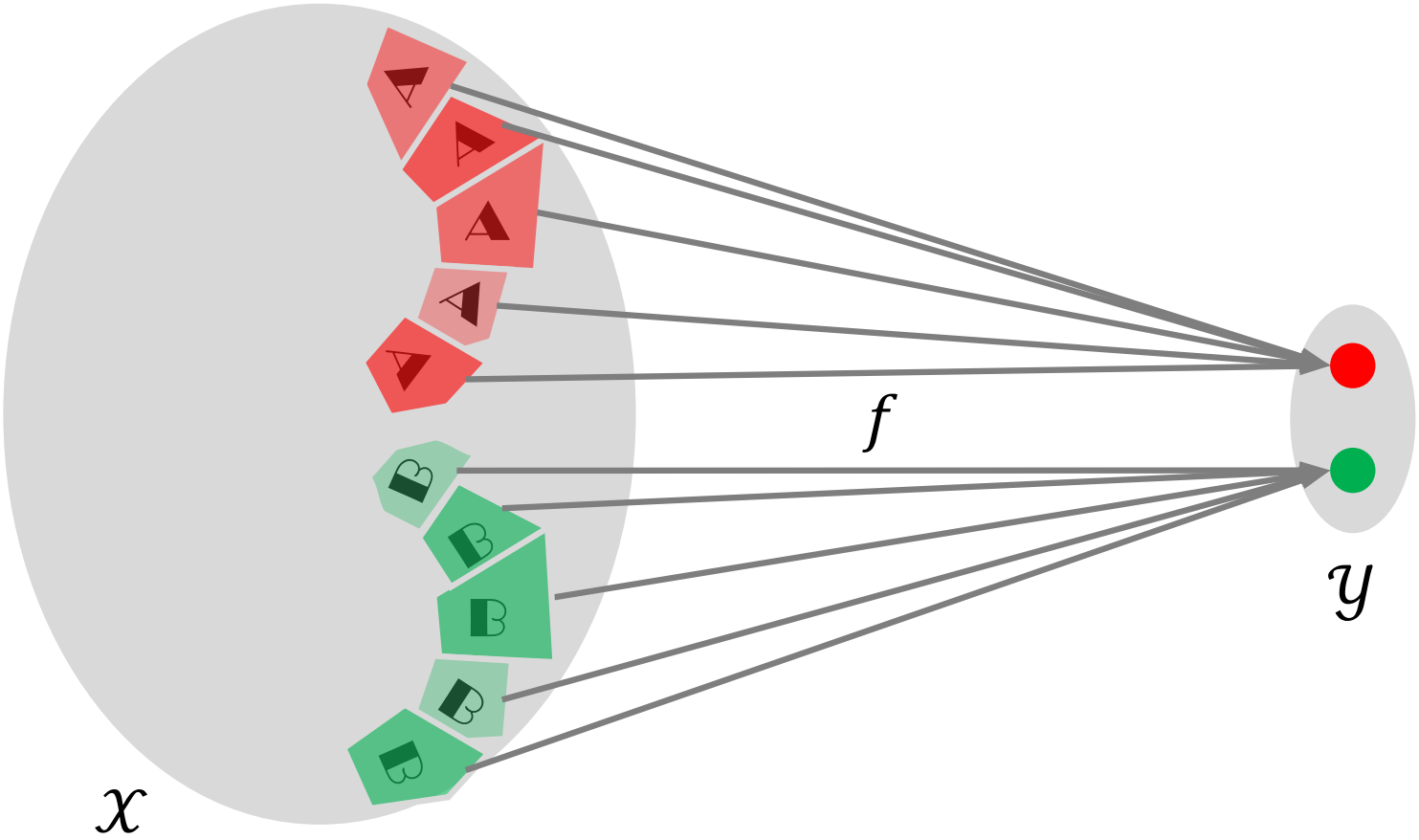
Geometric priors: Invariance



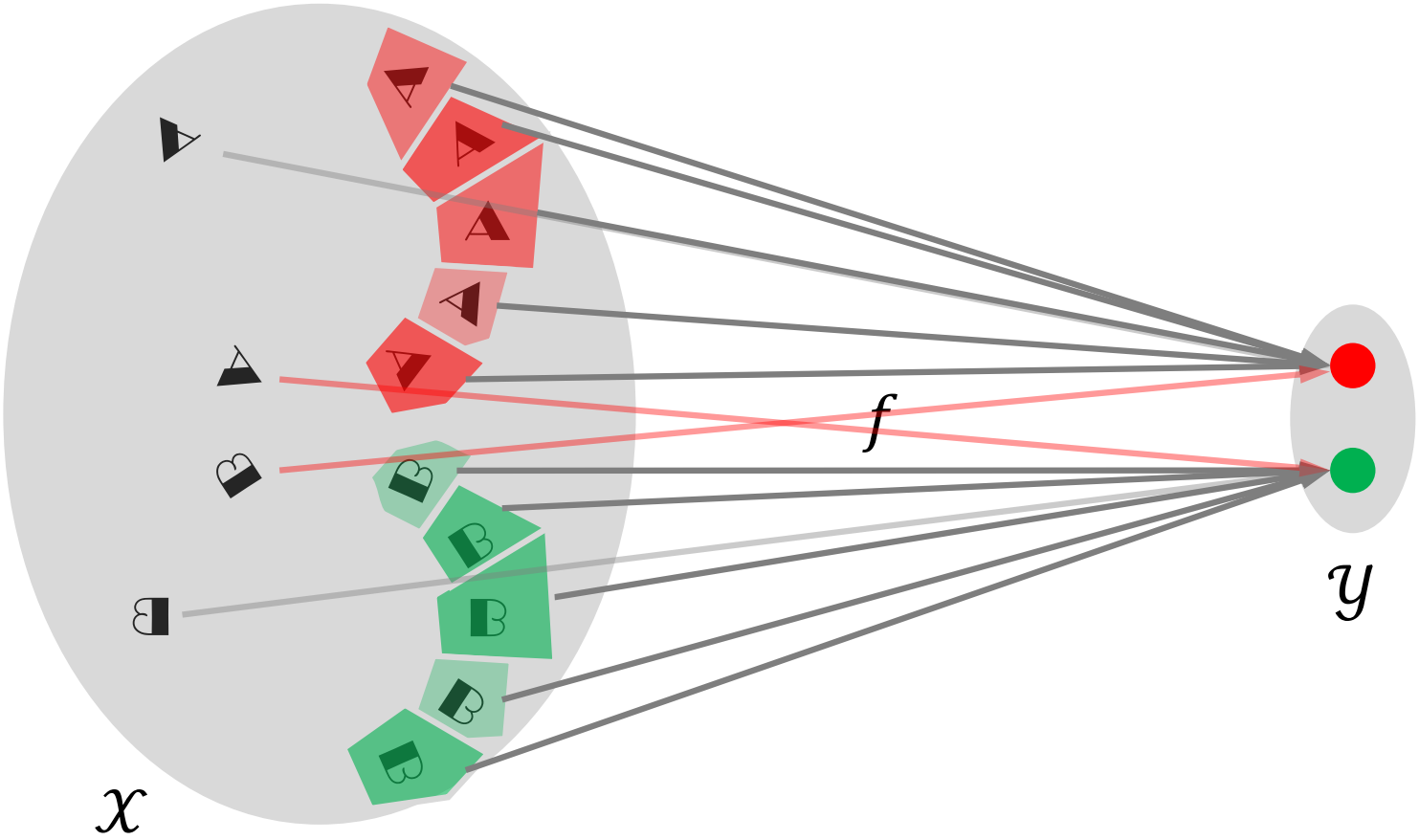


$$f(\rho(g)x) = f(x) \quad \forall g \in G$$

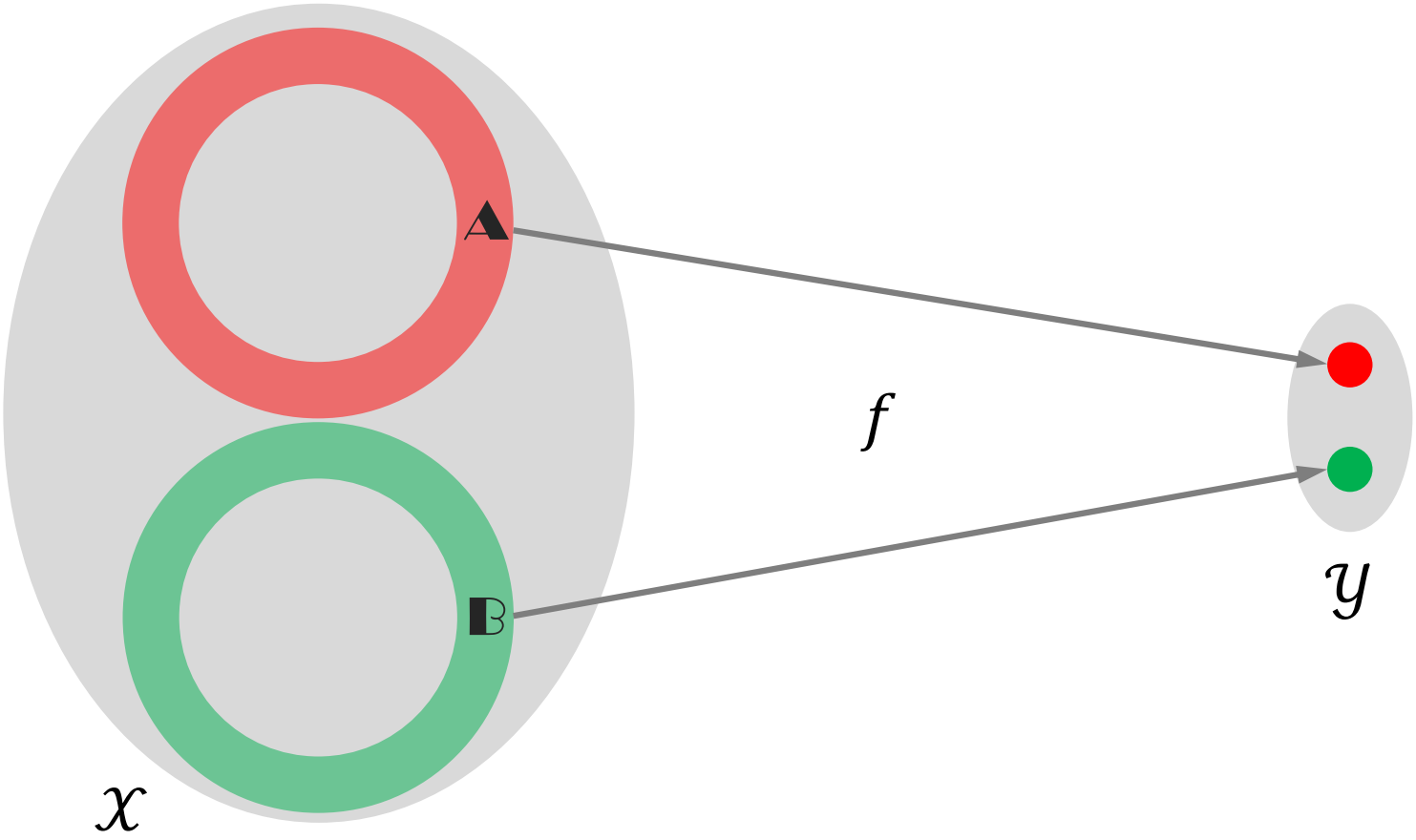
Symmetries of the Label Function



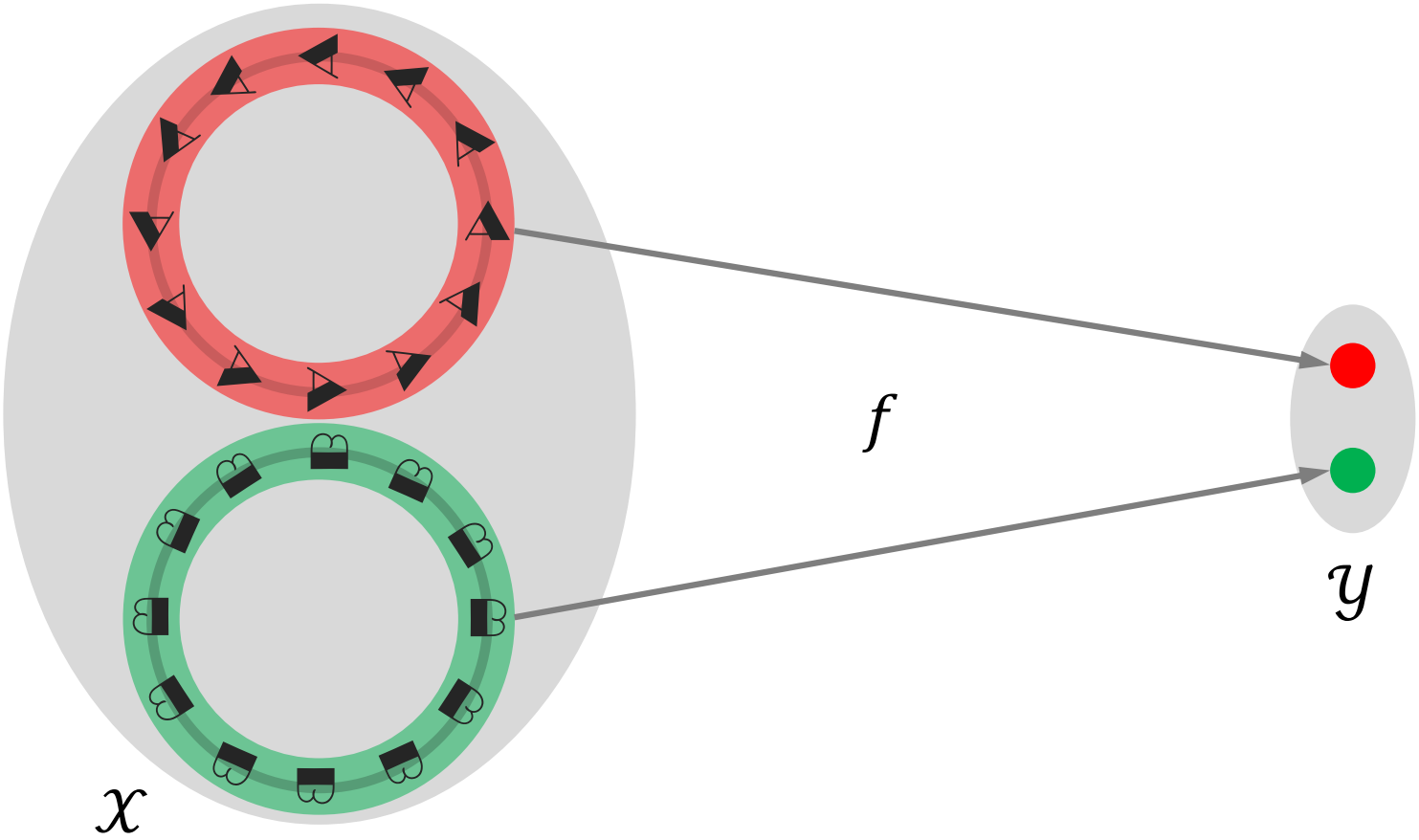
Symmetries of the Label Function



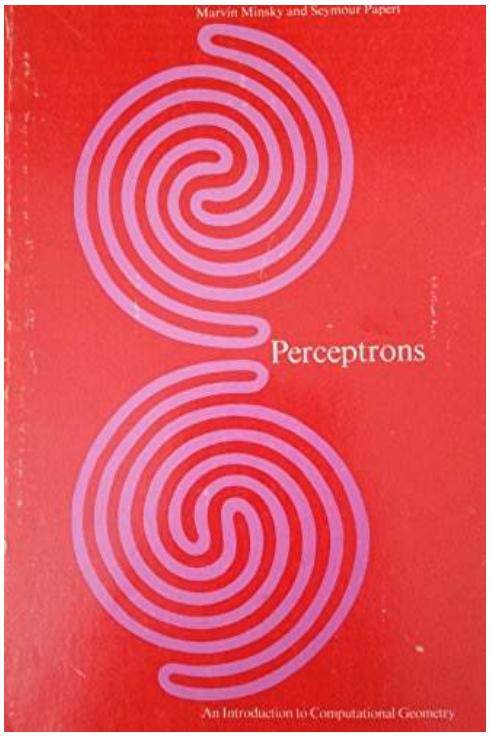
Symmetries of the Label Function



Symmetries of the Label Function



First "geometric" machine learning



M. Minsky

S. Papert

1969

Minsky, Papert 1969

First "geometric" machine learning

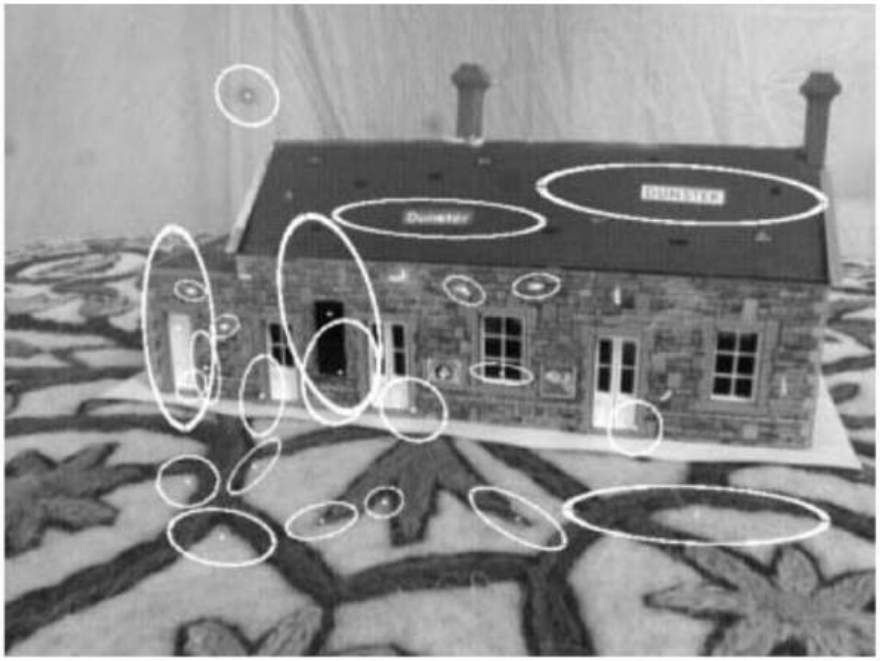
Group Invariance Theorem: "if a neural network is invariant to a group, then its output can be expressed as functions of the orbits of the group"

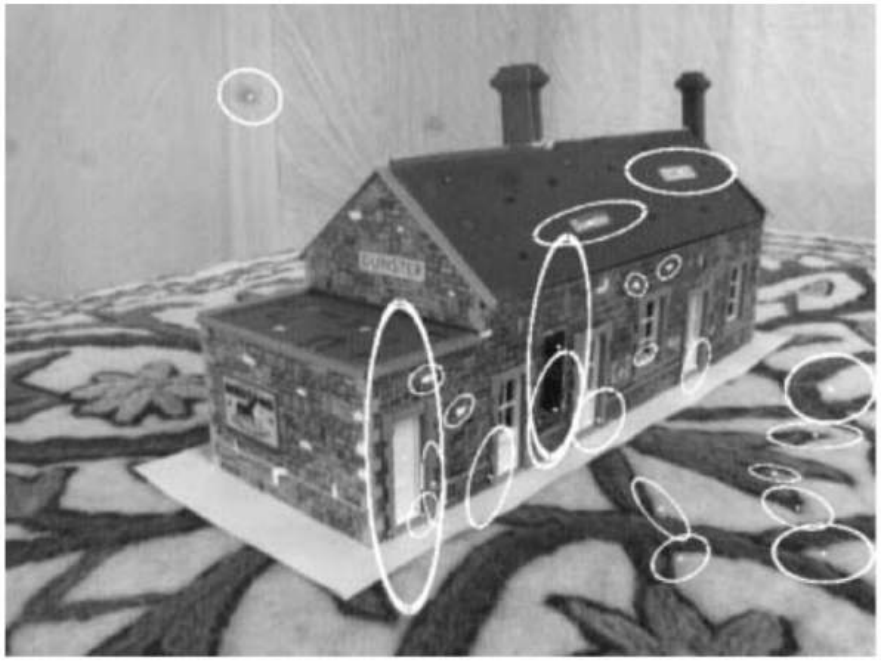
M. Minsky **S. Papert**

1969

Minsky, Papert 1969

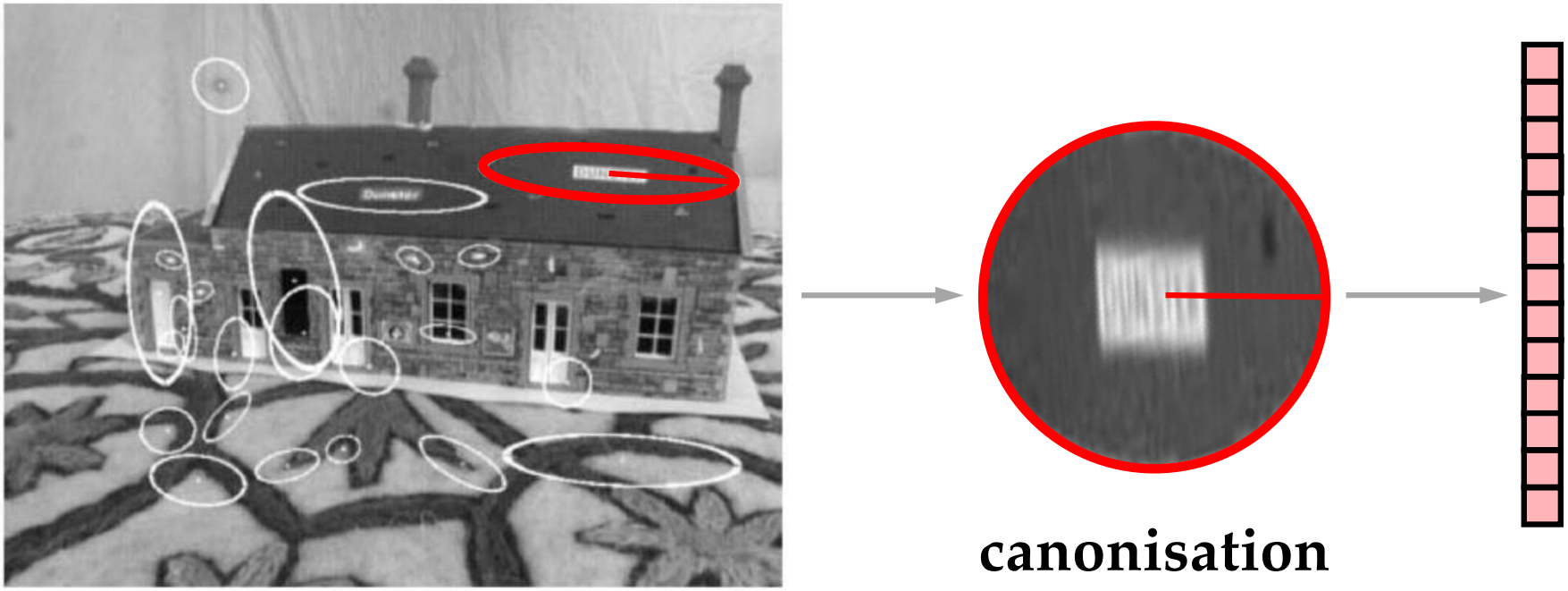
Canonisation



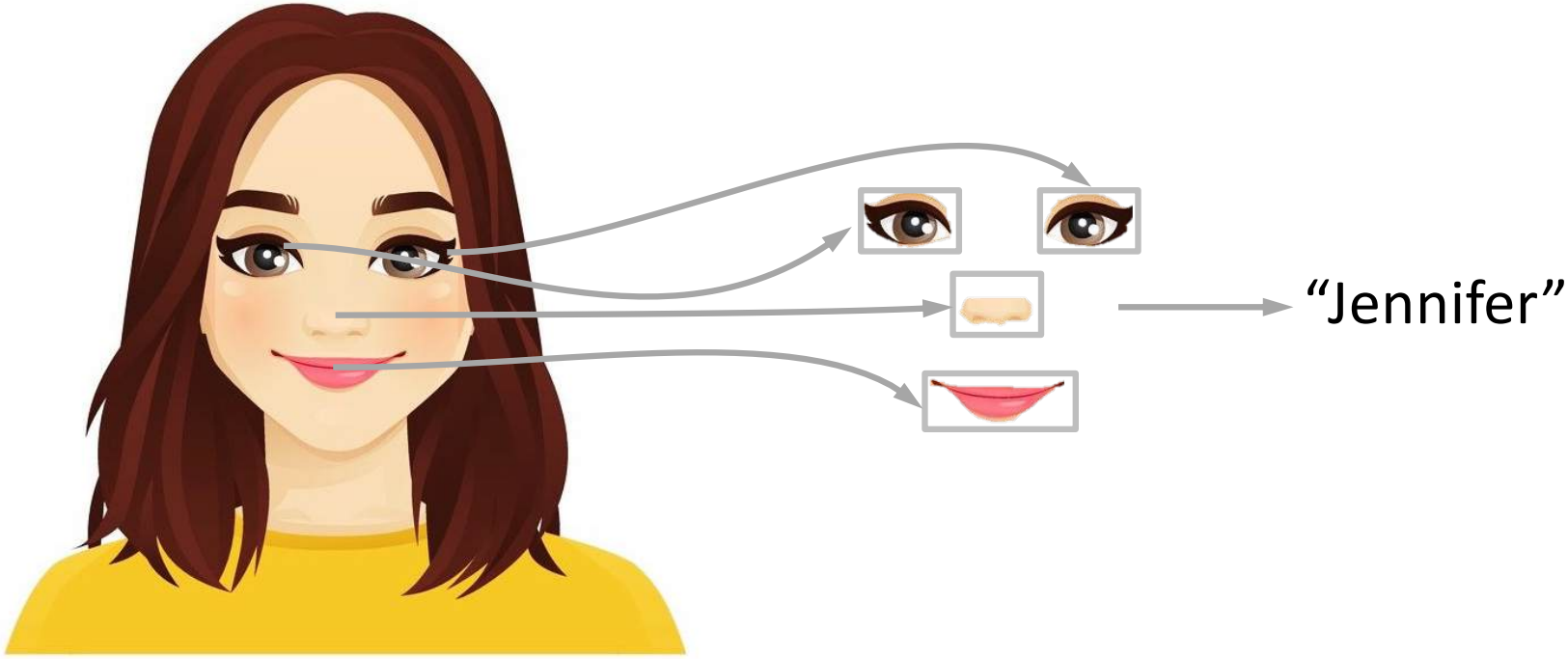


Mikolajczyk, Schmid 2004

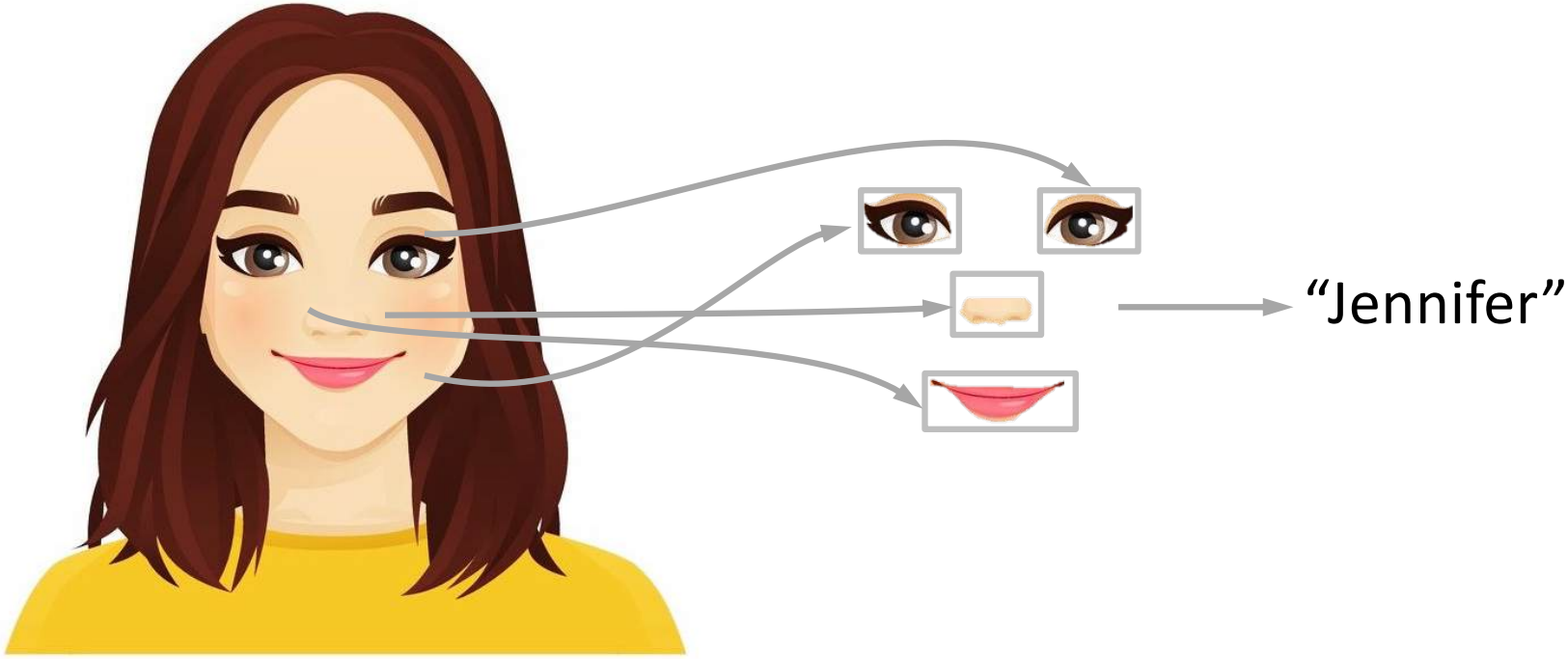
Canonisation



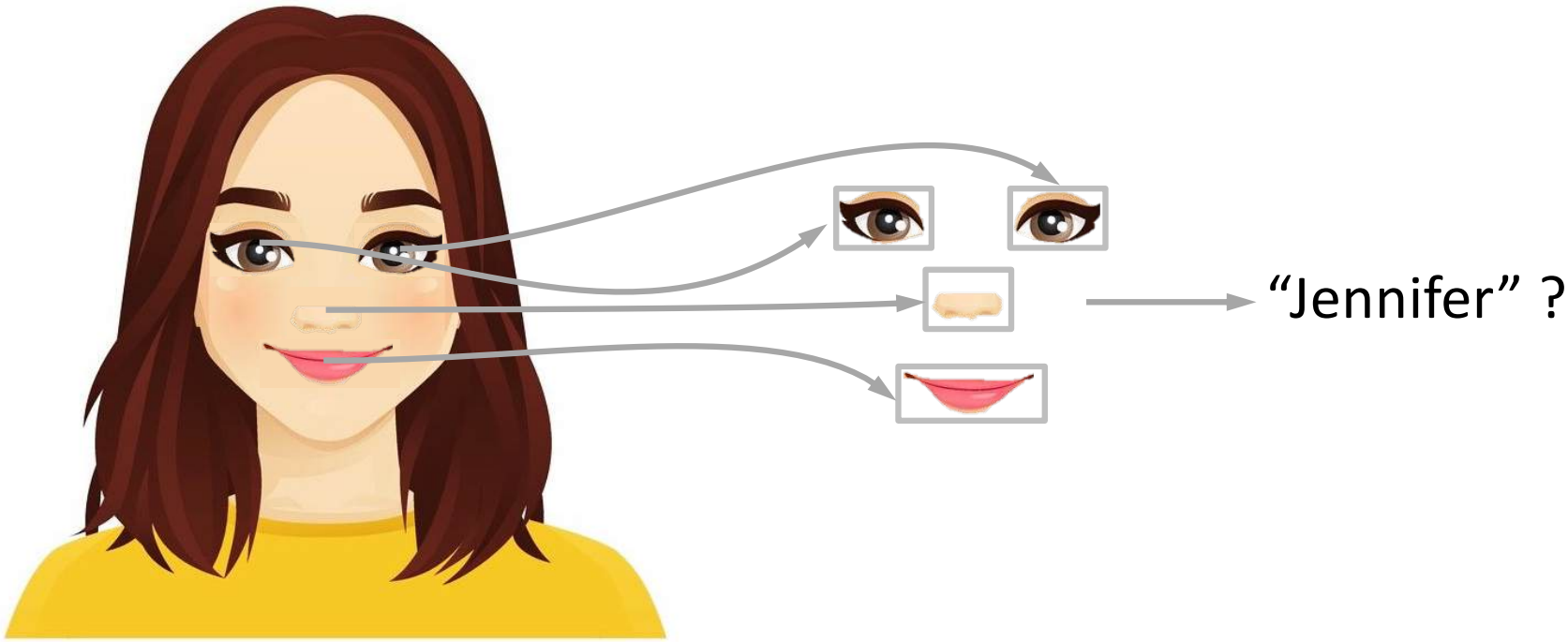
Canonisation

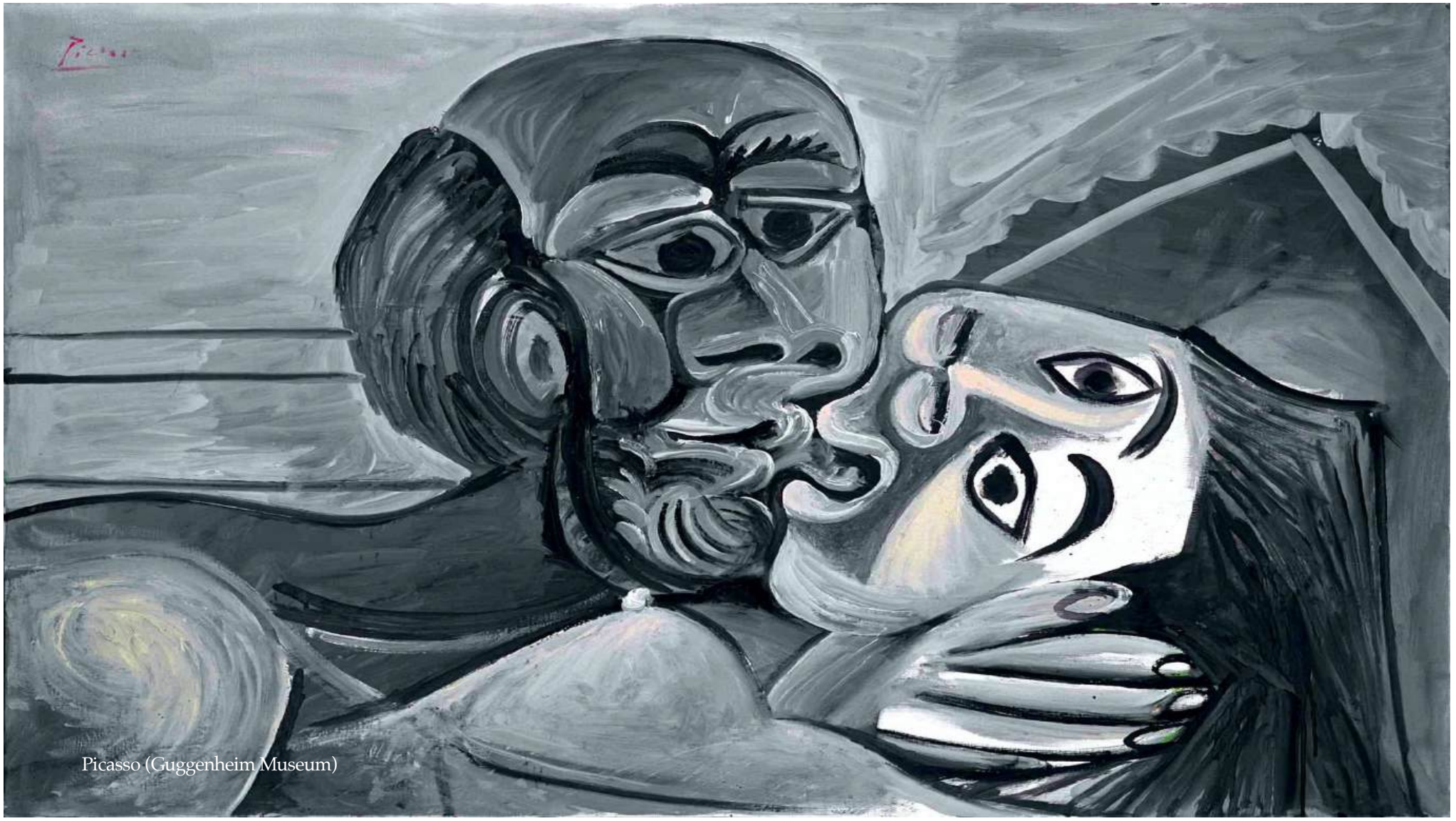


Canonisation



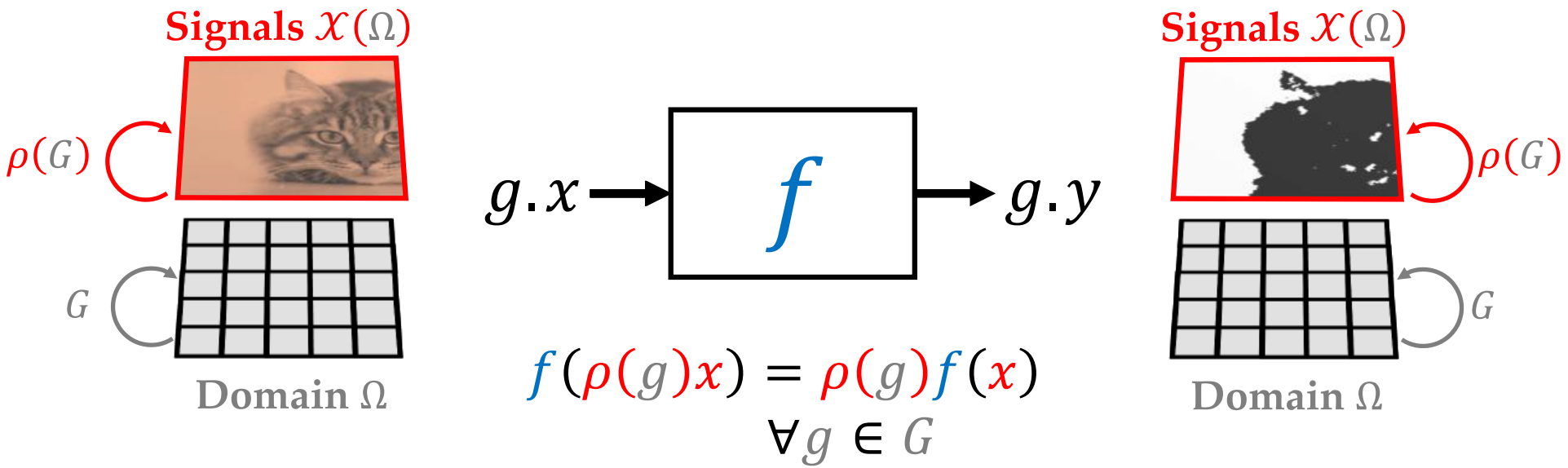
Canonisation



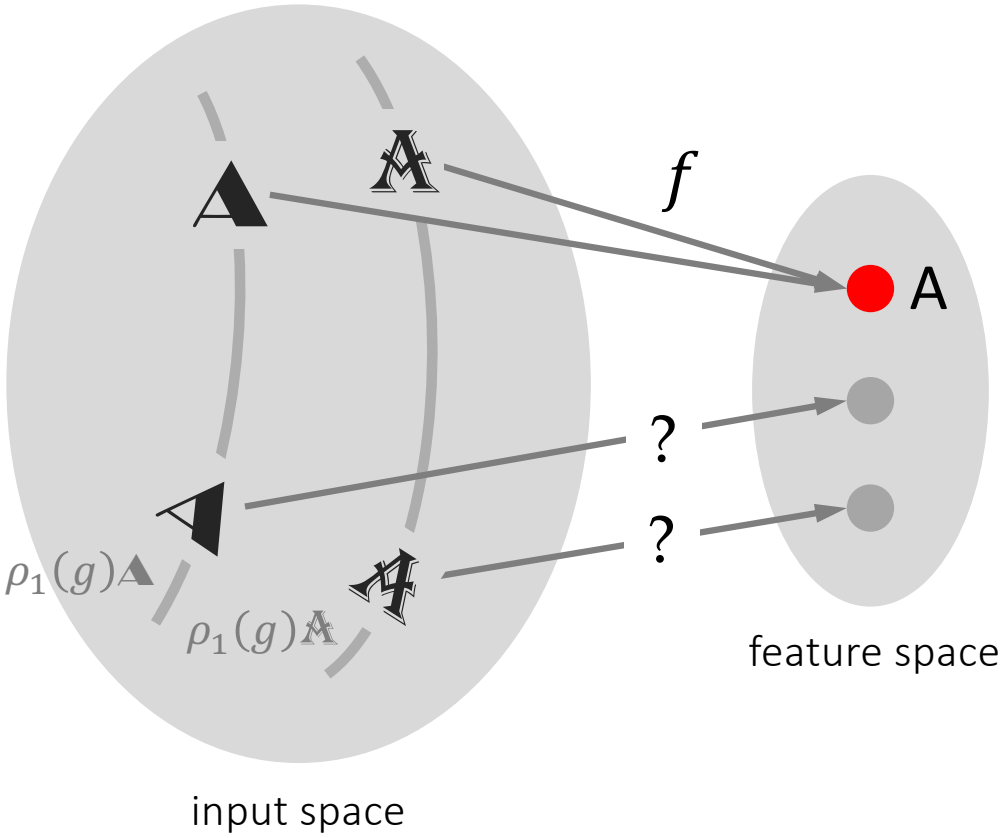


Picasso (Guggenheim Museum)

Geometric priors: Equivariance



Equivariance = Symmetry-consistent generalisation

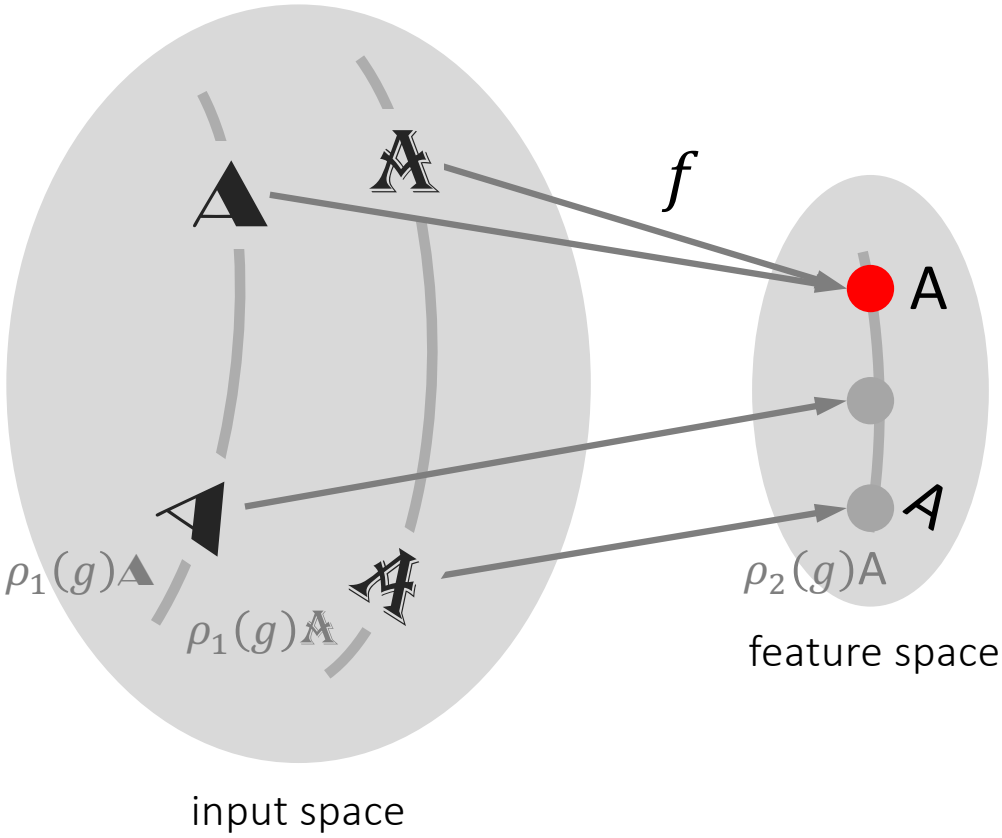


$$f(\rho_1(g)\mathbb{A}) = \rho_2(g)\, f(\mathbb{A})$$

$$\| \qquad \|$$

$$f(\rho_1(g)\mathbb{A}) = \rho_2(g)\, f(\mathbb{A})$$

Equivariance = Symmetry-consistent generalisation

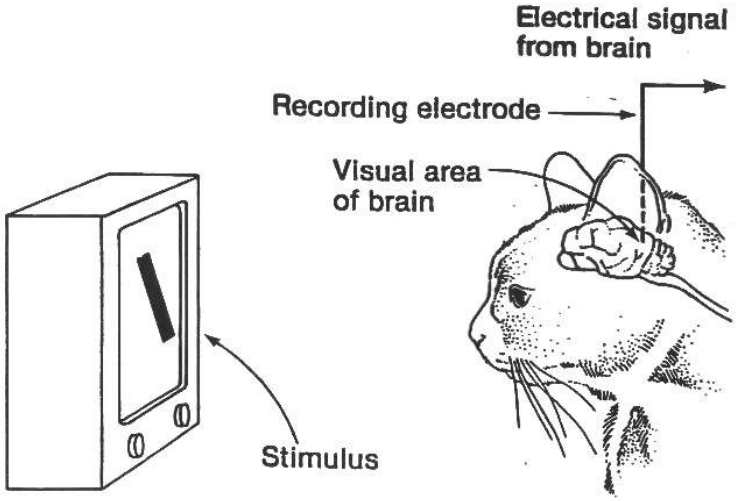


$$f(\rho_1(g)\mathbf{A}) = \rho_2(g)\, f(\mathbf{A})$$

$$\| \qquad \|$$

$$f(\rho_1(g)\mathbf{A}) = \rho_2(g)\, f(\mathbf{A})$$

Early Geometric Architectures

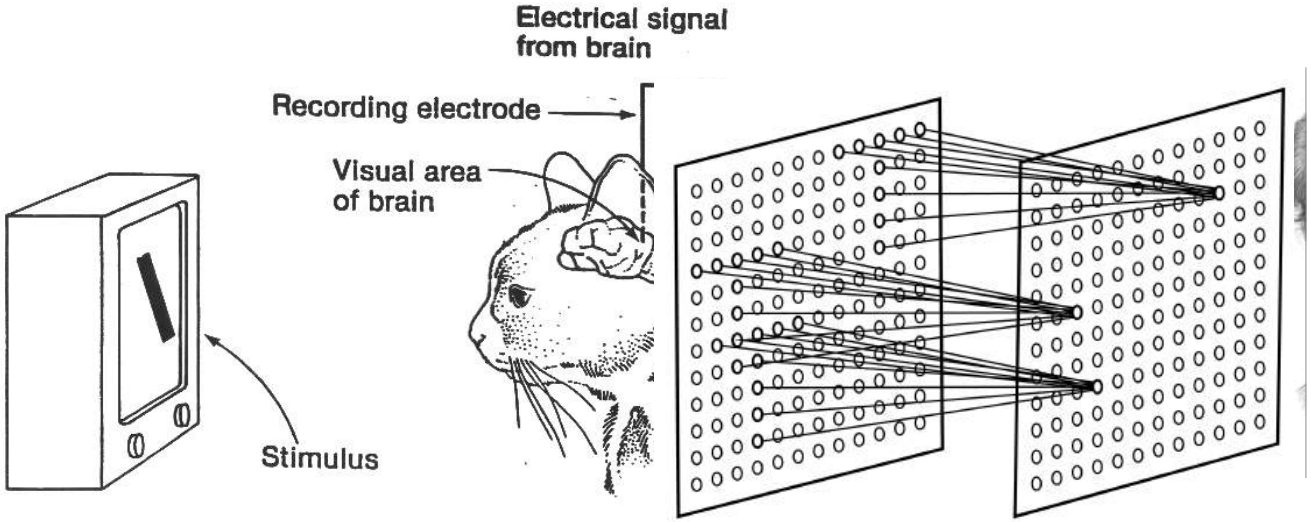


D. Hubel **T. Wiesel**

1959

Hubel, Wiesel 1959, 1962; Portraits: Ihor Gorskyi

Early Geometric Architectures



K. Fukushima

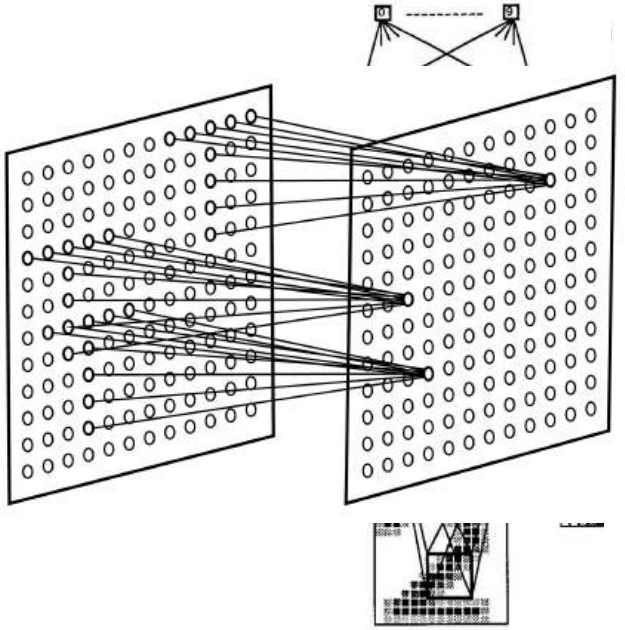
1959 1980

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

Early Geometric Architectures

D. Hubel **T. Wiesel**

1959

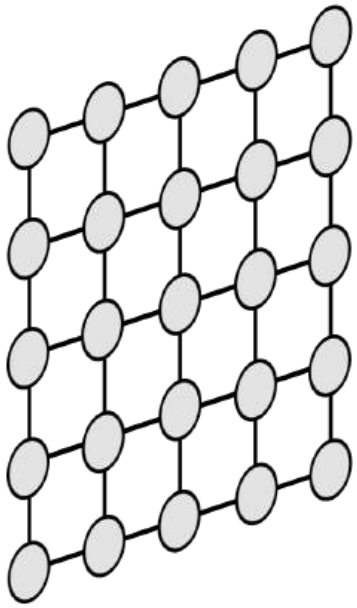


K. Fukushima

1989

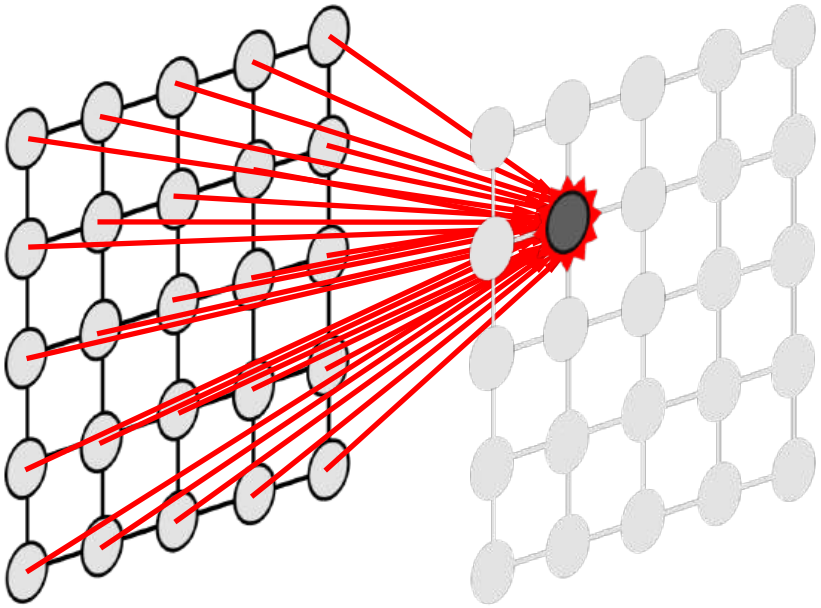
Hubel, Wiesel 1959, 1962; Fukushima 1980; LeCun et al. 1989; Portraits: Ihor Gorskyi

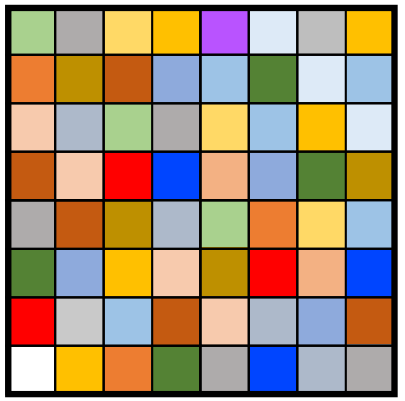
Multi-Layer Perceptron



LeCun et al. 1989

Multi-Layer Perceptron





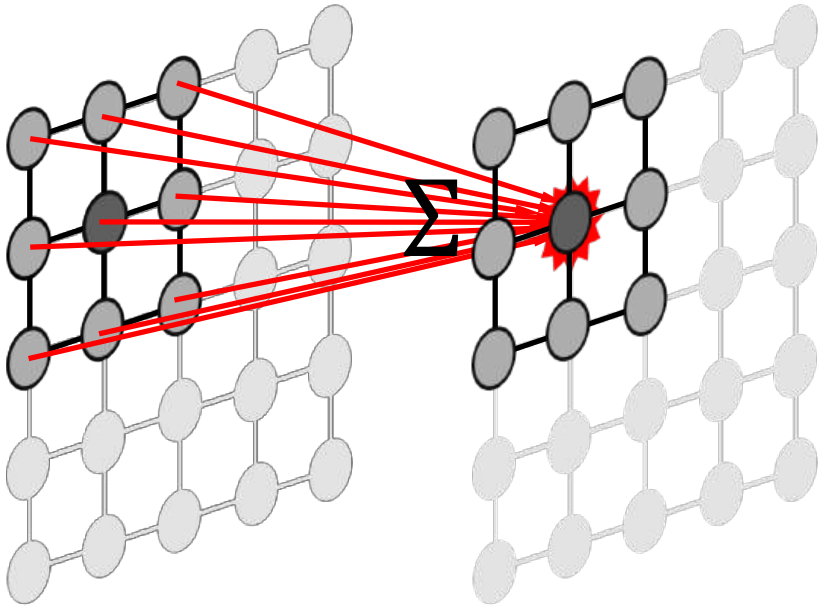
weight matrix

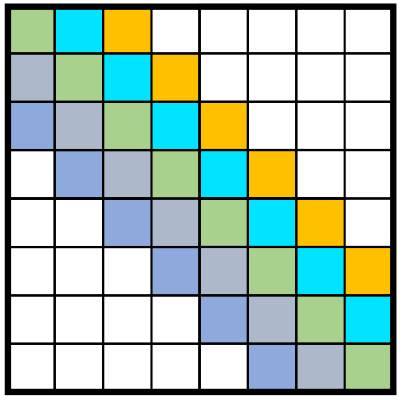
$\mathcal{O}(n^2)$ DOF

Fully connected layer

LeCun et al. 1989

Convolutional Neural Networks





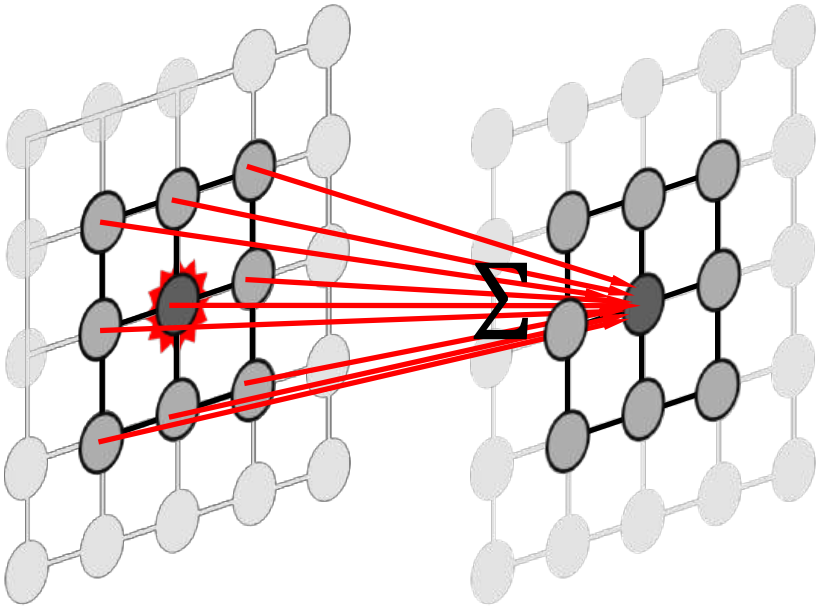
weight matrix

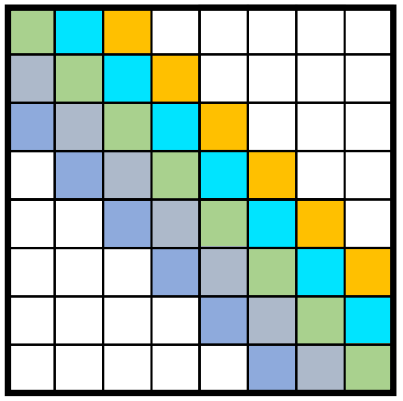
$\mathcal{O}(1)$ DOF

Locality + Shared parameters

LeCun et al. 1989

Convolutional Neural Networks





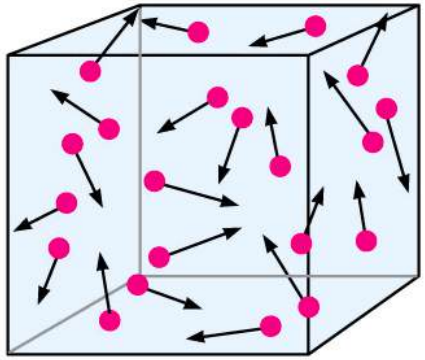
weight matrix

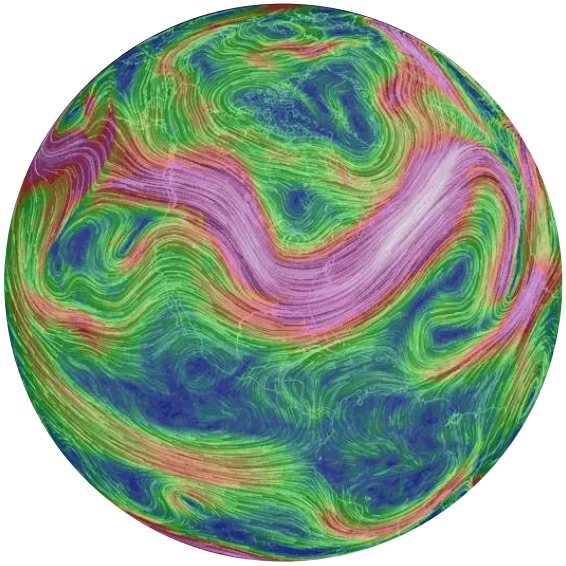
$\mathcal{O}(1)$ DOF

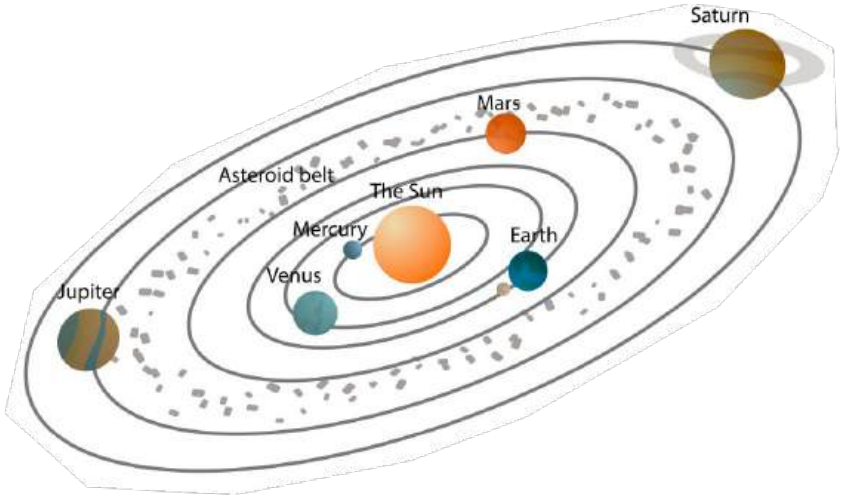
Locality + Shared parameters

LeCun et al. 1989

Scale Separation

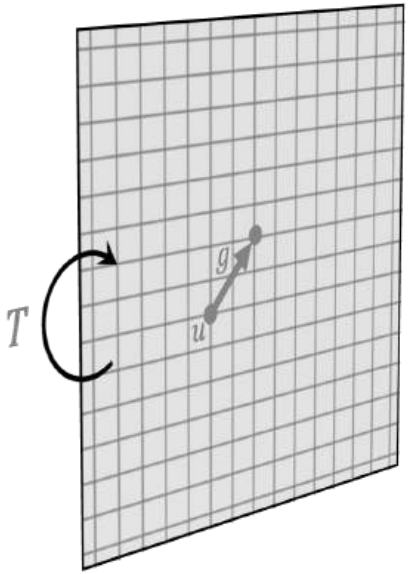






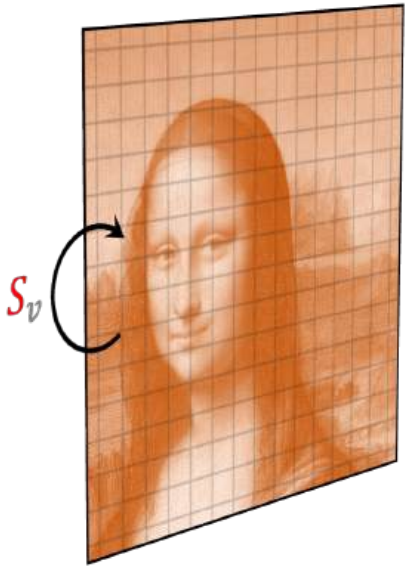
Convolutional Neural Networks

Plane $\mathbb{R}^2$



Translation group $T(2)$

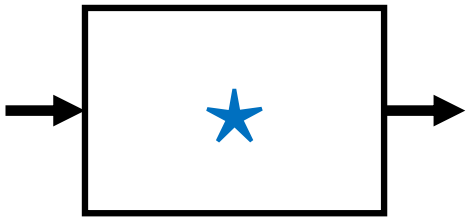
Images $\mathcal{X}(\mathbb{R}^2)$



Shift operator S

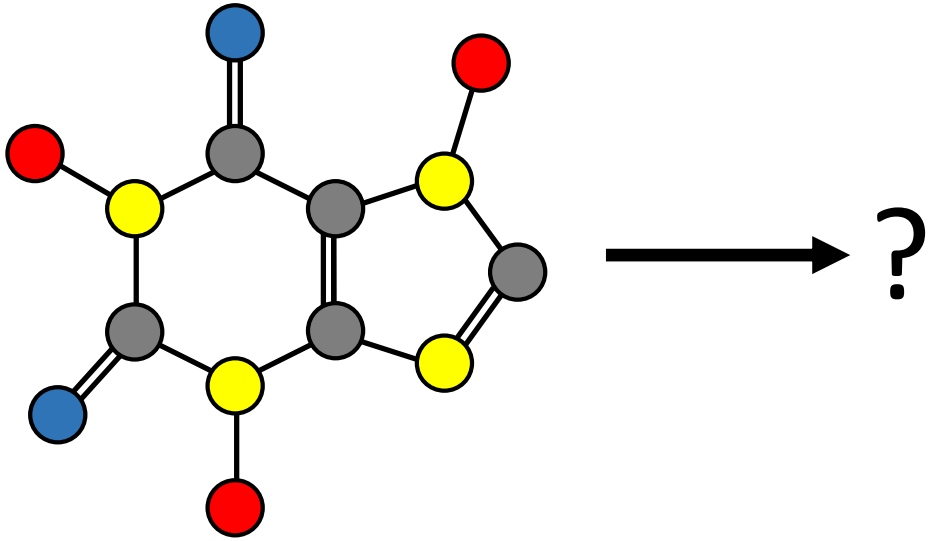
$$S_v x(u) = x(u - v)$$

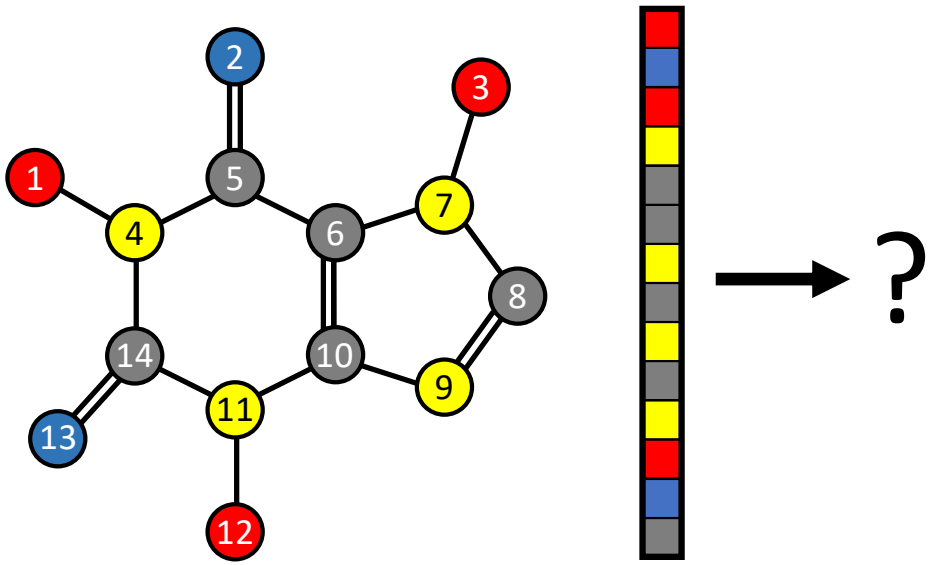
Functions $\mathcal{F}(\mathcal{X}(\mathbb{R}^2))$

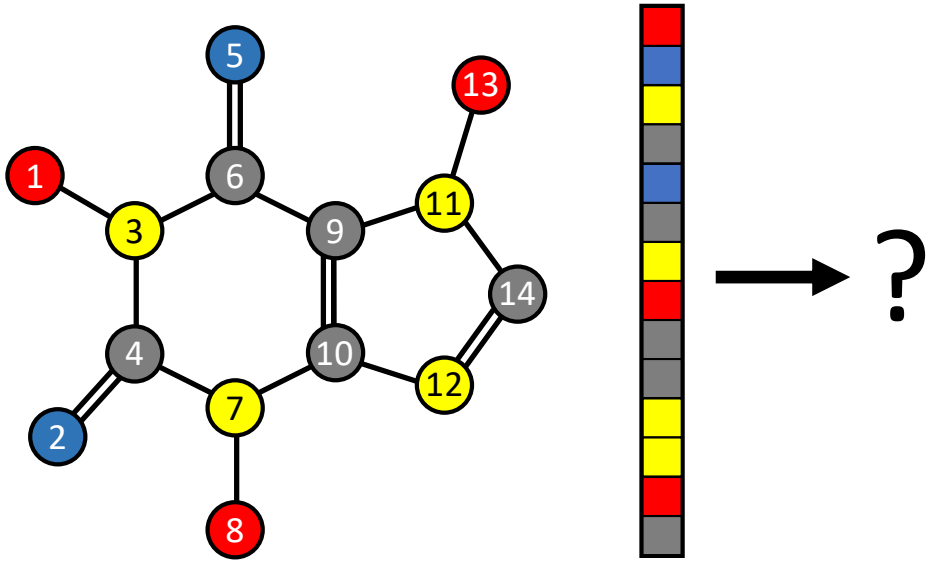


Convolution

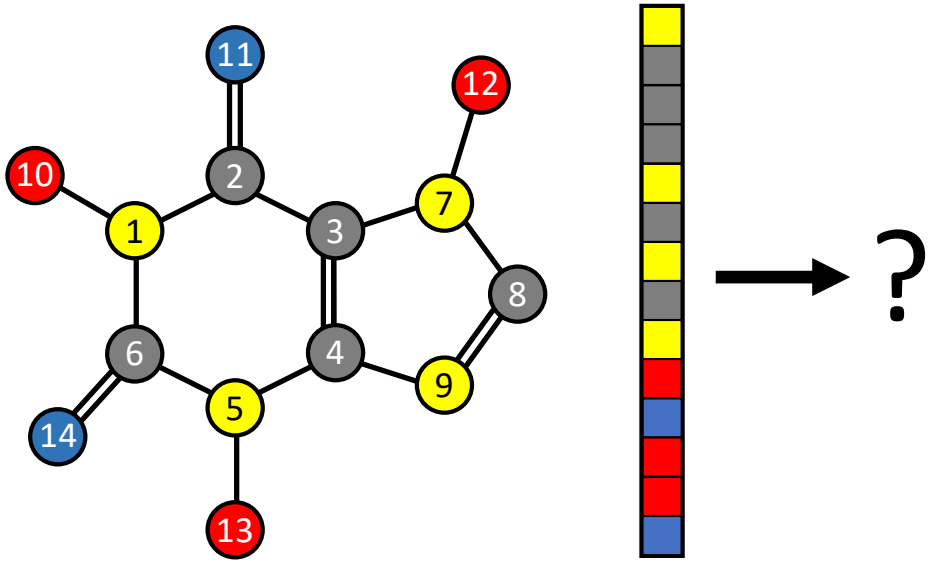
$$(Sx \star y) = S(x \star y)$$




1
2
3
4
5
6
7
8
9
10
11
12
13
14
?

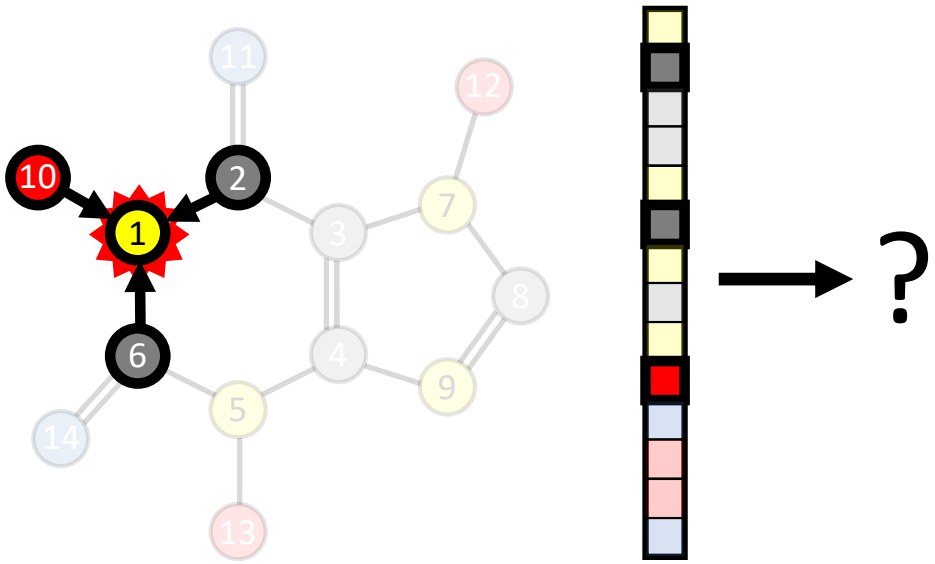

1
2
3
4
5
6
7
8
9
10
11
12
13
14
?

Permutation Invariance



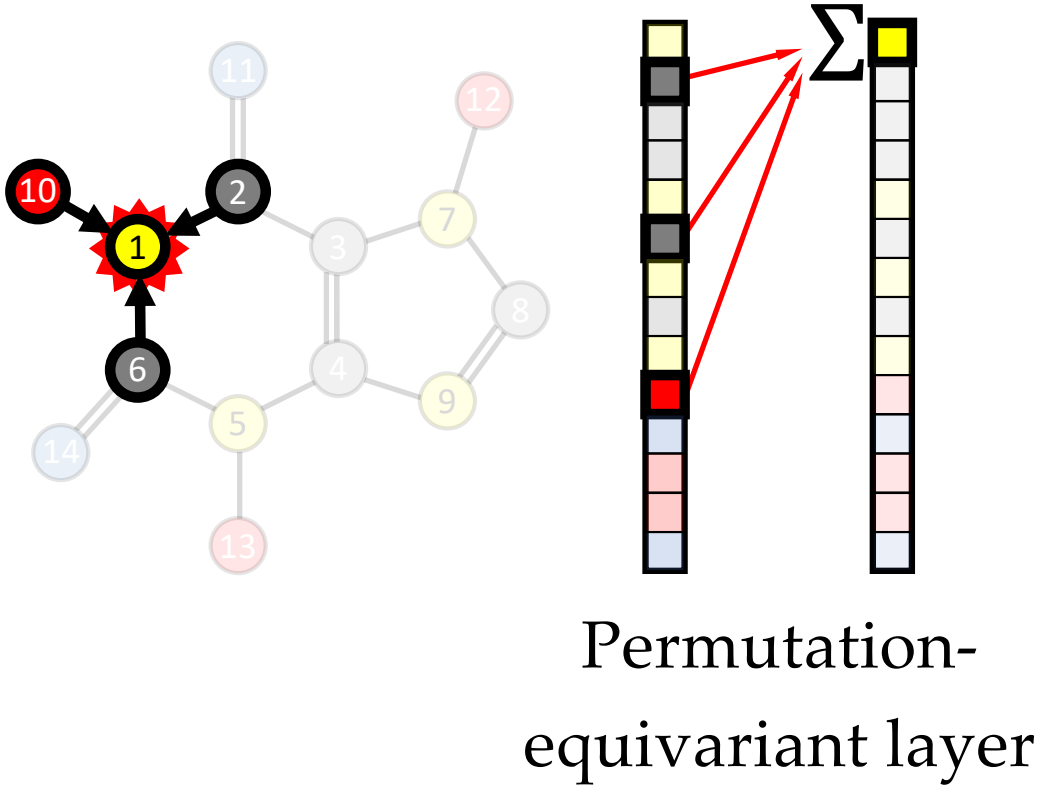
"properties of a molecule do not change if we reorder the atoms"

Graph Neural Networks



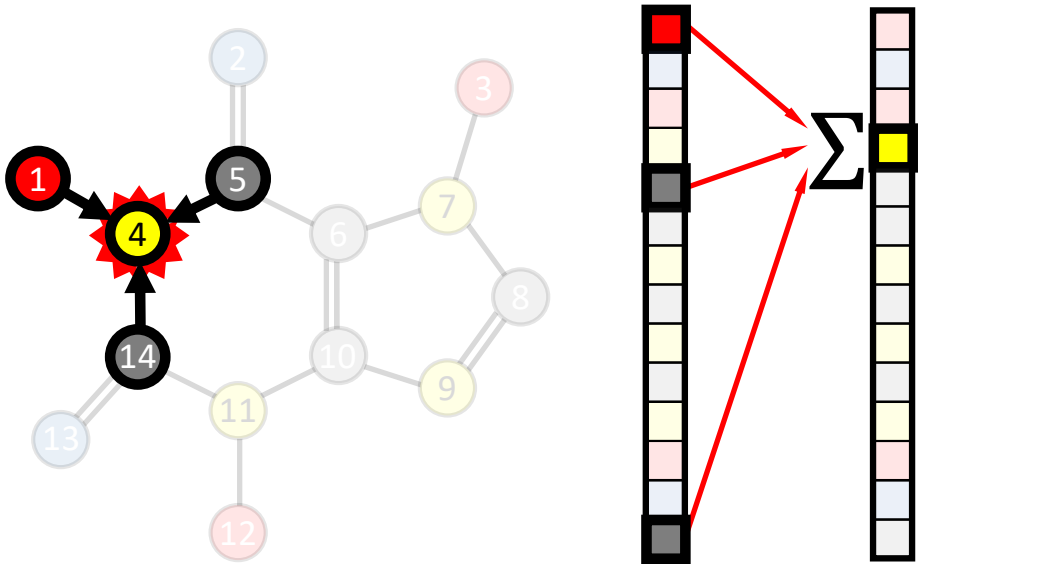
Locality + Shared parameters

Graph Neural Networks



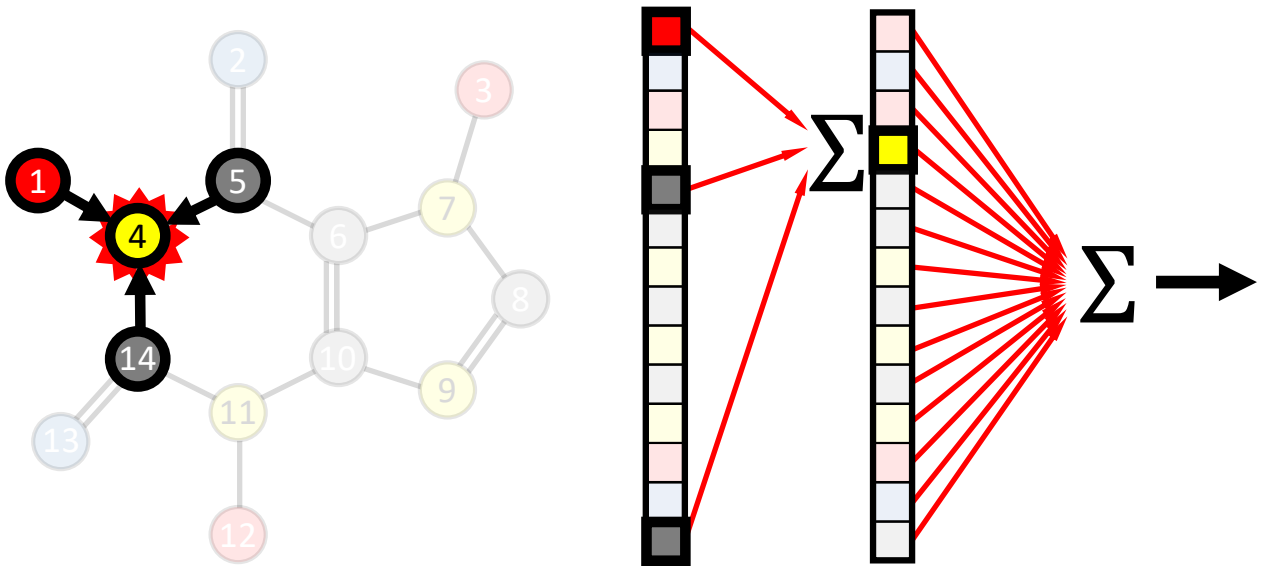
Permutation-
equivariant layer

Graph Neural Networks



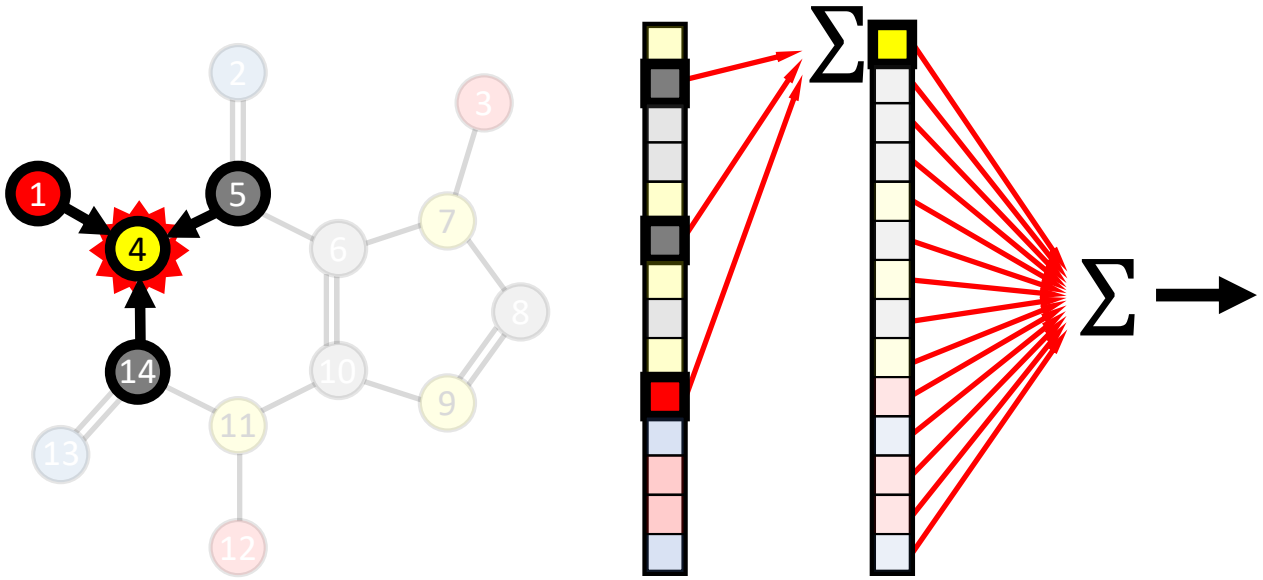
Permutation-
equivariant layer

Graph Neural Networks



Permutation-
invariant readout

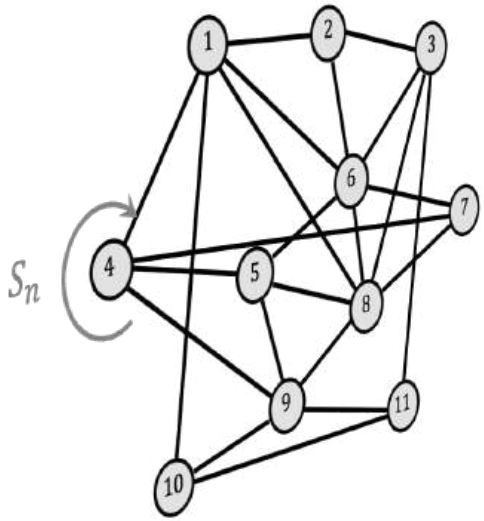
Graph Neural Networks



Permutation-
invariant readout

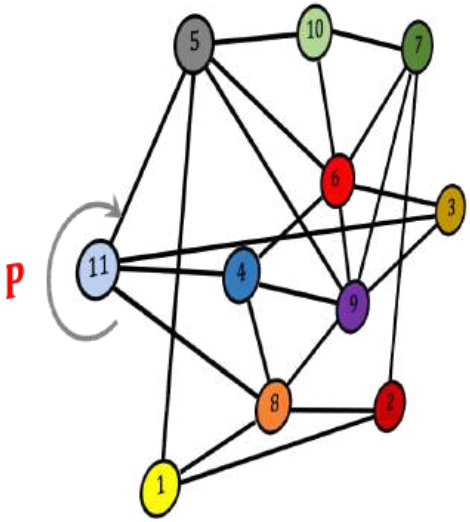
Graph Neural Networks

Graph $G = (V, E)$



Permutation group S_n

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



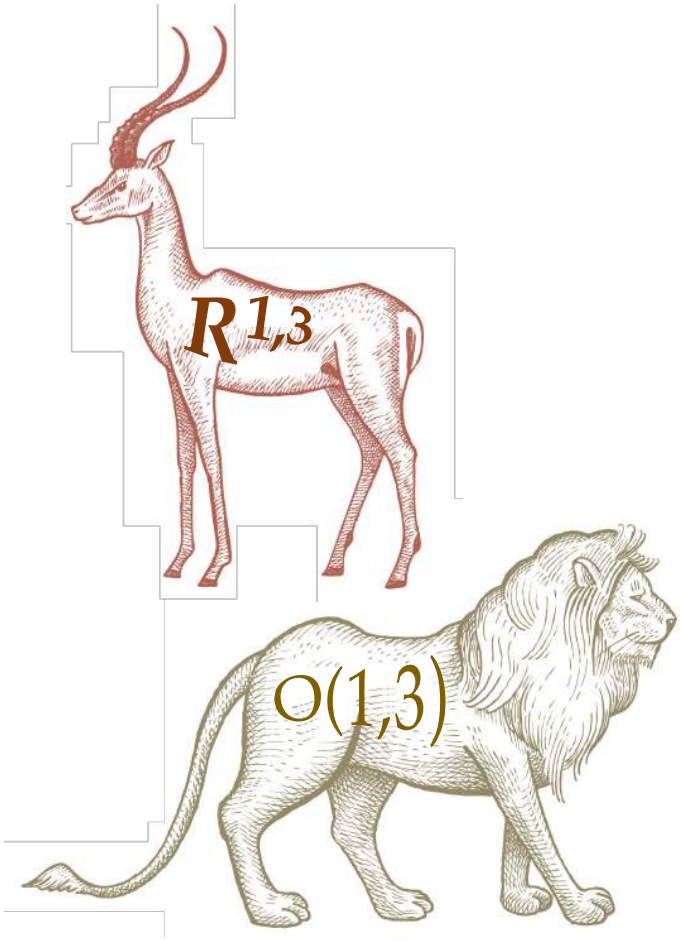
Permutation matrix P

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

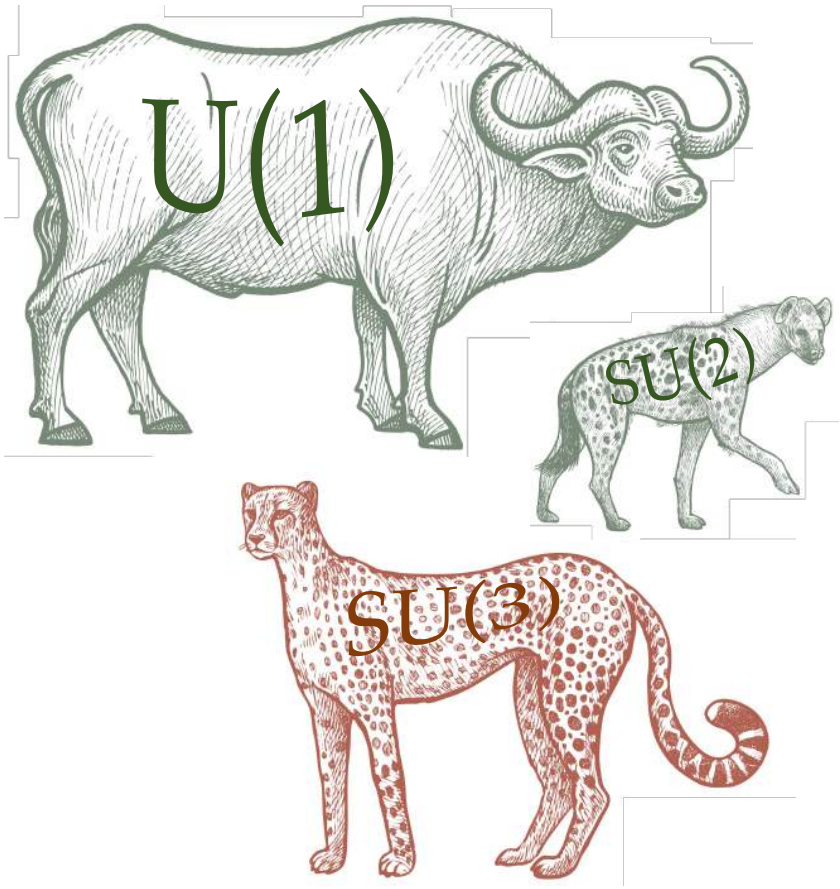
Functions $\mathcal{F}(\mathcal{X}(G))$

Message passing

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

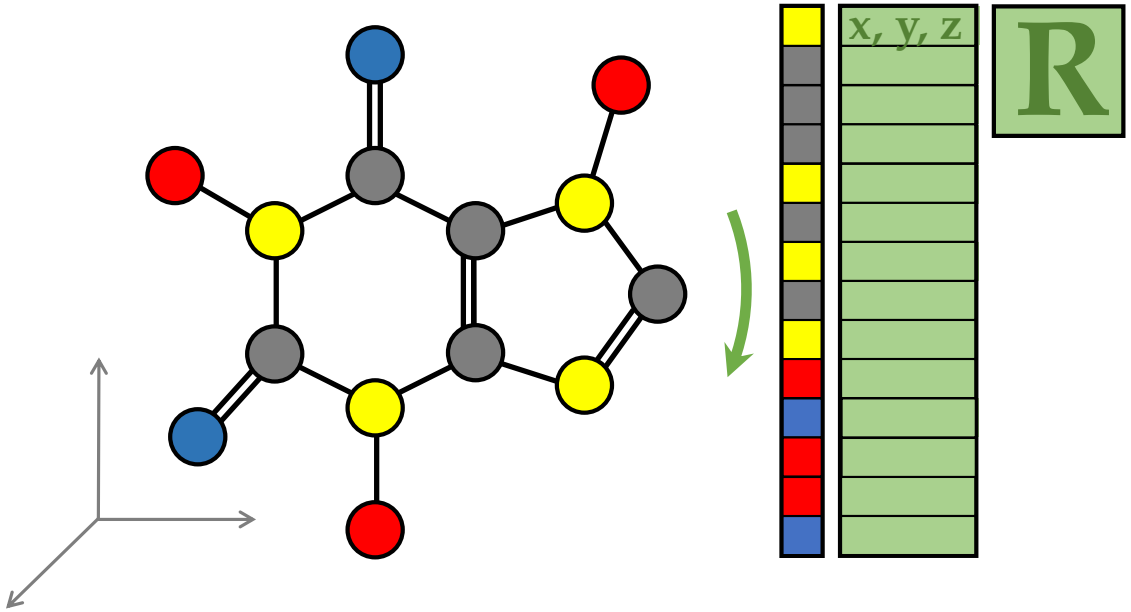


External symmetry



Internal symmetry

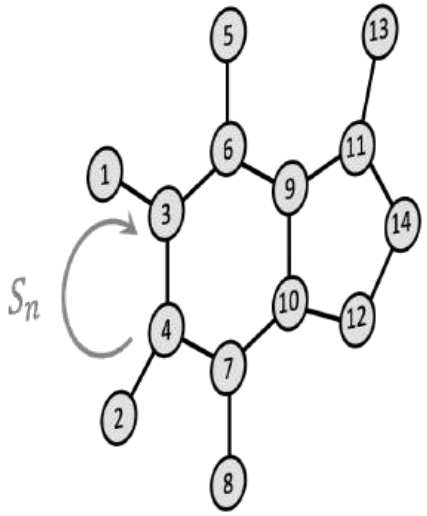
SO(3)-*invariance*



"properties of a molecule do not change if we rotate it"

Geometric ("Equivariant") Graph Neural Networks

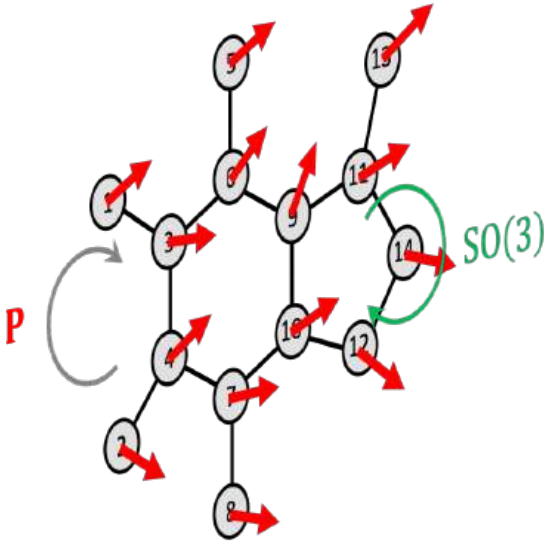
Geometric Graph G



Permutation group S_n

"domain symmetry"

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



Permutation matrix P

Rotation R

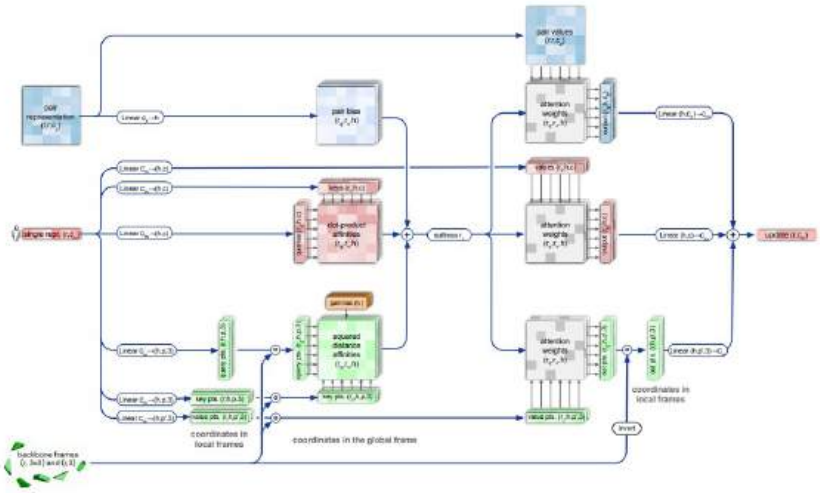
"data symmetry"

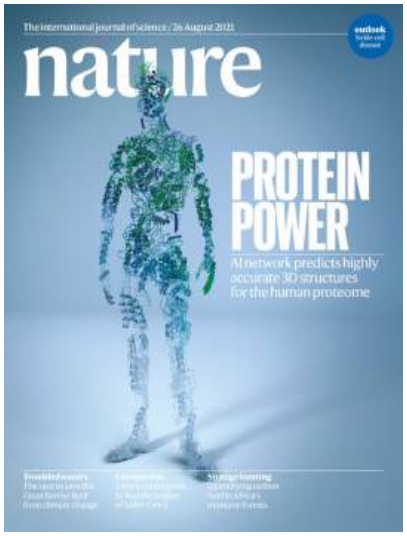
Functions $\mathcal{F}(\mathcal{X}(G))$

Geometric message passing

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

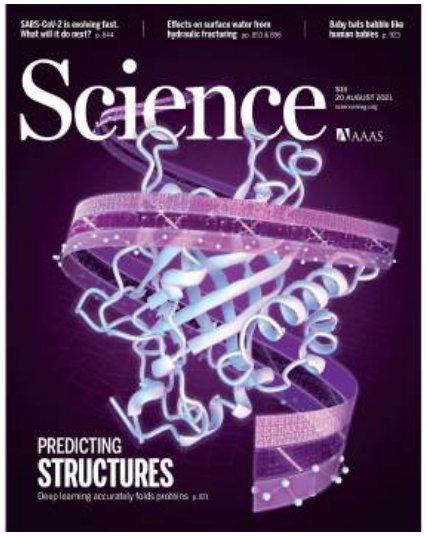
Revolution in Structural Biology





Jumper et al. 2021

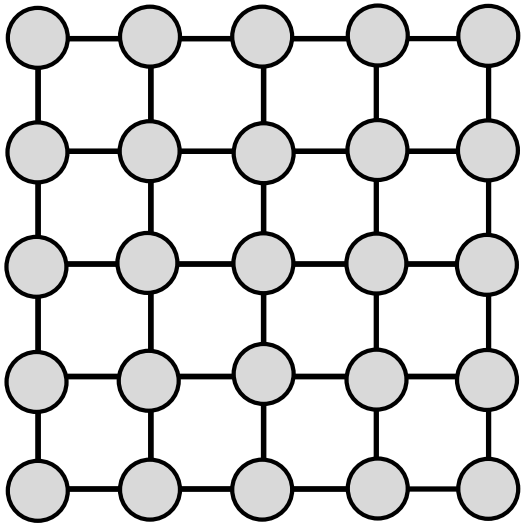
AlphaFold 2
"Invariant point attention"



Baek et al. 2021

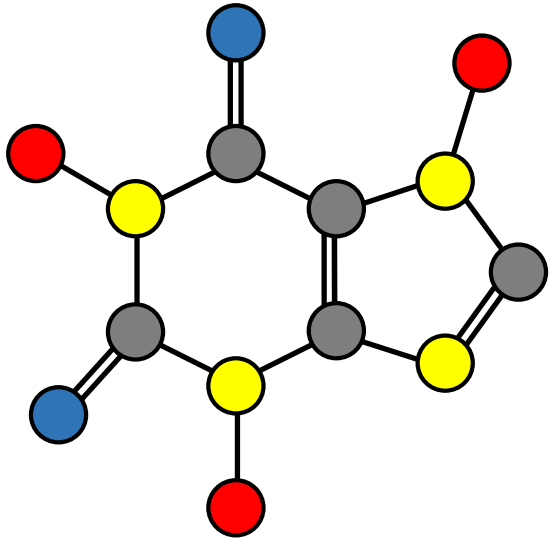
RosettaFold
SE(3)-equivariant Transformer

Grids



Translation

Graphs



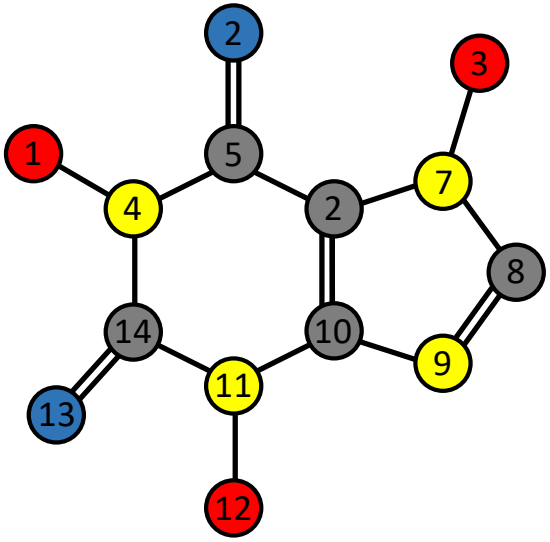
Permutation

Meshes

Local Rotation

$$+\ T(2)\ =\ \text{CNN}$$

B et al 2017; 2021

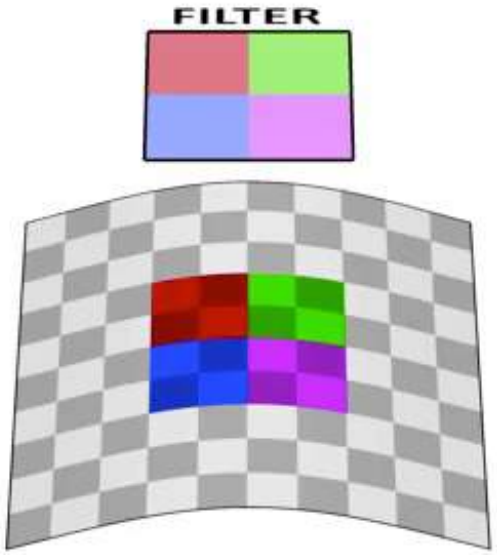


$$+ \quad S_n \quad = \text{GNN}$$

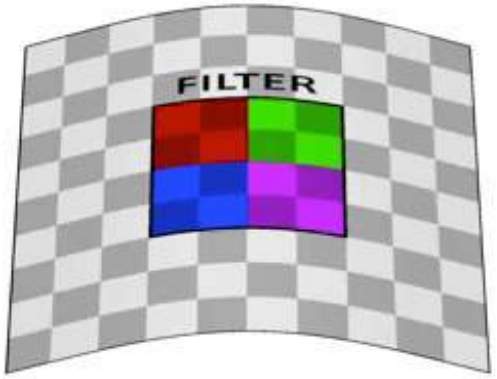
B et al 2017; 2021

+ SO(2) = MeshCNN

B et al 2017; 2021

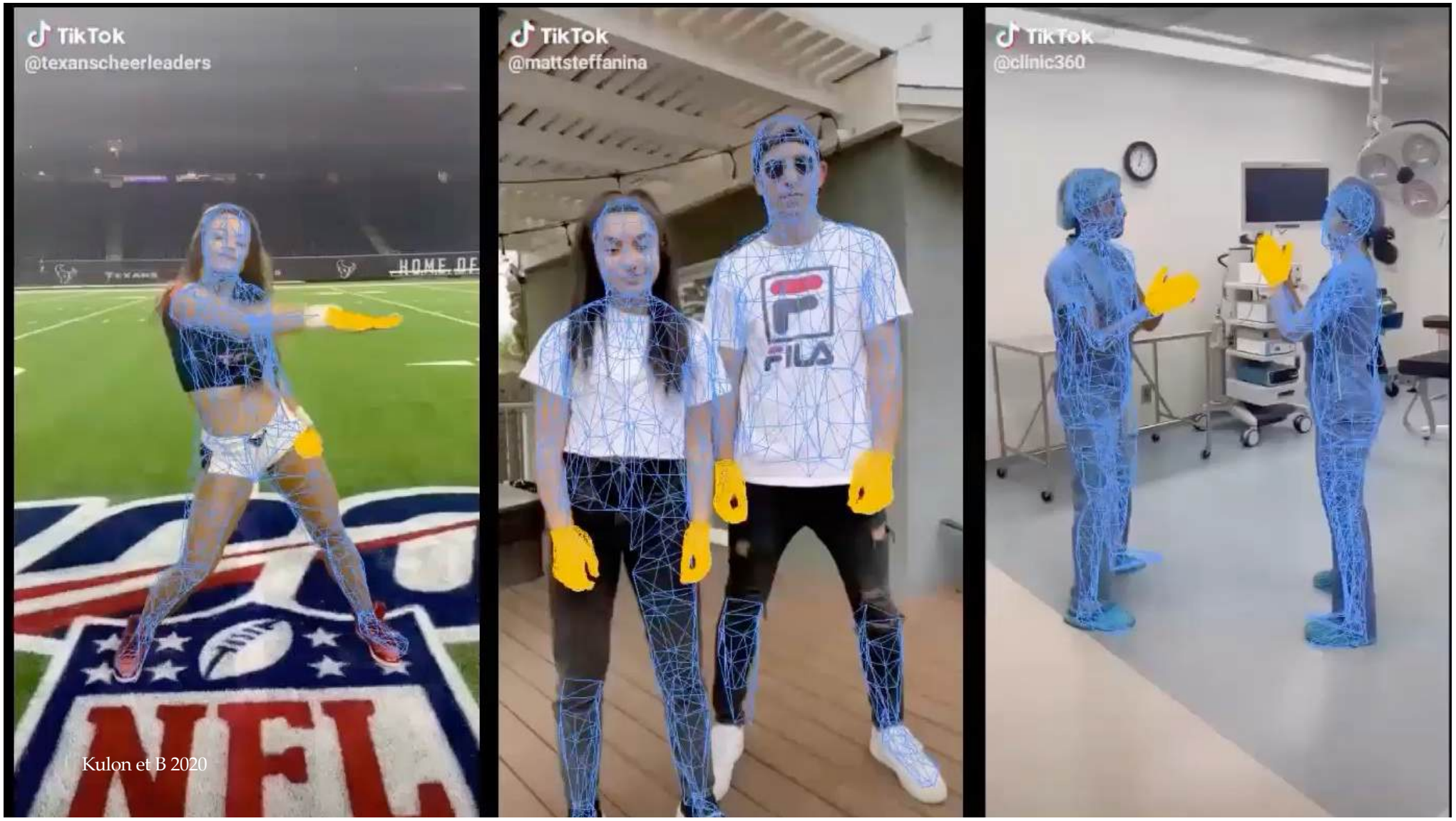


Euclidean (extrinsic) convolution



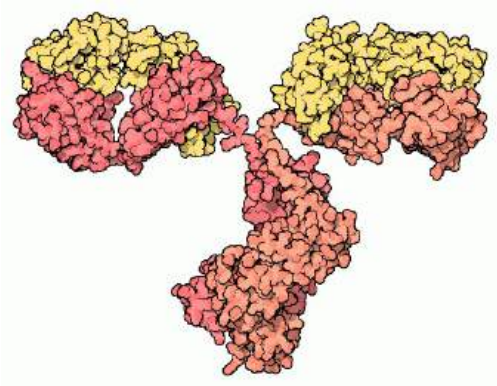
Geometric (intrinsic) convolution

Masci et B 2015; Monti et B 2017

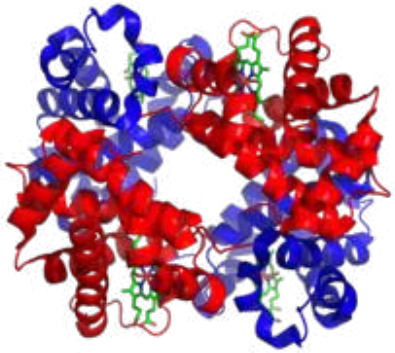


Kulon et B 2020

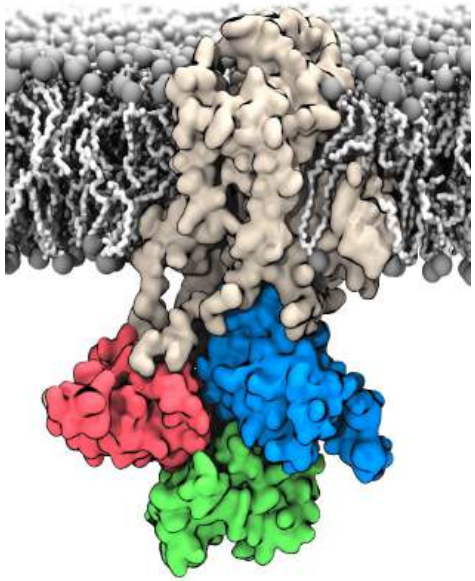
Snap Acquires Ariel AI To Enhance AR Features



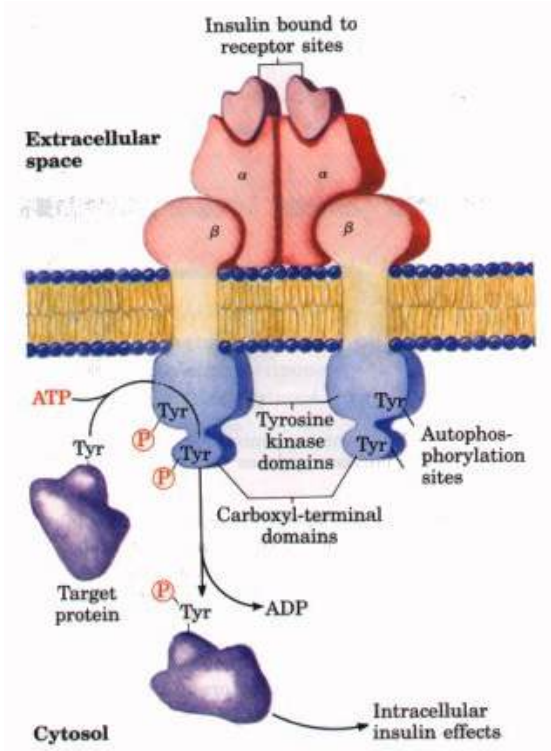
Defense (antibody)



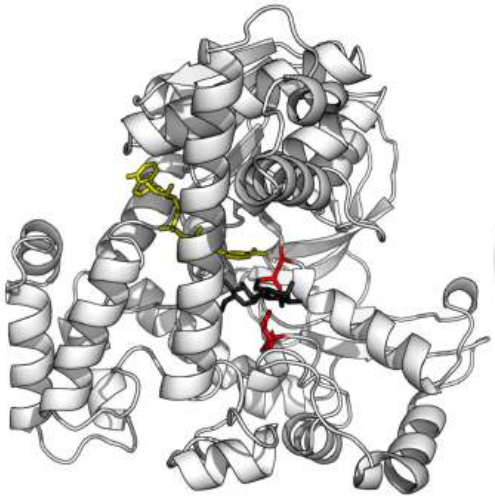
Storage (haemoglobin)

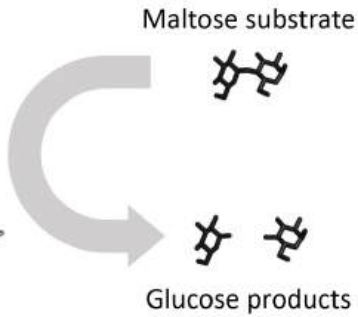


Transport (calcium pump)



Communication (insulin)



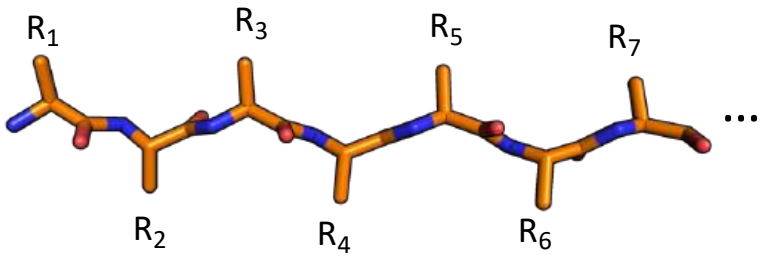


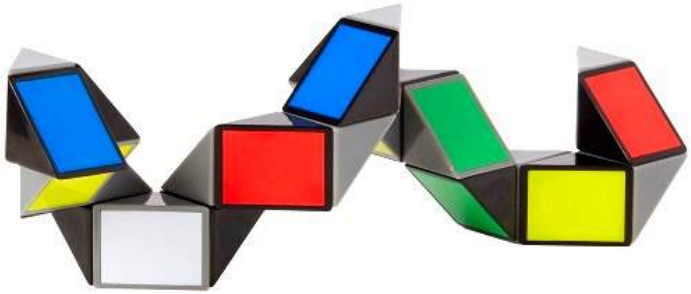
Catalysis (enzyme)

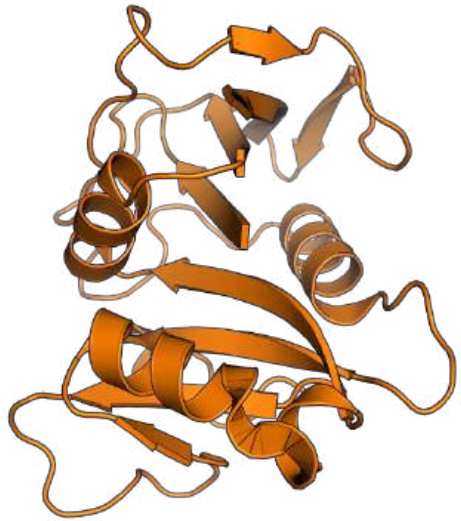
Structure (collagen)

Protein Folding

Protein folding



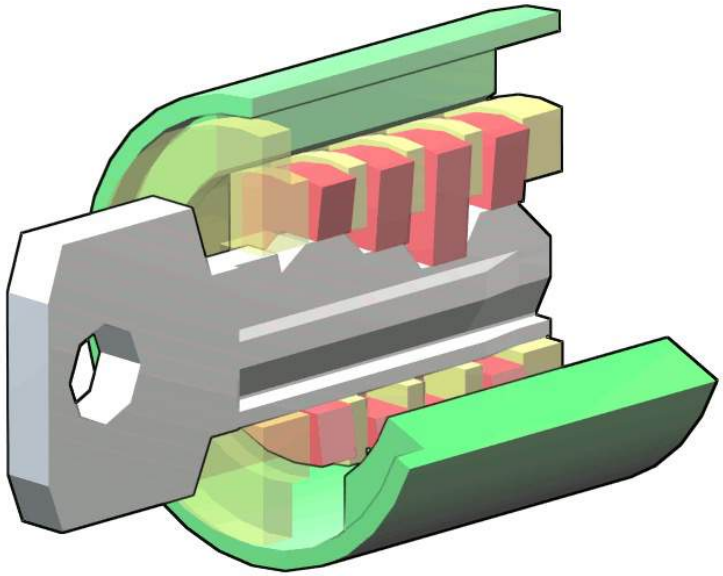


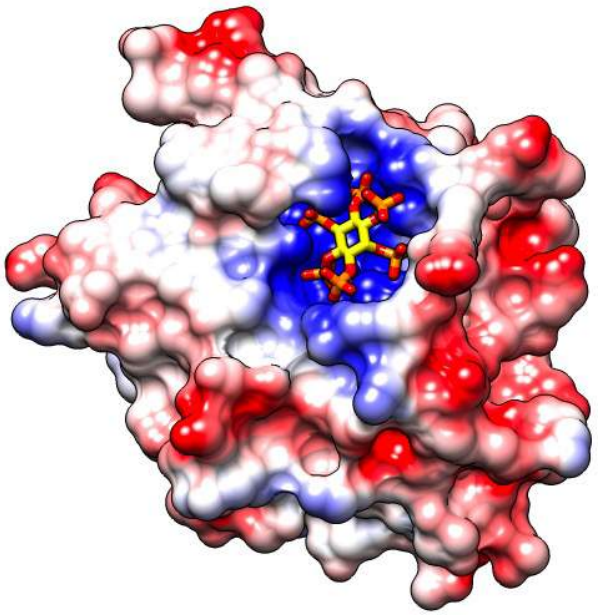


Sequence of amino acids

3D structure

Lock-Key Metaphor



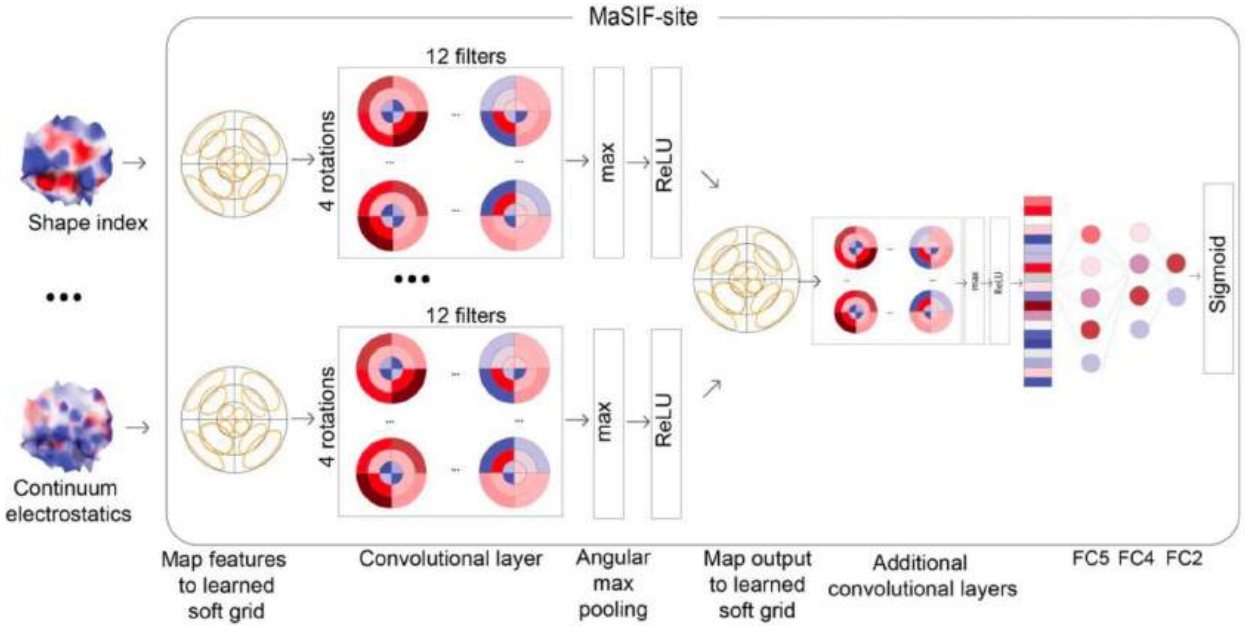


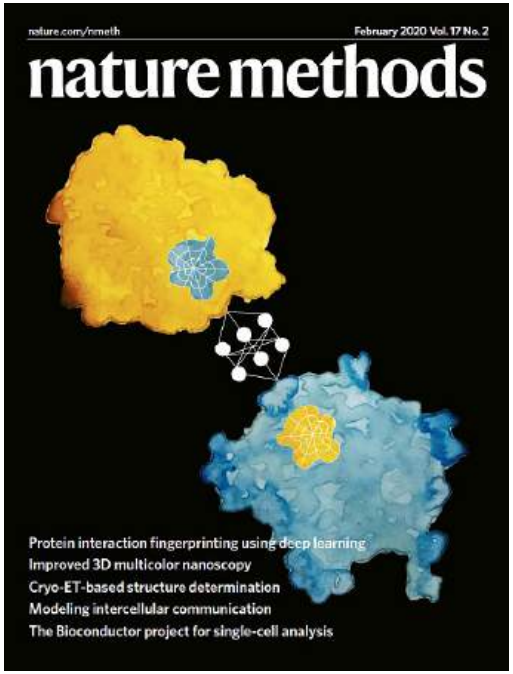
Emil Fischer “Schlüssel-Schloss-Prinzip” 1894

Surface-based Protein Representation

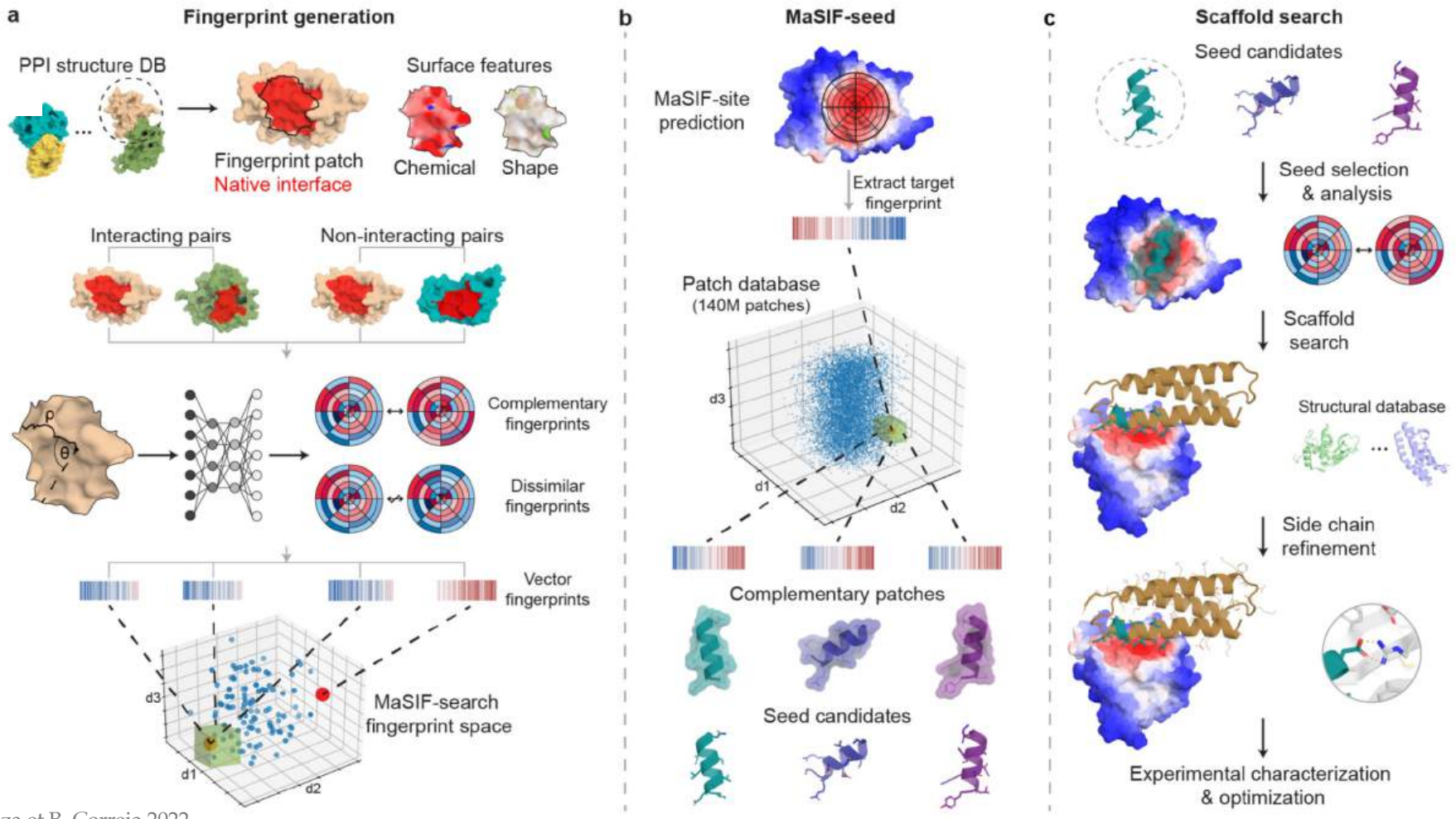
Abstract out internal structure that is irrelevant for interactions + allow some **conformation changes**

MaSIF: Geometric ML for Protein Function Prediction & Design



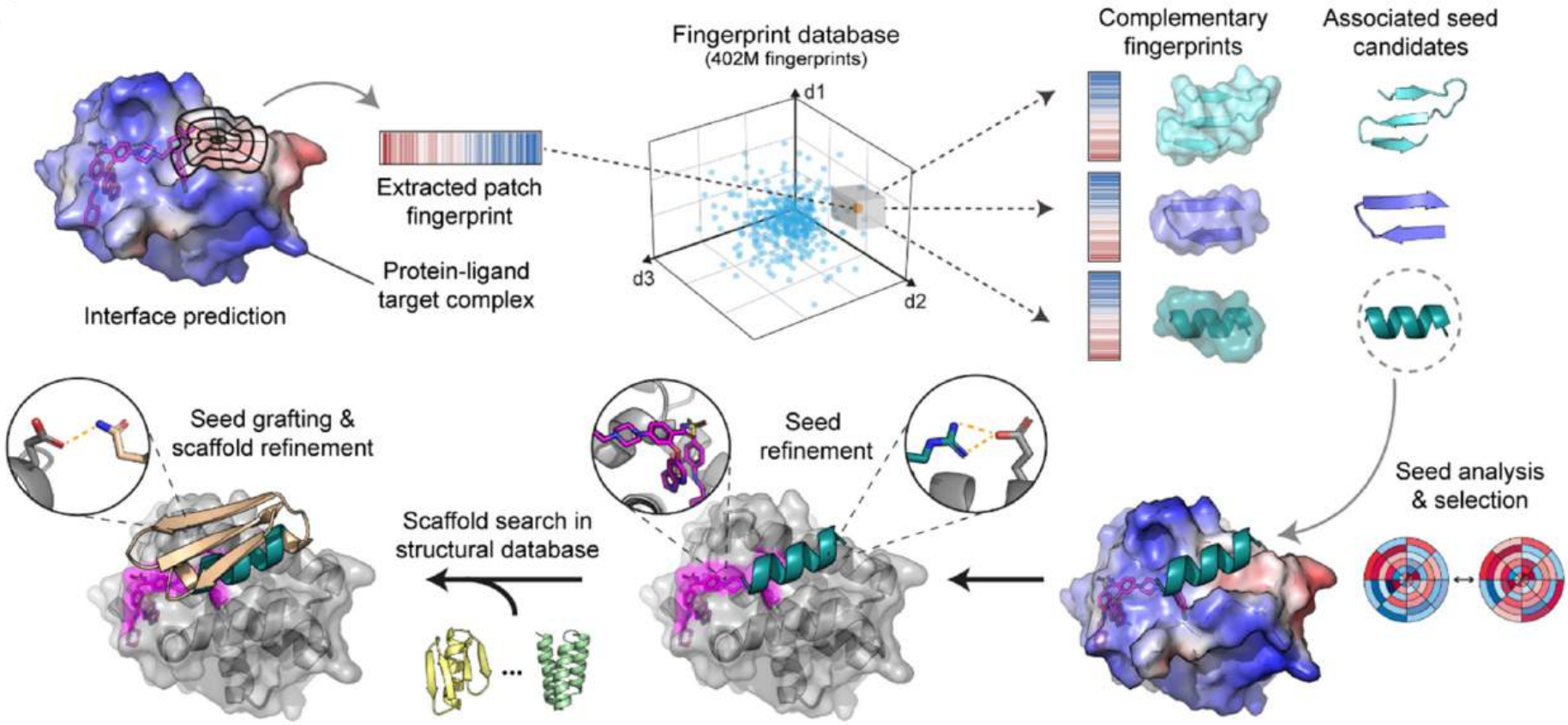


Gainza et B, Correia 2020



Gainza et B, Correia 2022

Switchable interactions



Marchand et B, Correia 2024

Article

De novo design of protein interactions w learned surface fingerprints

https://doi.org/10.1038/s41586-023-05993-x

Received: 16 June 2022

Accepted: 21 March 2023

Published online: 26 April 2023

Open access

Pablo Gainza[1,2,3,4], Sarah Wehrle[1,2,3], Alexandra Van Hall-Beauvais[1,2,3], Anth Andreas Scheck[1,2,5], Zander Harteveld[1,2], Stephen Buckley[1,2], Dongchun Ni[1] Freyr Sverrisson[1,2], Casper Goverde[1,2], Priscilla Turelli[6], Charlène Raclot[6], Alexandra Teslenko[7], Martin Pacesa[1,2], Stéphane Rosset[1,2], Sandrine Georg Jane Marsden[1,2], Aaron Petruzzella[8], Kefang Liu[1], Zepeng Xu[1], Yan Chai[1], Pu George F. Gao[9], Elisa Oricchio[8], Beat Fierz[7], Didier Trono[6], Henning Stahlbe Michael Bronstein[2,3] & Bruno E. Correia[1,2]

Physical interactions between proteins are essential for most biologi governing life[1]. However, the molecular determinants of such interac challenging to understand, even as genomic, proteomic and structur This knowledge gap has been a major obstacle for the comprehensiv of cellular protein–protein interaction networks and for the de novo binders that are crucial for synthetic biology and translational applic we use a geometric deep-learning framework operating on protein su generates fingerprints to describe geometric and chemical features t drive protein–protein interactions[10]. We hypothesized that these fing the key aspects of molecular recognition that represent a new paradi computational design of novel protein interactions. As a proof of prin computationally designed several de novo protein binders to engage targets: SARS-CoV-2 spike, PD-1, PD-L1 and CTLA-4. Several designs w experimentally optimized, whereas others were generated purely in s nanomolar affinity with structural and mutational characterization s accurate predictions. Overall, our surface-centric approach captures chemical determinants of molecular recognition, enabling an approa de novo design of protein interactions and, more broadly, of artificia function.

Designing novel protein–protein interactions (PPIs) remains a fundamental challenge in computational protein design, with broad basic and translational applications in biology. The challenge consists of generating amino acid sequences that engage a target site and form a quaternary complex with a given protein. This represents a stringent test of our understanding of the physicochemical determinants that drive biomolecular interactions[1]. Robust computational methods to design de novo PPIs could be used to rapidly engineer protein-based therapeutics such as antibodies and protein inhibitors or vaccines among others, and are therefore of considerable interest for biomedical and translational applications[2–4].

Despite recent advances in rational PPI design[2,4,5] and prediction[11], designing novel protein binders against specific targets is very challenging, particularly when no structural elements from pre-existing binders are known. Current state-of-the-art metho design[1,6,12,13], such as hotspot-centric approaches[4] mation fields[2,5], rely on placing disembodied resi interface and then optimizing their presentation fold. Intrinsic limitations of these approaches rela energetic signatures provided by scoring functions placements, which is compounded in flat interfa pockets. These methods also face the challenge of protein scaffolds to precisely display the generate residues. To circumvent these limitations, new app to design de novo binders to various surface types

A long-standing model of molecular recogniti PPIs form between protein molecular surfaces with metric complementarity[11,14]. The complementarity

[1]Laboratory of Protein Design and Immunoengineering, Institute of Bioengineering, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland. [2]Swiss Institute Lausanne, Switzerland. [3]Laboratory of Biological Electron Microscopy, Institute of Physics, School of Basic Science, Ecole Polytechnique Fédérale de Lausanne, Lausa [4]Department of Fundamental Microbiology, Faculty of Biology and Medicine, University of Lausanne, Lausanne, Switzerland. [5]CAS Key Laboratory of Pathogen Microbi Institute of Microbiology, Chinese Academy of Sciences, Beijing, China. [6]Laboratory of Virology and Genetics, School of Life Sciences, Ecole Polytechnique Fédérale d Switzerland. [7]Laboratory of Biophysical Chemistry of Macromolecules, School of Basic Sciences, Institute of Chemical Sciences and Engineering (ISIC), Ecole Polytech Lausanne, Lausanne, Switzerland. [8]Swiss Institute for Experimental Cancer Research, School of Life Sciences, Ecole Polytechnique Fédérale de Lausanne, Lausanne, S of Computer Science, University of Oxford, Oxford, UK. [10]Present address: Monte Rosa Therapeutics, Basel, Switzerland. [11]These authors contributed equally: Pablo Ga Alexandra Van Hall-Beauvais, Anthony Marchand, Andreas Scheck. ✉e-mail: michael.bronstein@cs.ox.ac.uk; bruno.correia@epfl.ch

g protein–ligand neosurfaces with a able deep learning tool

-024-08435-4

025

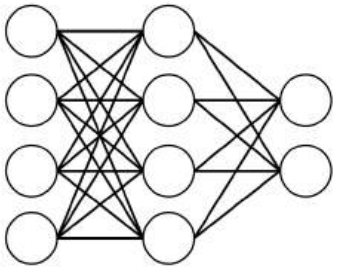
Anthony Marchand[1,8], Stephen Buckley[1,8], Arne Schneuing[1,8], Martin Pacesa[1], Maddalena Elia[1], Pablo Gainza[1,2], Evgenia Elizarova[1], Rebecca M. Neeser[1,3], Pao-Wan Lee[1], Luc Reymond[4], Yangyang Miao[1], Leo Schaller[1], Sandrine Georgeon[1], Joseph Schmidt[5], Philippe Schwaller[3], Sebastian J. Maerkl[1], Michael Bronstein[2,6] & Bruno E. Correia[1,2]

Molecular recognition events between proteins drive biological processes in living systems[1]. However, higher levels of mechanistic regulation have emerged, in which protein–protein interactions are conditioned to small molecules[2–5]. Despite recent advances, computational tools for the design of new chemically induced protein interactions have remained a challenging task for the field[6,7]. Here we present a computational strategy for the design of proteins that target neosurfaces, that is, surfaces arising from protein–ligand complexes. To develop this strategy, we leveraged a geometric deep learning approach based on learned molecular surface representations[8,9] and experimentally validated binders against three drug-bound protein complexes: Bcl2–venetoclax, DB3–progesterone and PDF1–actinonin. All binders demonstrated high affinities and accurate specificities, as assessed by mutational and structural characterization. Remarkably, surface fingerprints previously trained only on proteins could be applied to neosurfaces induced by interactions with small molecules, providing a powerful demonstration of generalizability that is uncommon in other deep learning approaches. We anticipate that such designed chemically induced protein interactions will have the potential to expand the sensing repertoire and the assembly of new synthetic pathways in engineered cells for innovative drug-controlled cell-based therapies[10].

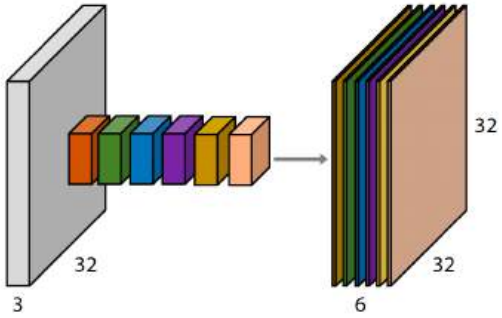
PPIs) have essential roles in healthy cell olved in numerous diseases[1,11]. For this ting PPIs have been developed over the putational tools for the design of new ently been proposed[12]. The governing pensity of proteins to form interactions rplay of several contributions, such as ementarity, dynamics and solvent inter- challenging to predict and design new of evolutionary constraints. Native PPIs tory layers such as allostery[3], post trans- ct ligand binding[4,5]. Compound-bound neosurfaces, are among the most fasci- ular recognition instances, as relatively binding site can have a large impact on uch interactions has been fuelled by the alities, specifically, molecular glues that otein interactions for degradation and s represent a promising route for the rapeutics.

In synthetic biology, molecular components that rely on small-molecule-induced neosurfaces have been used to engineer chemically responsive systems with precise spatiotemporal control of cellular activities[13]. Small-molecule triggers have been used to both induce and disrupt PPIs, thereby functioning as ON or OFF switches for engineered cellular functions[10,11,14]. There are several practical advantages to using small molecules as triggers, including their simple administration, biodistribution, cell permeability, safety, and high affinity and specificity to their target proteins. Protein-based switches controlled by small molecules have already been used to regulate transcription[15], protein degradation[16,17] and protein localization[13,18], among many other applications. In addition to their use in basic research, engineering molecular switches are increasingly used to control protein-based and cellular therapeutics, the activity of which may need to be regulated to mitigate potentially dangerous side effects[10,12,13]. Although several chemically disruptable heterodimer (OFF-switch) systems have been proposed[16,19,20], computationally designed chemically induced dimerization (CID, ON-switch) systems remain challenging owing to the complexity of modelling neosurfaces. Previous attempts to design CID systems primarily relied on experimental methods[10,12,21–26], and,

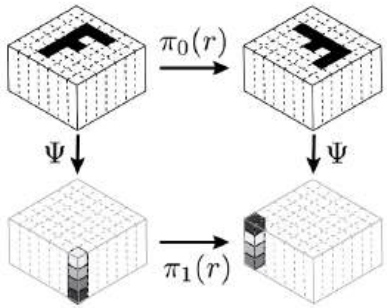
oengineering, Institute of Bioengineering, Ecole polytechnique fédérale de Lausanne, Lausanne, Switzerland. [3]Laboratory of Chemical Artificial as and Engineering, Ecole polytechnique fédérale de Lausanne, Lausanne, Switzerland. [4]Laboratory of Biological Network Characterization, Institute fédérale de Lausanne, Lausanne, Switzerland. [5]Biomolecular Screening Core Facility, School of Life Sciences, Ecole polytechnique fédérale de ment of Computer Science, University of Oxford, Oxford, UK. [7]Aithera Research Institute for Biomedical Artificial Intelligence, Austrian Academy ense. Monte Rose Therapeutics, Boston, MA, USA. [9]These authors contributed equally: Anthony Marchand, Stephen Buckley, Arne Schneuing.



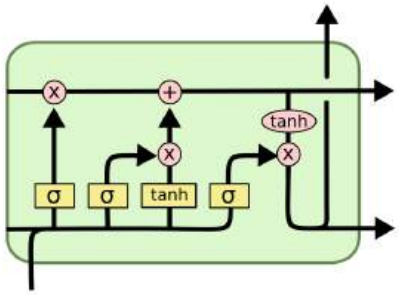
Perceptrons
Function regularity



CNNs
Translation

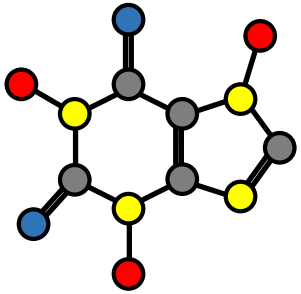


Group-CNNs
Translation+Rotation,
Global groups



LSTMs
Time warping

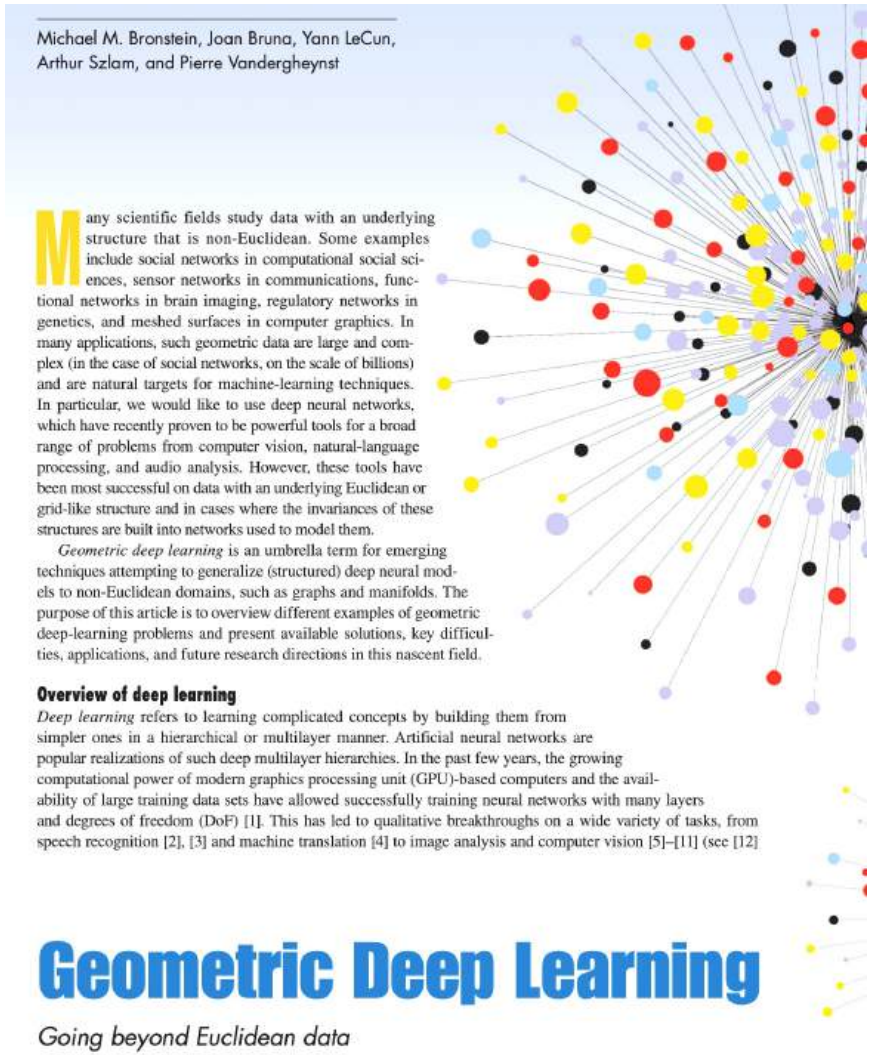
DeepSets / Transformers
Permutation



GNNs
Permutation

Intrinsic CNNs
Isometry / Gauge choice


ANY NN
ARCHITECTURE
GEOMETRIC
DEEP LEARNING
TikTok
@tired_actor



Michael M. Bronstein, Joan Bruna, Yann LeCun,
Arthur Szlam, and Pierre Vandergheynst

Many scientific fields study data with an underlying structure that is non-Euclidean. Some examples include social networks in computational social sciences, sensor networks in communications, functional networks in brain imaging, regulatory networks in genetics, and meshed surfaces in computer graphics. In many applications, such geometric data are large and complex (in the case of social networks, on the scale of billions) and are natural targets for machine-learning techniques. In particular, we would like to use deep neural networks, which have recently proven to be powerful tools for a broad range of problems from computer vision, natural-language processing, and audio analysis. However, these tools have been most successful on data with an underlying Euclidean or grid-like structure and in cases where the invariances of these structures are built into networks used to model them.

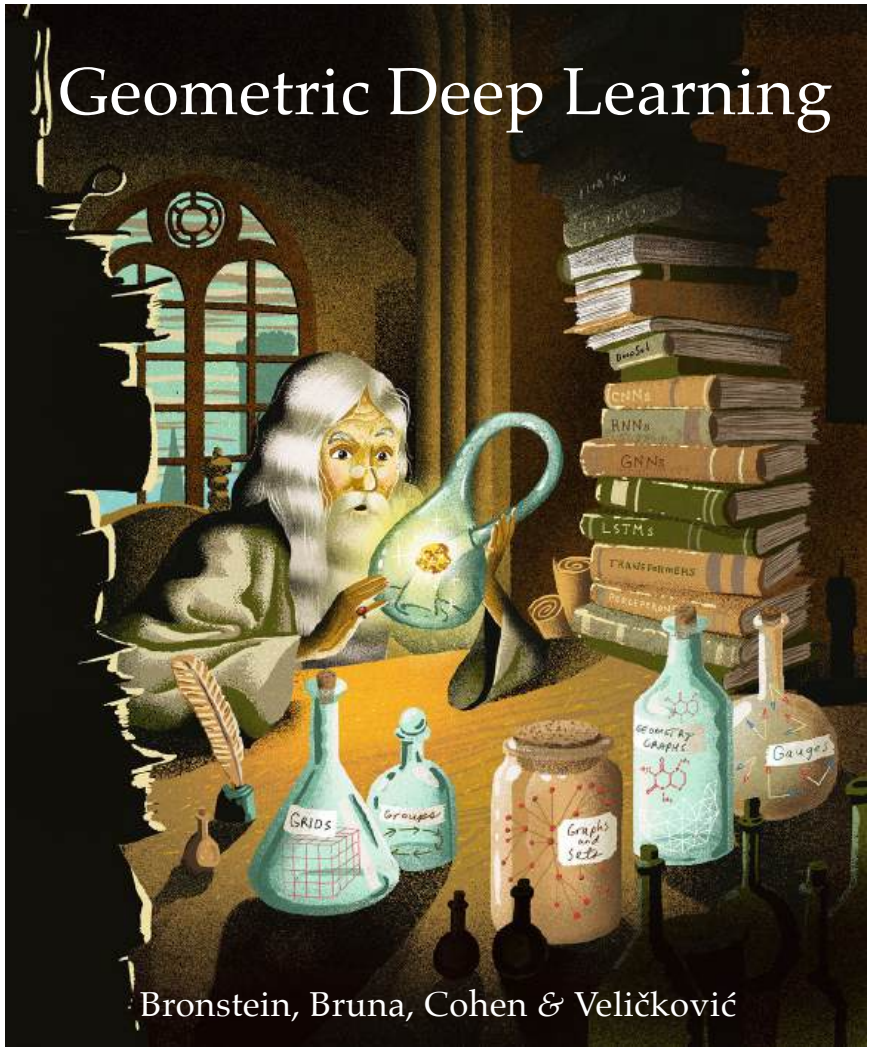
Geometric deep learning is an umbrella term for emerging techniques attempting to generalize (structured) deep neural models to non-Euclidean domains, such as graphs and manifolds. The purpose of this article is to overview different examples of geometric deep-learning problems and present available solutions, key difficulties, applications, and future research directions in this nascent field.

Overview of deep learning

Deep learning refers to learning complicated concepts by building them from simpler ones in a hierarchical or multilayer manner. Artificial neural networks are popular realizations of such deep multilayer hierarchies. In the past few years, the growing computational power of modern graphics processing unit (GPU)-based computers and the availability of large training data sets have allowed successfully training neural networks with many layers and degrees of freedom (DoF) [1]. This has led to qualitative breakthroughs on a wide variety of tasks, from speech recognition [2], [3] and machine translation [4] to image analysis and computer vision [5]–[11] (see [12]

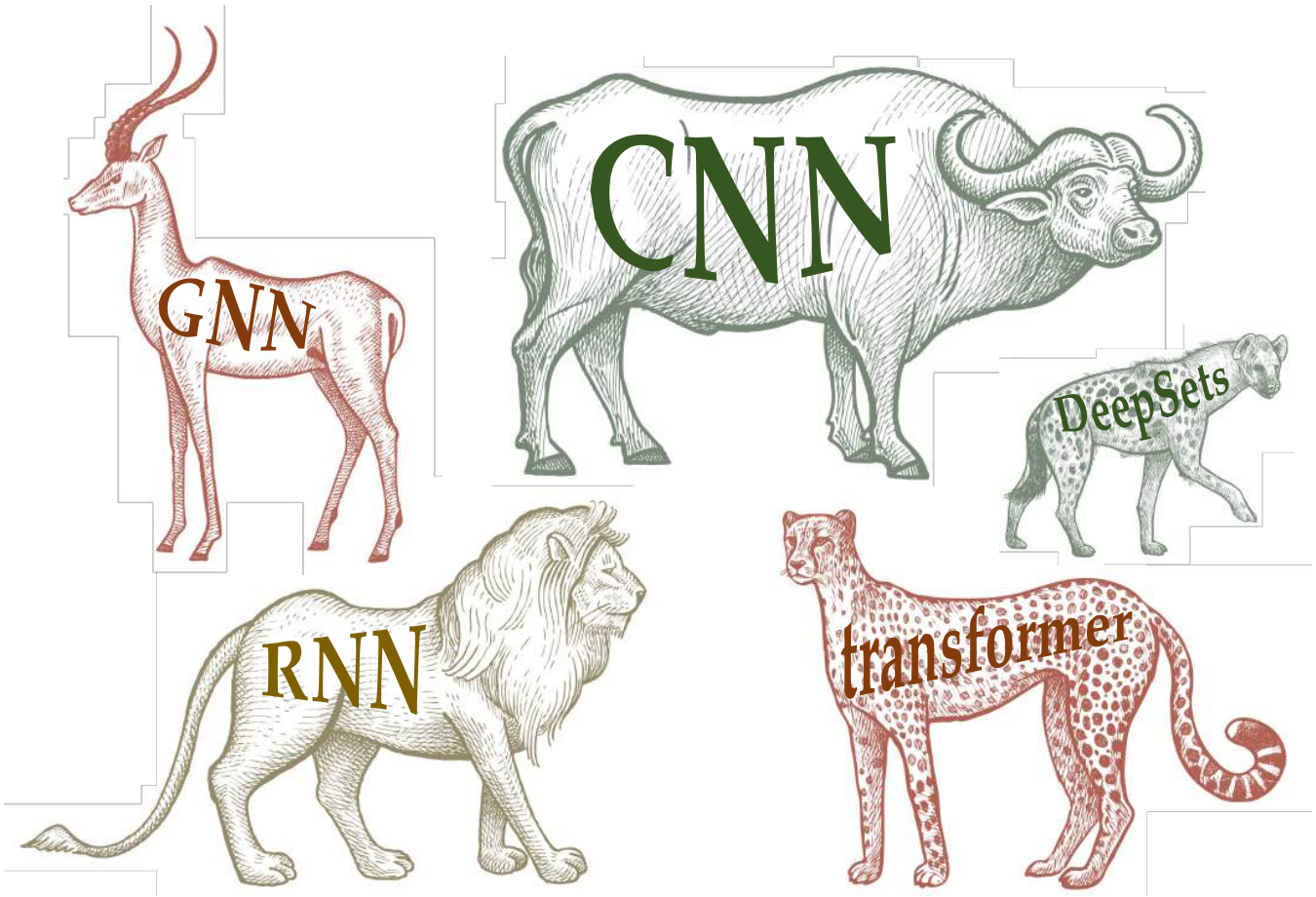
Geometric Deep Learning

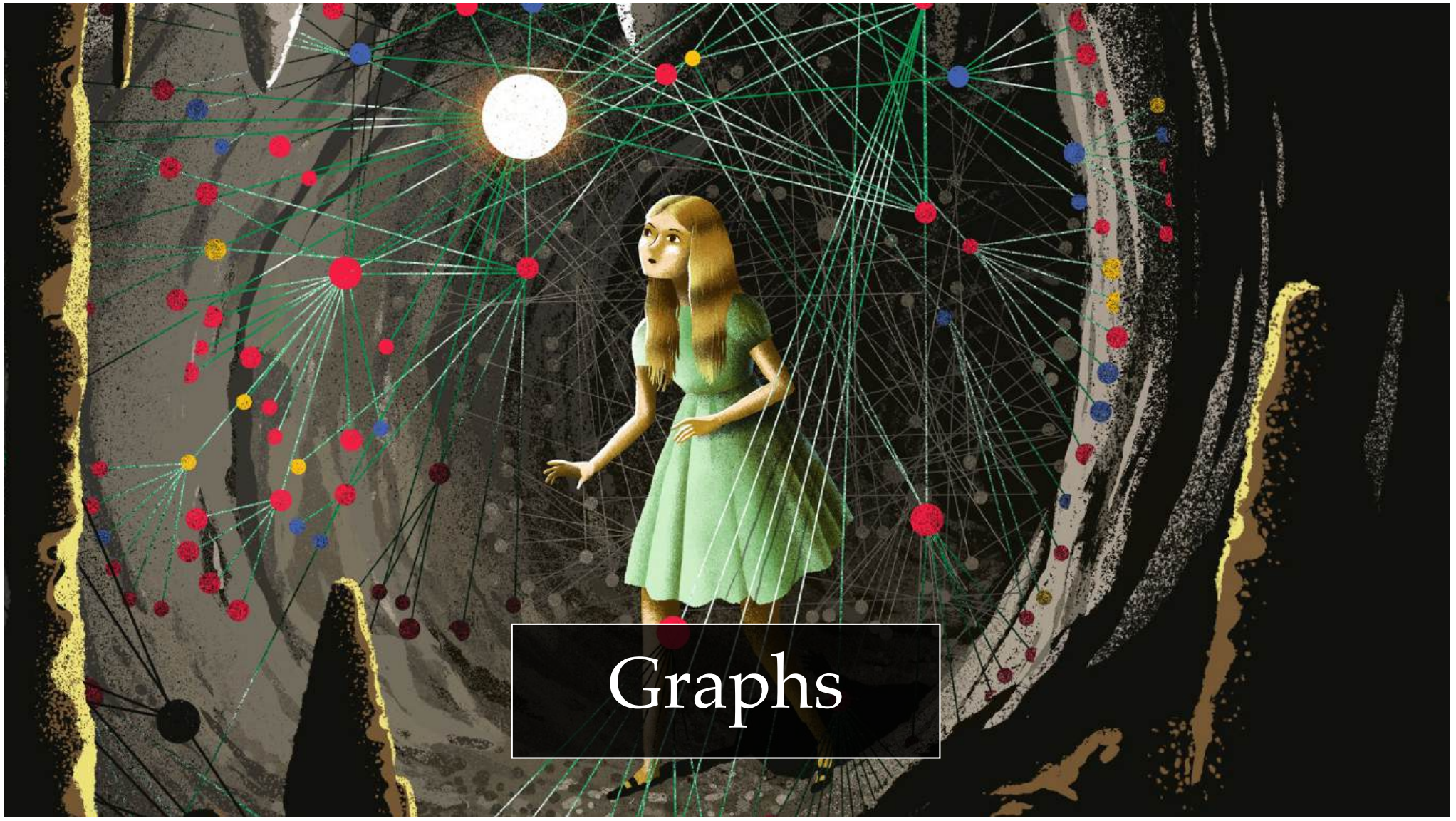
Going beyond Euclidean data



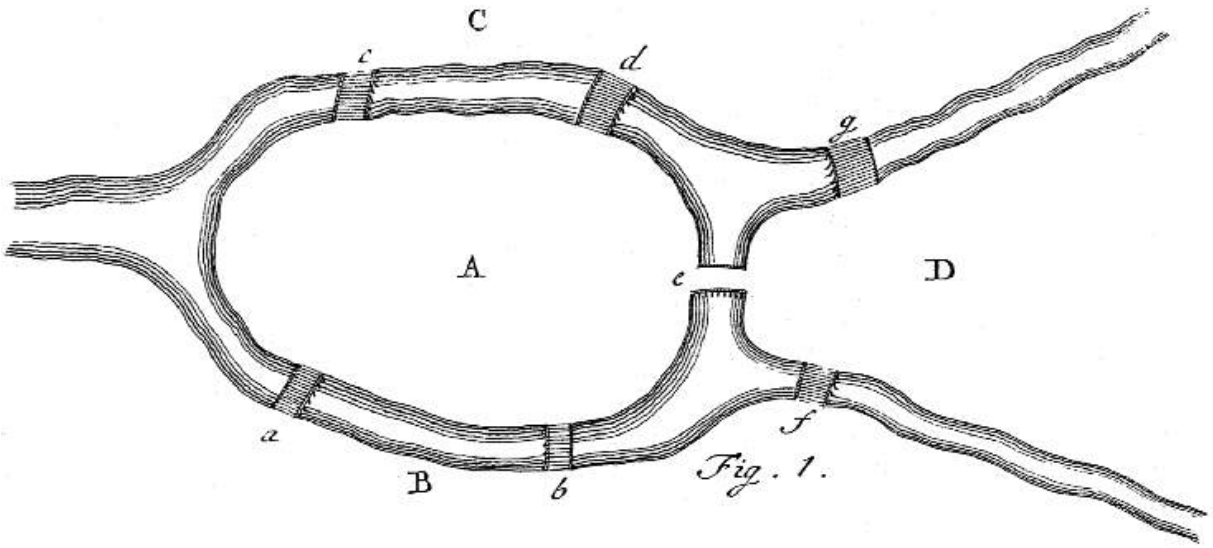
"The Erlangen Programme of ML"

Geometric Deep Learning


GNN
CNN
DeepSets
RNN
transformer


Graphs

Origins of Graph Theory



L. Euler

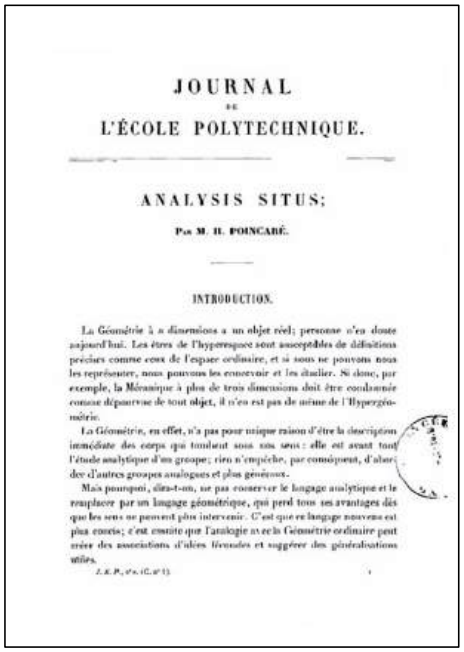
1741

The soluton of the classical problem of the "Bridges of Königsberg" by Euler in 1736 first showed the power of graphs to abstract out the geometry (*"geometria situs"*)

Euler 1741

Origins of Topology

Poincaré's *"analysis situs."* His famous Conjecture appeared in a supplement published in 1904.



JOURNAL
DE
L'ÉCOLE POLYTECHNIQUE.

ANALYSIS SITUS;

Par M. H. POINCARÉ.

INTRODUCTION.

La Géométrie à n dimensions a un objet réel; personne n'en doute aujourd'hui. Les êtres de l'hyperespace sont susceptibles de définitions précises comme ceux de l'espace ordinaire, et si nous ne pouvons nous les représenter, nous pouvons les concevoir et les étudier. Si donc, par exemple, la Mécanique à plus de trois dimensions doit être condamnée comme dépourvue de tout objet, il n'en est pas de même de l'Hypergéométrie.

La Géométrie, en effet, n'a pas pour unique raison d'être la description immédiate des corps qui tombent sous nos sens : elle est avant tout l'étude analytique d'un groupe; rien n'empêche, par conséquent, d'abord de d'autres groupes analogues et plus généraux.

Mais pourquoi, dira-t-on, ne pas conserver le langage analytique et le remplacer par un langage géométrique, qui perd tous ses avantages dès que les sens ne peuvent plus intervenir. C'est que ce langage nouveau est plus concis; c'est ensuite que l'analogie avec la Géométrie ordinaire peut créer des associations d'idées fécondes et suggérer des généralisations utiles.

H. Poincaré

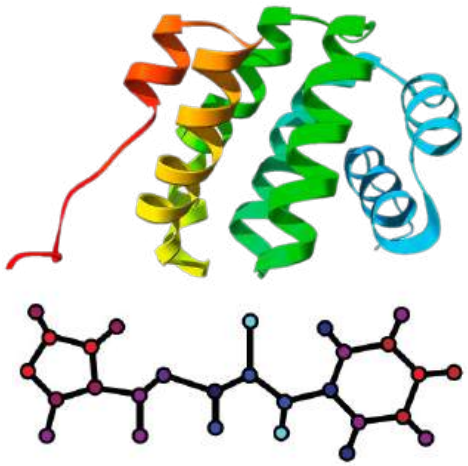
1895

L. Euler

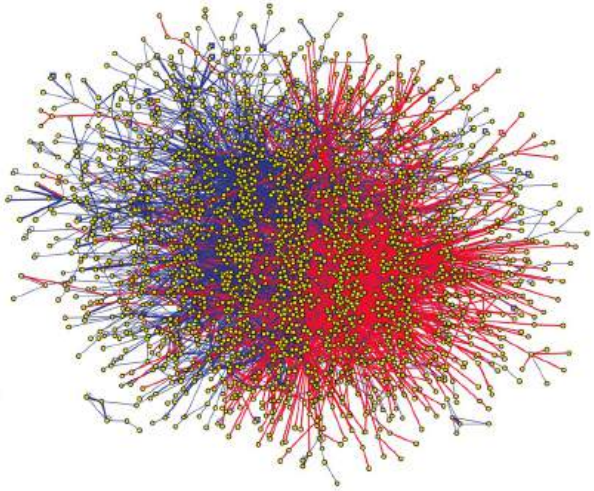
1741

Euler 1741; Poincaré 1895

Graphs = Systems of Relations and Interactions



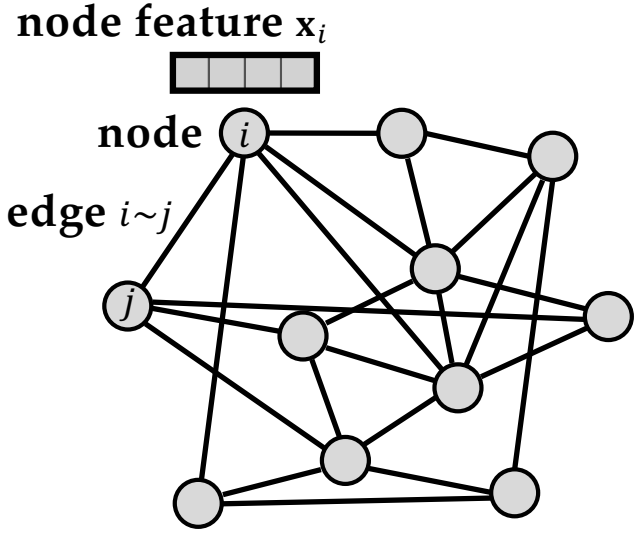
Molecules



Interactomes

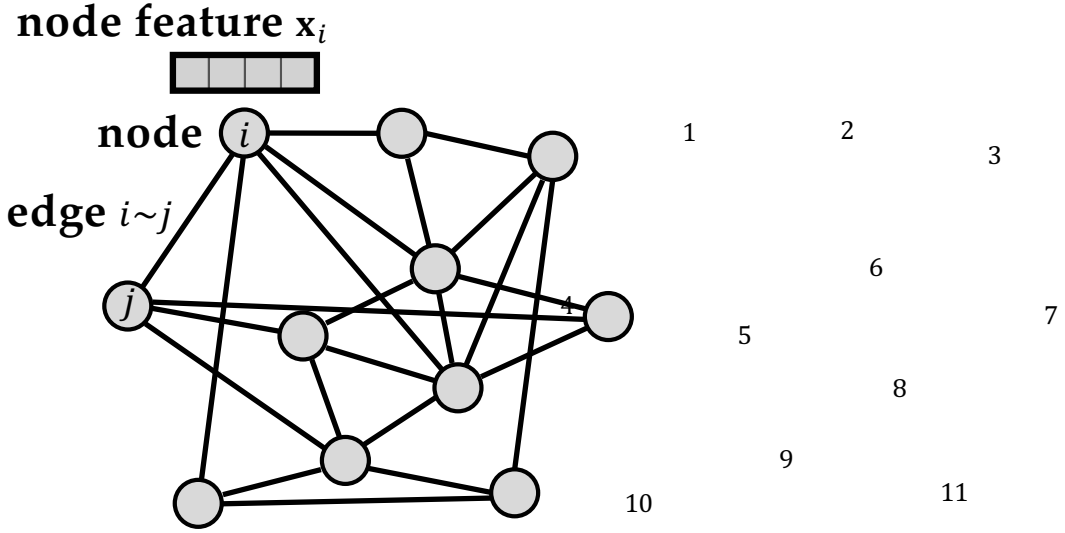
Social networks

Graphs: The Basics



graph

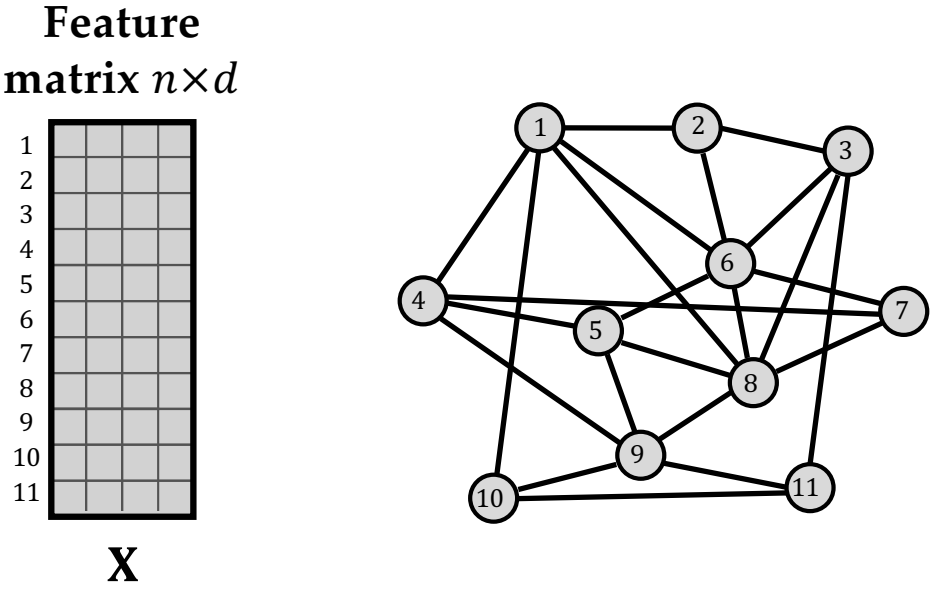
Key Structural Properties of Graphs



graph

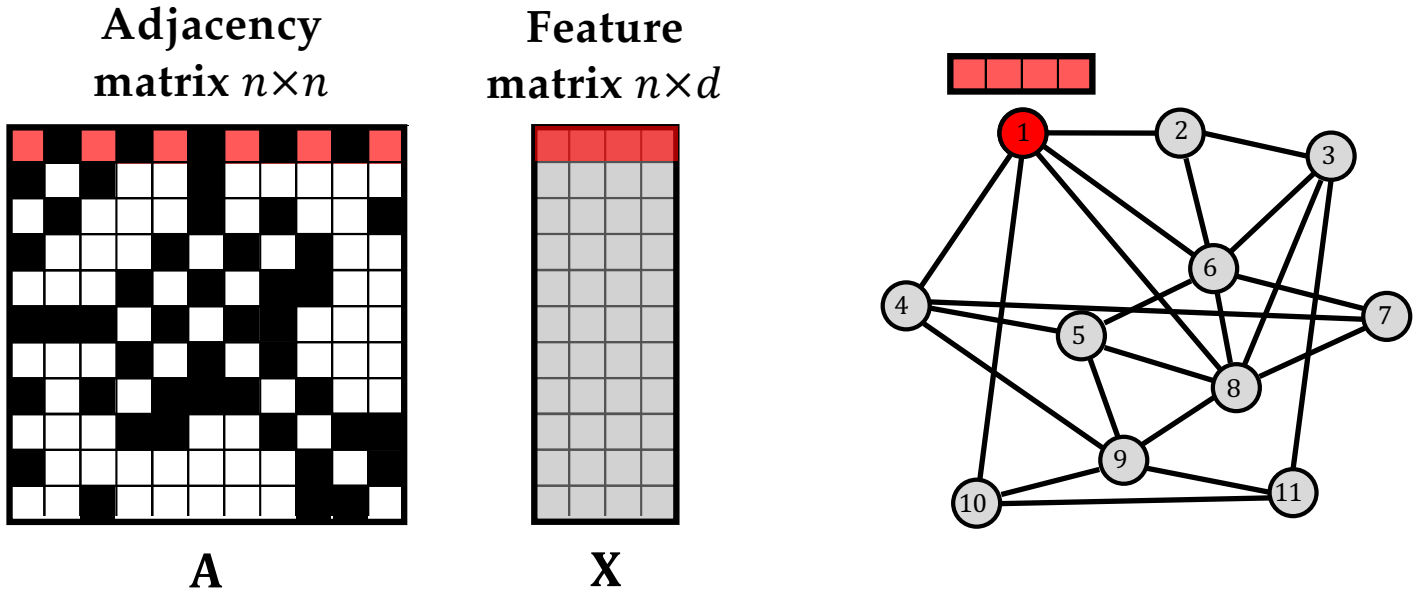
arbitrary ordering of nodes

Key Structural Properties of Graphs



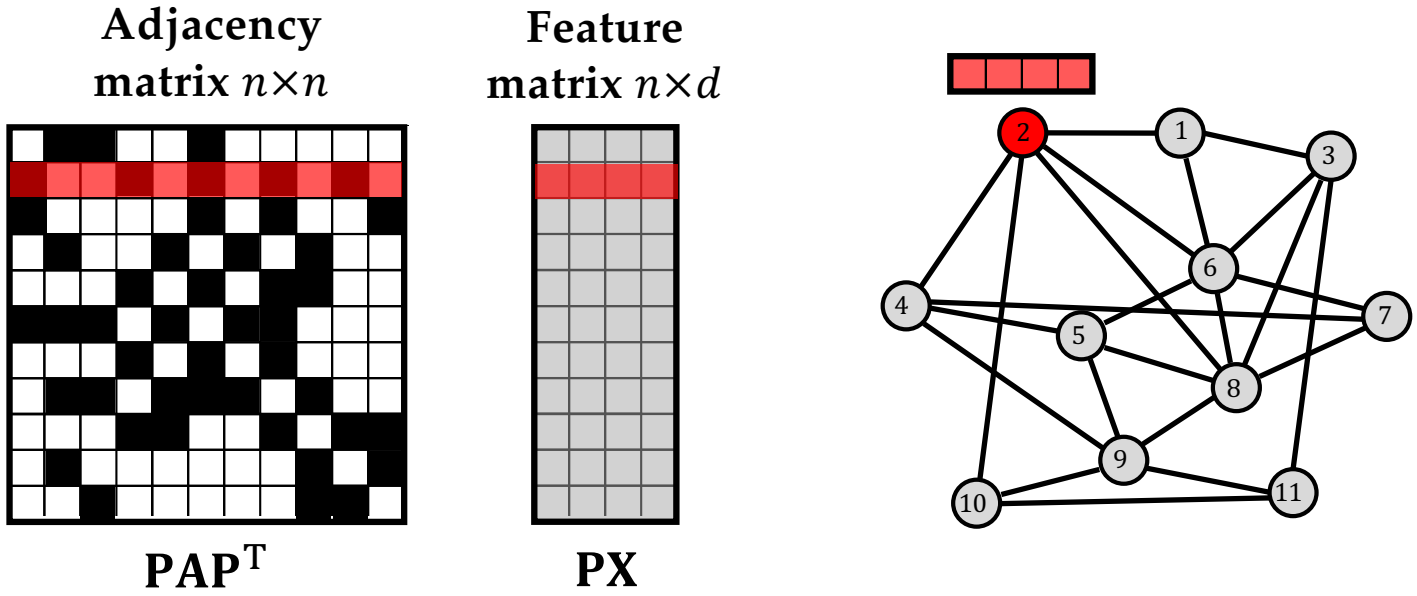
arbitrary ordering of nodes

Key Structural Properties of Graphs



arbitrary ordering of nodes

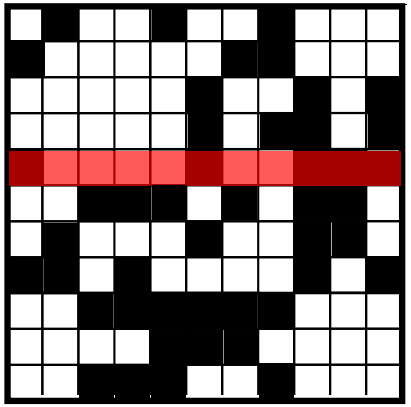
Key Structural Properties of Graphs



arbitrary ordering of nodes

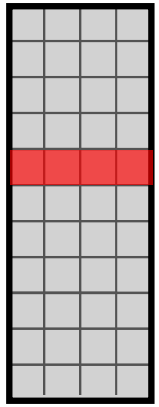
Key Structural Properties of Graphs

Adjacency matrix $n \times n$

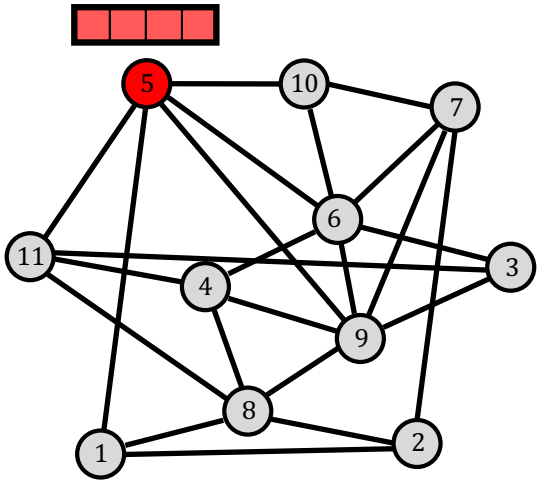


$\mathbf{PAP^T}$

Feature matrix $n \times d$



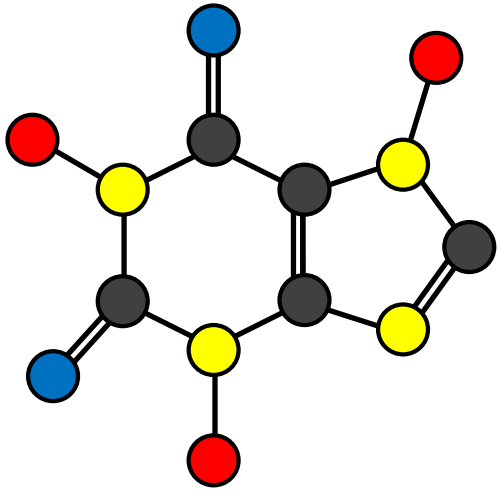
$\mathbf{PX}$



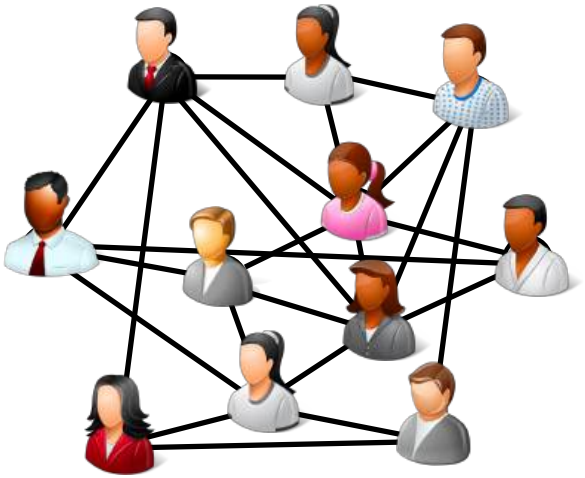
arbitrary ordering of nodes

$n!$ permutations

Invariant vs Equivariant tasks



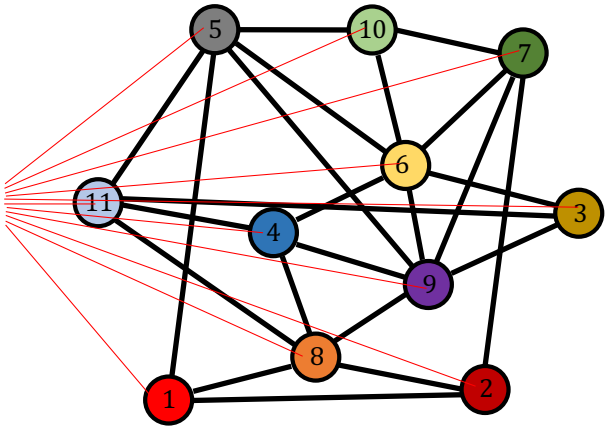
water solubility?



who is a spammer?

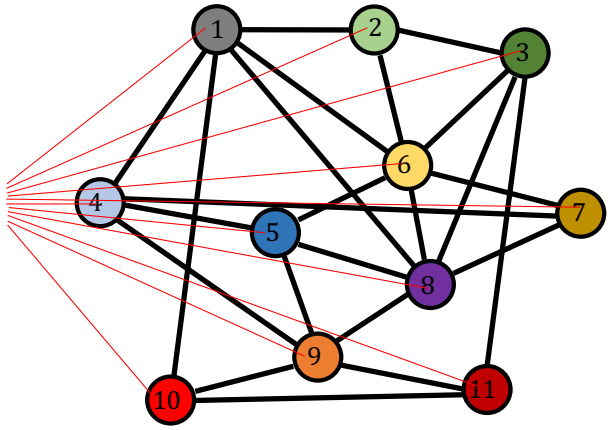
Invariant Graph Functions

graph function $f(\mathbf{X}, \mathbf{A})$



Invariant Graph Functions

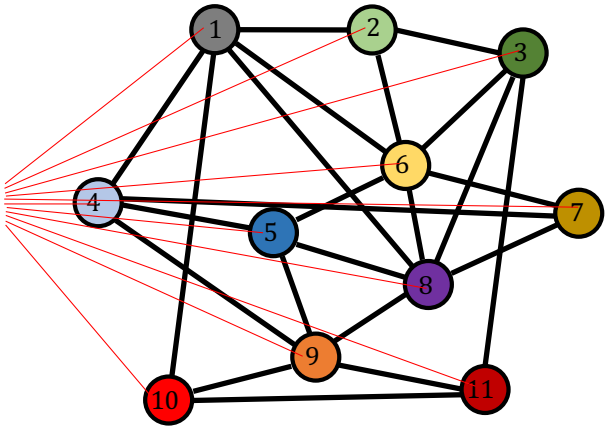
graph function $f(\mathbf{X}, \mathbf{A})$



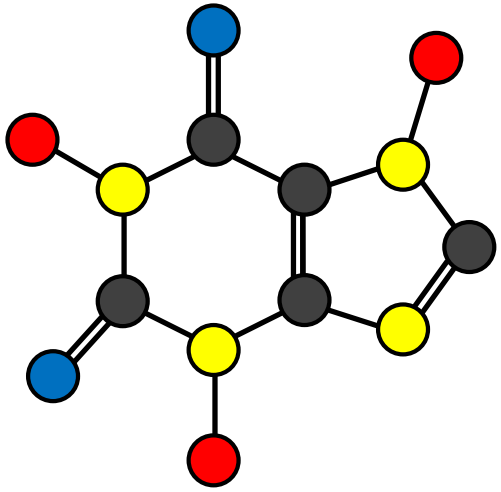
Invariant Graph Functions

permutation-invariant

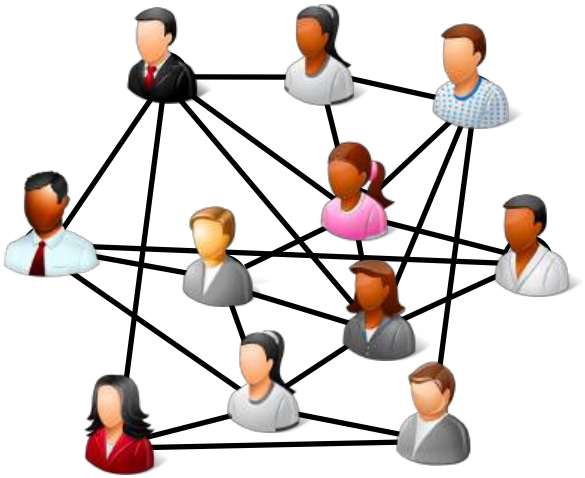
$$f\left(\mathbf{PX}, \mathbf{PAP}^{\top}\right) = f(\mathbf{X}, \mathbf{A})$$



Invariant vs Equivariant tasks



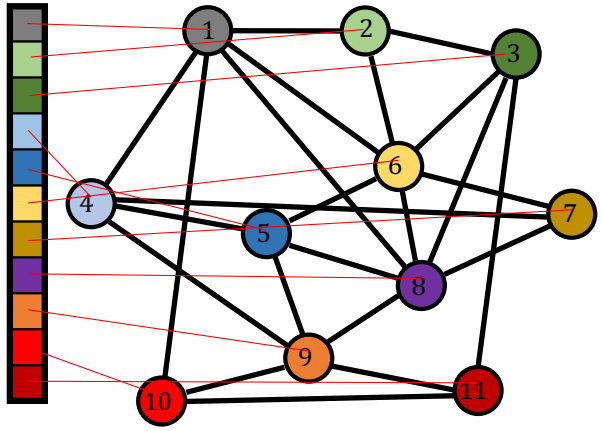
water solubility?



who is a spammer?

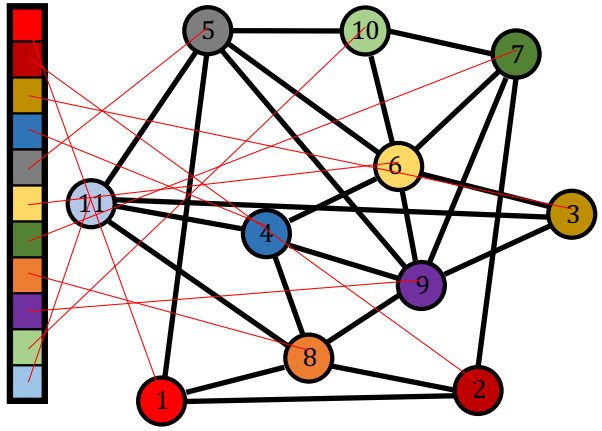
Equivariant Graph Functions

node function $\mathbf{F}(\mathbf{X}, \mathbf{A})$



Equivariant Graph Functions

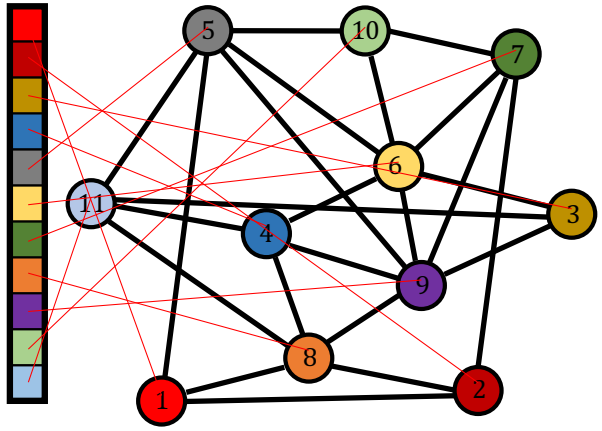
node function $\mathbf{F}(\mathbf{X}, \mathbf{A})$



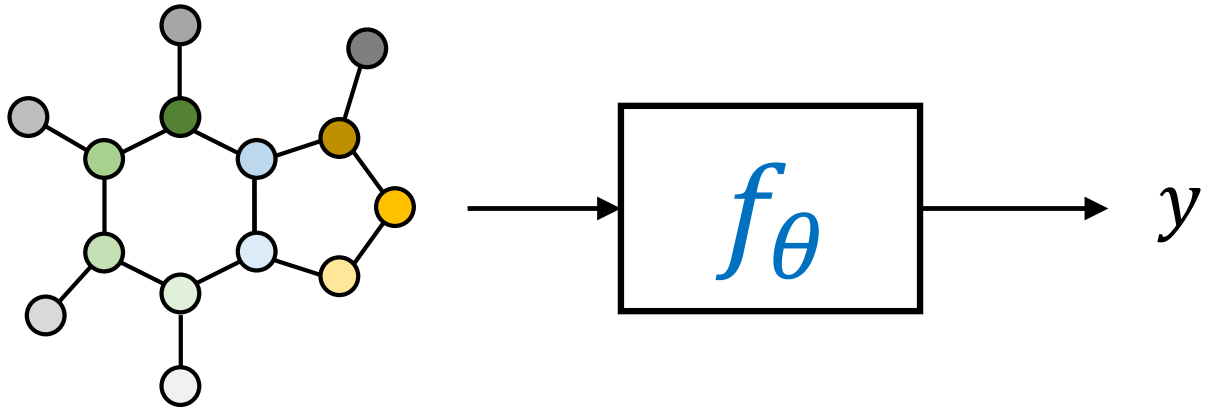
Equivariant Graph Functions

permutation-equivariant

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



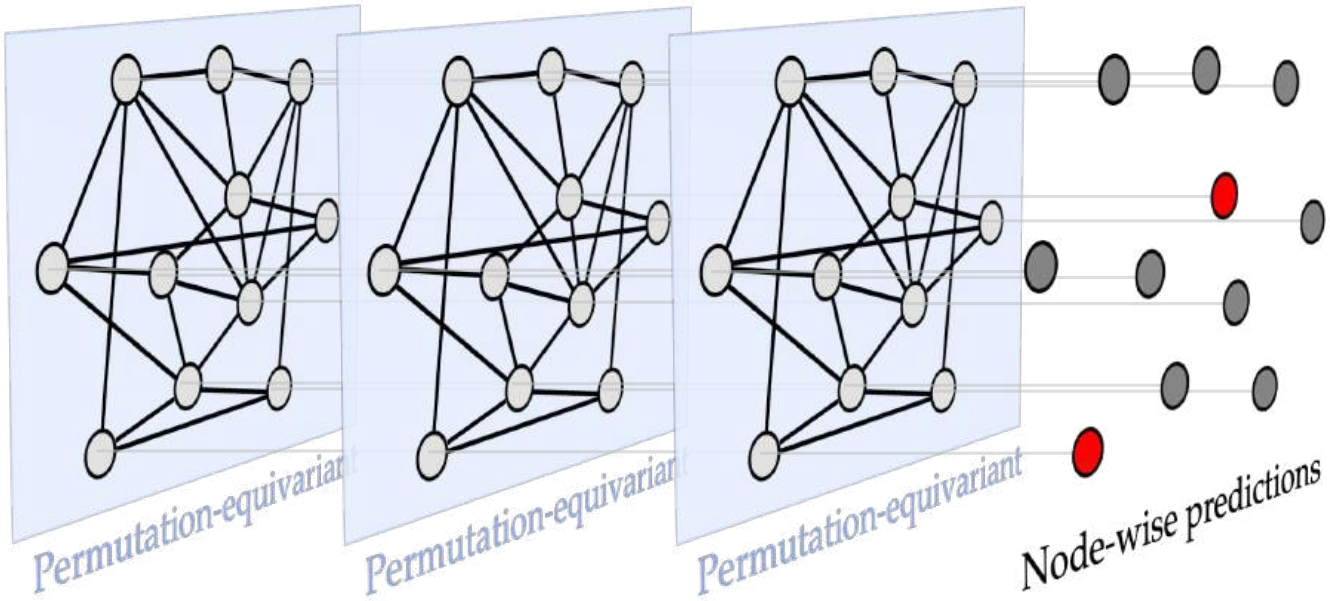
GNNs = Parametric graph functions



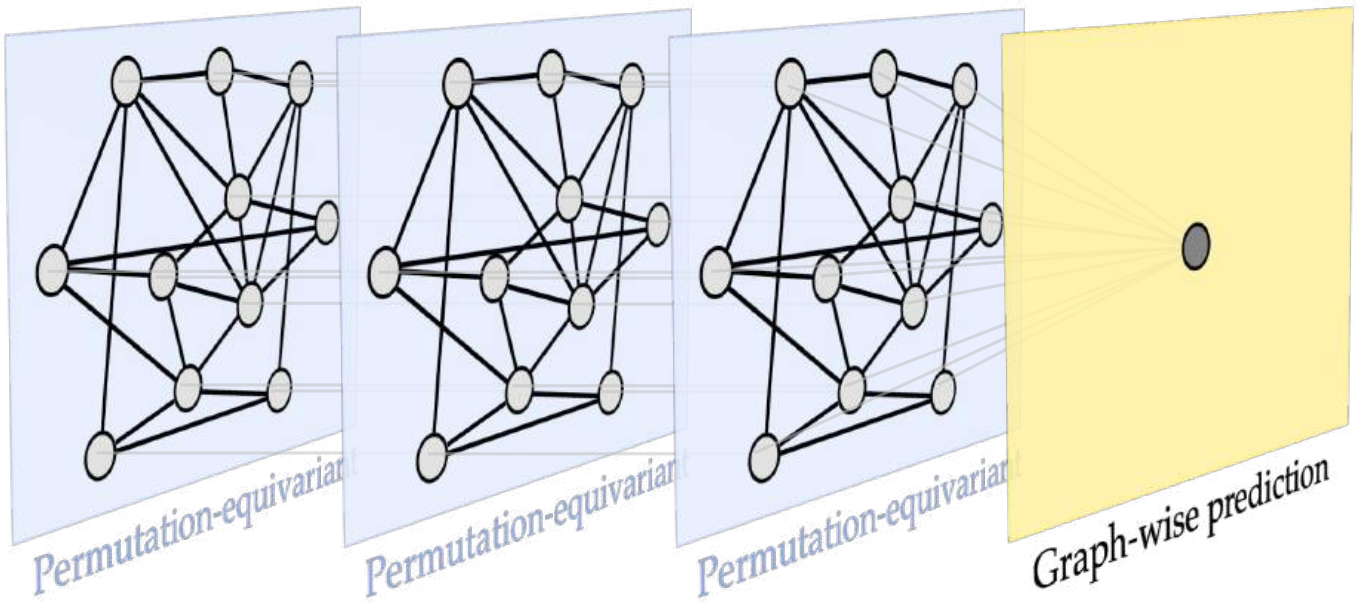
Graph+Features

Note: we use the term "GNN" to refer to general parametric graph functions. The particular class of GNNs we consider are Message Passing Neural Networks (MPNNs). Petar Veličković argues that "it's all message passing"

Graph Neural Networks: Node tasks



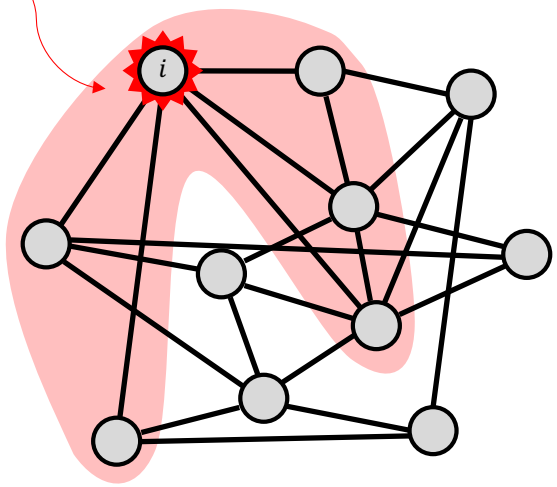
Graph Neural Networks: Graph tasks



Neighbour Aggregation

neighbourhood

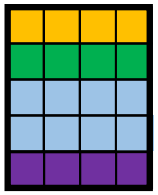
$\mathcal{N}_i = \{j : i \sim j\}$



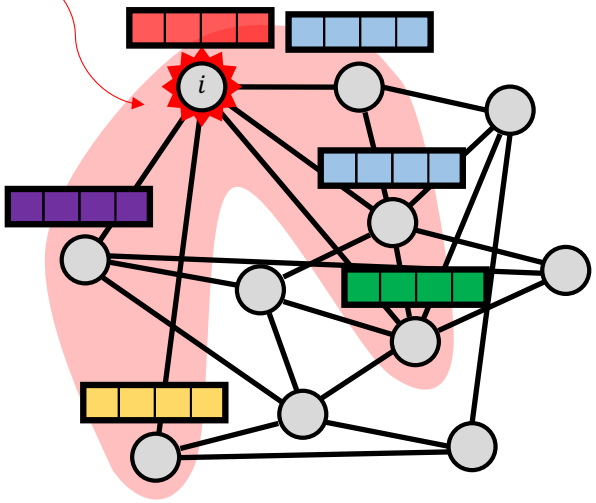
Neighbour Aggregation

neighbourhood
$\mathcal{N}_i = \{j : i \sim j\}$

multiset of
neighbour features



$$\mathbf{X}_{\mathcal{N}_i} = \{\!\{\mathbf{x}_{j \in \mathcal{N}_i}\}\!\}$$



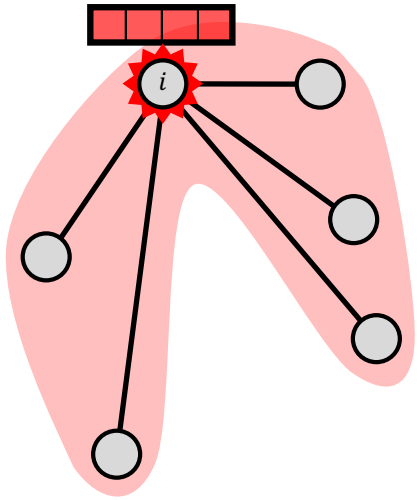
Neighbour Aggregation

multiset of neighbour features

local function

$$\phi\left(\mathbf{x}_i, \mathbf{x}_{\mathcal{N}_i}\right)$$

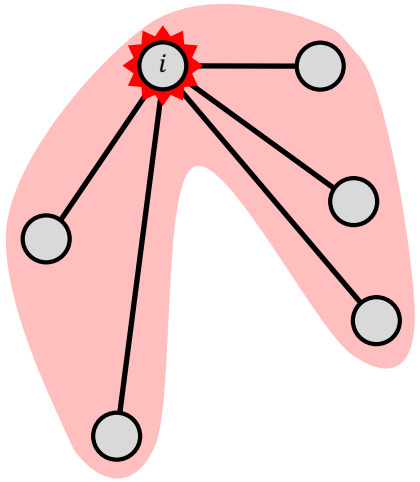
$$\mathbf{x}_{\mathcal{N}_i} = \{\!\{\mathbf{x}_{j \in \mathcal{N}_i}\}\!\}$$



Neighbour Aggregation

local function

$$\phi\left(\mathbf{x}_i, \mathbf{x}_{\mathcal{N}_i}\right)$$

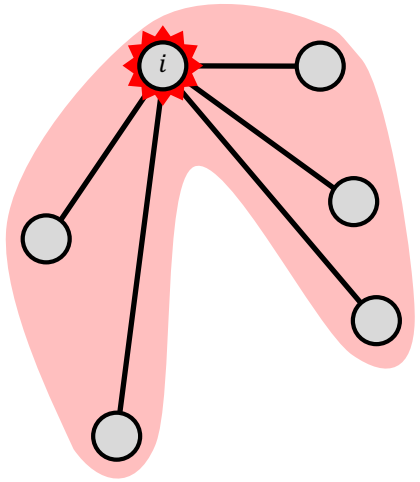


Neighbour Aggregation

local function

$$\phi\left(\begin{bmatrix}\mathbf{x}_i \\ \mathbf{x}_{\mathcal{N}_i}\end{bmatrix}\right)$$

permutation invariant

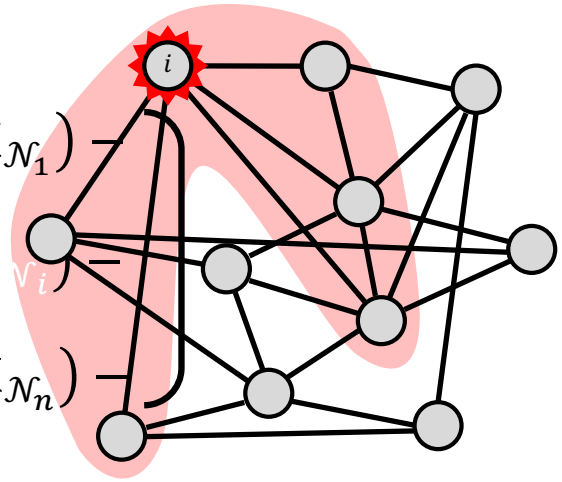


Neighbour Aggregation

local function

$$\mathbf{x}_i \quad \mathbf{x}_{\mathcal{N}_i} \quad \begin{bmatrix} -\phi(\mathbf{x}_1, \mathbf{X}_{\mathcal{N}_1}) - \\ \vdots \\ -\phi(\quad , \quad) - \\ \vdots \\ -\phi(\mathbf{x}_n, \mathbf{X}_{\mathcal{N}_n}) - \end{bmatrix}$$

permutation invariant

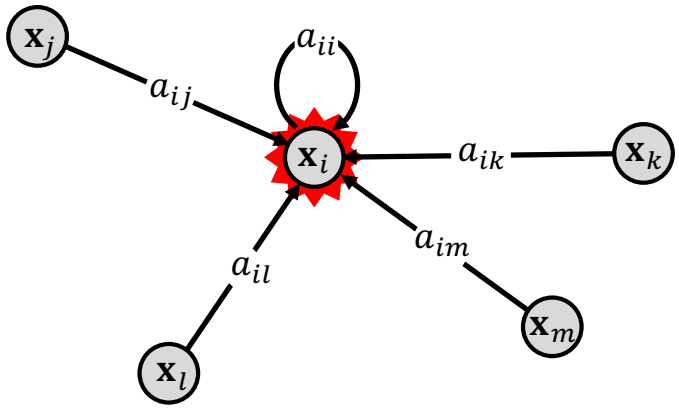


GNN Layer

$$\mathbf{F}(\mathbf{X}, \mathbf{A}) = \begin{pmatrix} - \phi(\mathbf{x}_1, \mathbf{X}_{\mathcal{N}_1}) - \\ \vdots \\ - \phi(\mathbf{x}_i, \mathbf{X}_{\mathcal{N}_i}) - \\ \vdots \\ - \phi(\mathbf{x}_n, \mathbf{X}_{\mathcal{N}_n}) - \end{pmatrix}$$

permutation equivariant

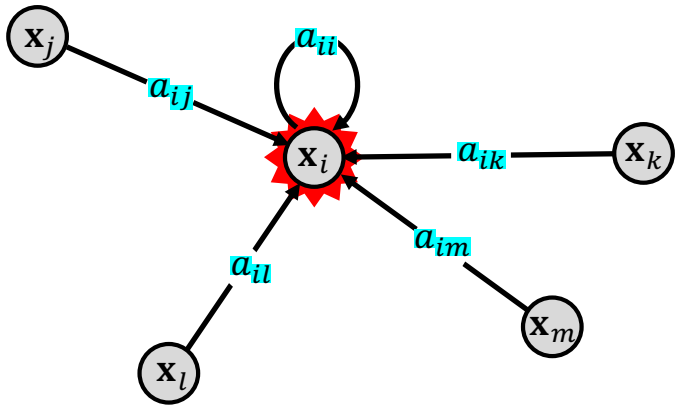
Convolutional GNNs



$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} a_{ij} \psi(\mathbf{x}_j) \right)$$

Defferard et al. 2016; Kipf, Welling 2016 (GCN)

Convolutional GNNs

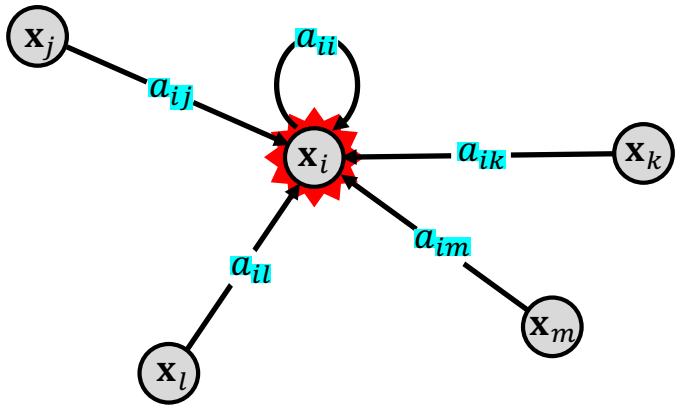


$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} a_{ij} \psi(\mathbf{x}_j) \right)$$

graph adjacency

Defferard et al. 2016; Kipf, Welling 2016 (GCN)

Convolutional GNNs



$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} a_{ij} \psi(\mathbf{x}_j) \right)$$

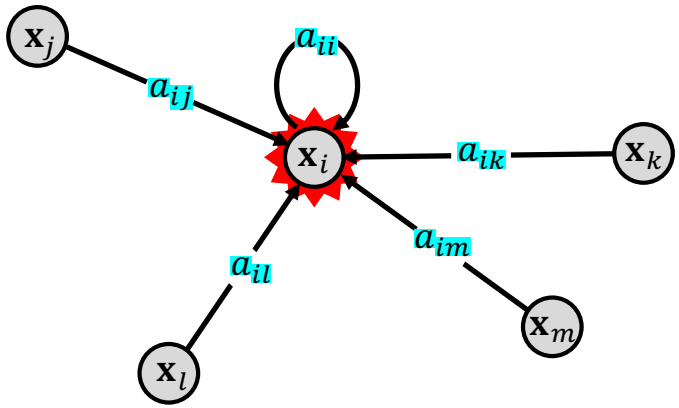
nonlinear activation

graph adjacency

node-wise transformation

Defferard et al. 2016; Kipf, Welling 2016 (GCN)

Convolutional GNNs



$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} a_{ij} \mathbf{W} \mathbf{x}_j \right)$$

nonlinear activation

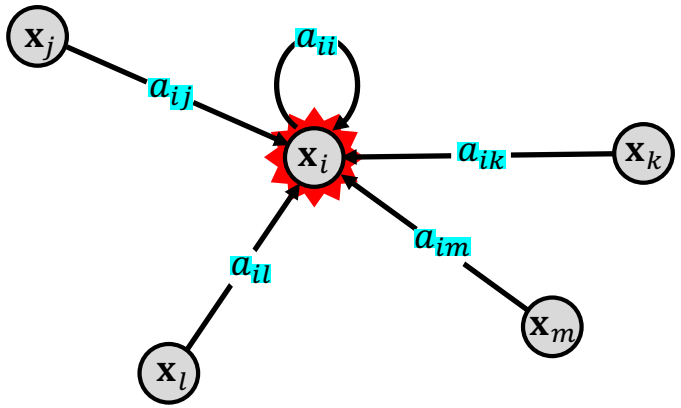
graph adjacency

node-wise linear transformation

Defferard et al. 2016; Kipf, Welling 2016 (GCN)

Convolutional GNNs

- Simplest GNN
- Highly scalable
- Industrial use cases
- Folklore: works only on homophilic graphs



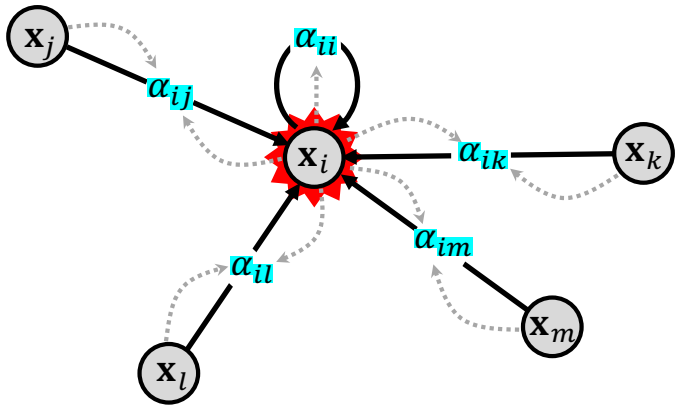
$$\mathbf{X} \leftarrow \sigma(\mathbf{AXW})$$

diffusion $n \times n$

channel mixing $d \times d$

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

Attentional GNNs

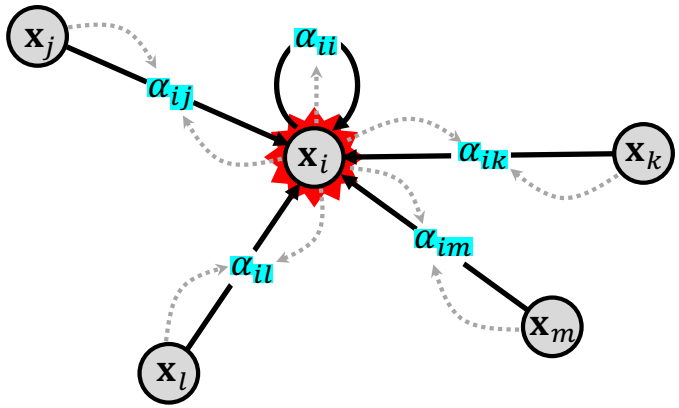


$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} \alpha_{ij}(\mathbf{x}_i, \mathbf{x}_j)\psi(\mathbf{x}_j) \right)$$

learnable attention weights

Monti et al. 2017; Veličković et al. 2018 (GAT)

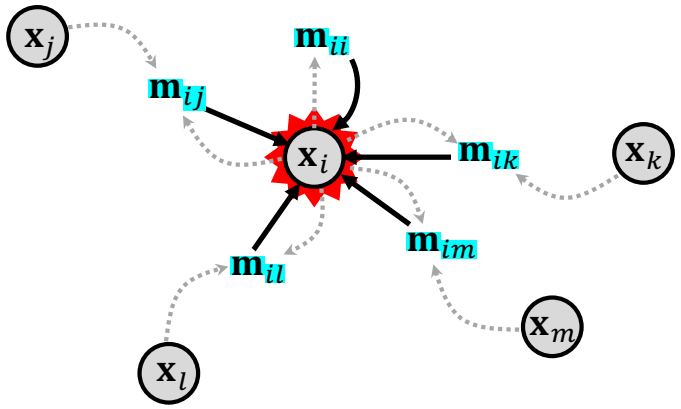
Attentional GNNs



$$\mathbf{X} \leftarrow \sigma(\mathbf{A}(\mathbf{X})\mathbf{X})$$

Monti et al. 2017; Veličković et al. 2018 (GAT)

Message-Passing GNNs



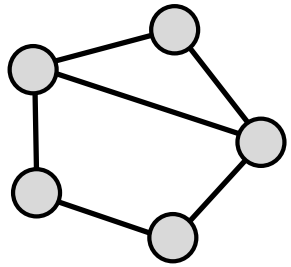
$$\mathbf{x}_i \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{i\}} \psi(\mathbf{x}_i, \mathbf{x}_j) \right)$$

message from node j to node i

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

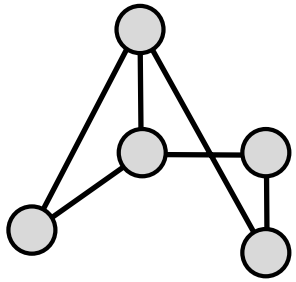
EXPRESSIVE POWER OF GNNS & WEISFEILER-LEHMAN TEST

Graph isomorphism



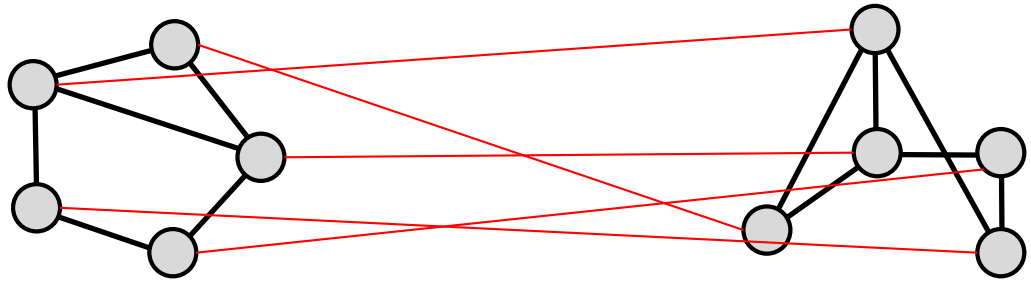
$G = (V, E)$

“=”



$G' = (V', E')$

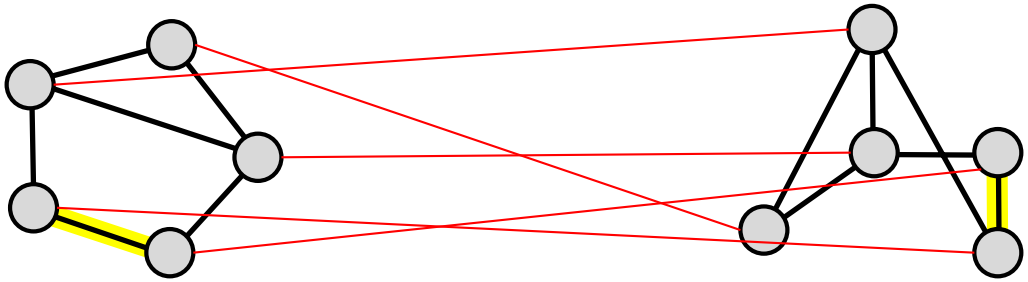
Graph isomorphism



$G = (V, E)$ $G' = (V', E')$

- Two graphs $G = (V, E)$ and $G' = (V', E')$ are **isomorphic** if there exists an *edge-preserving bijection* $\varphi: V \to V'$ s.t. $u \sim v$ in G iff $\varphi(u) \sim \varphi(v)$ in G'

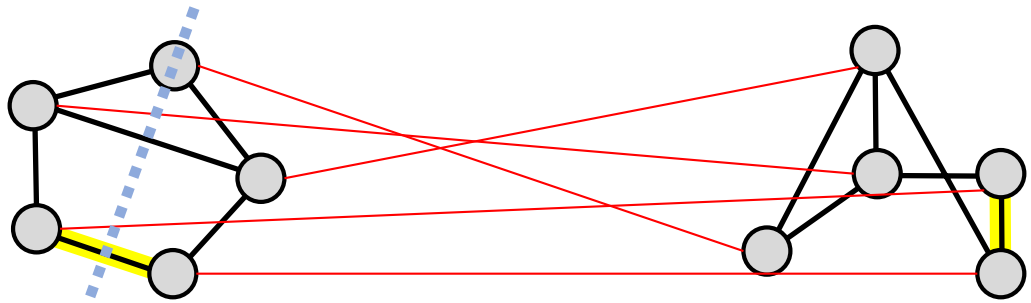
Graph isomorphism



$G = (V, E)$ $\qquad\qquad$ $G' = (V', E')$

- Two graphs $G = (V, E)$ and $G' = (V', E')$ are **isomorphic** if there exists an *edge-preserving bijection* $\varphi: V \to V'$ s.t. $u \sim v$ in G iff $\varphi(u) \sim \varphi(v)$ in G'

Graph isomorphism



- Two graphs $G = (V, E)$ and $G' = (V', E')$ are **isomorphic** if there exists an *edge-preserving bijection* $\varphi: V \to V'$ s.t. $u \sim v$ in G iff $\varphi(u) \sim \varphi(v)$ in G'

Note: φ is not unique where the graph has symmetries (edge-preserving automorphism)

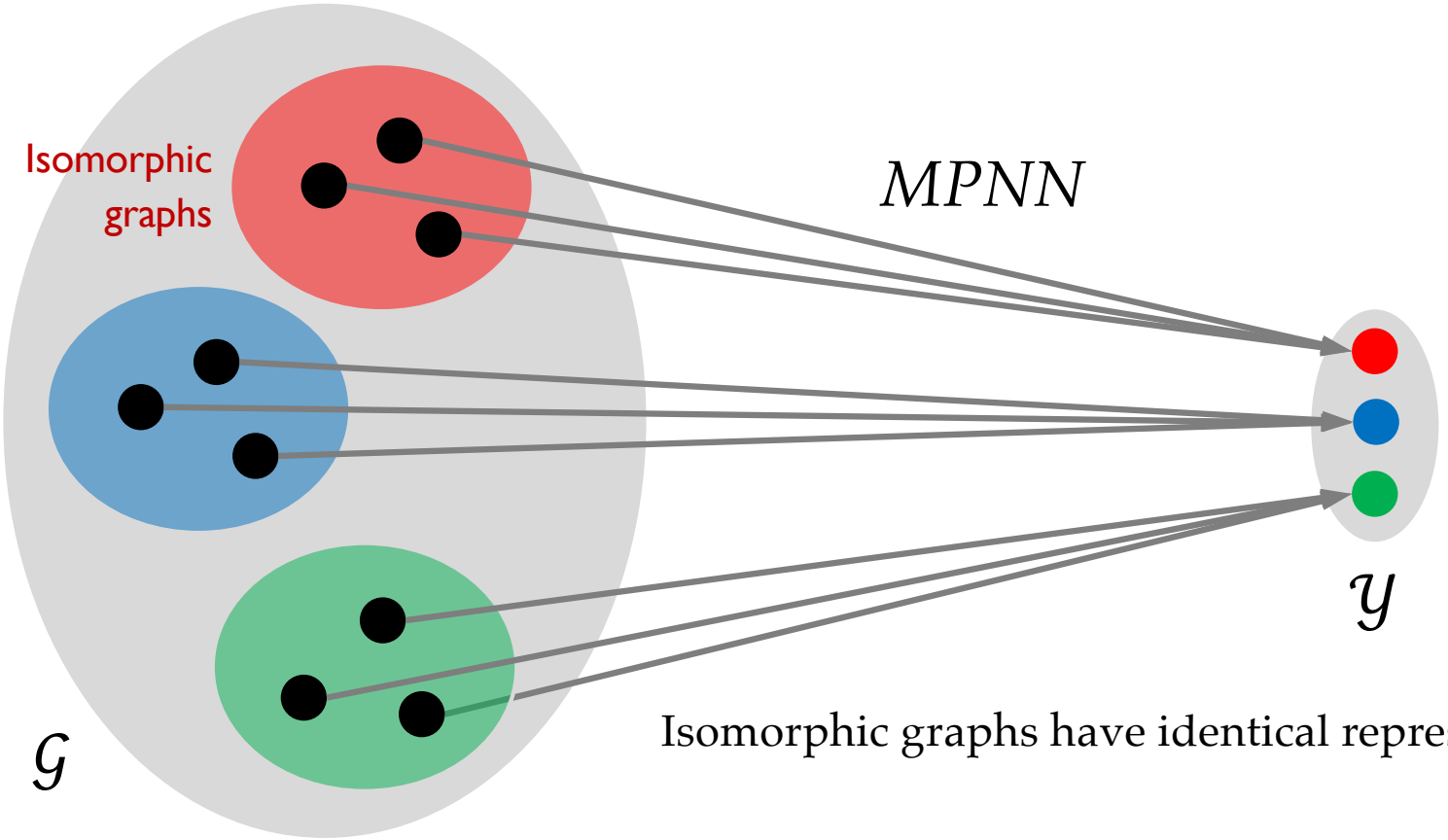
Universal Approximation on Graphs

Theorem: A class of functions is universally approximating permutation-invariant functions on graphs with finite node features iff it can discriminate graph isomorphisms.

Universal approximation on graphs is equivalent to graph isomorphism testing

Chen 2019

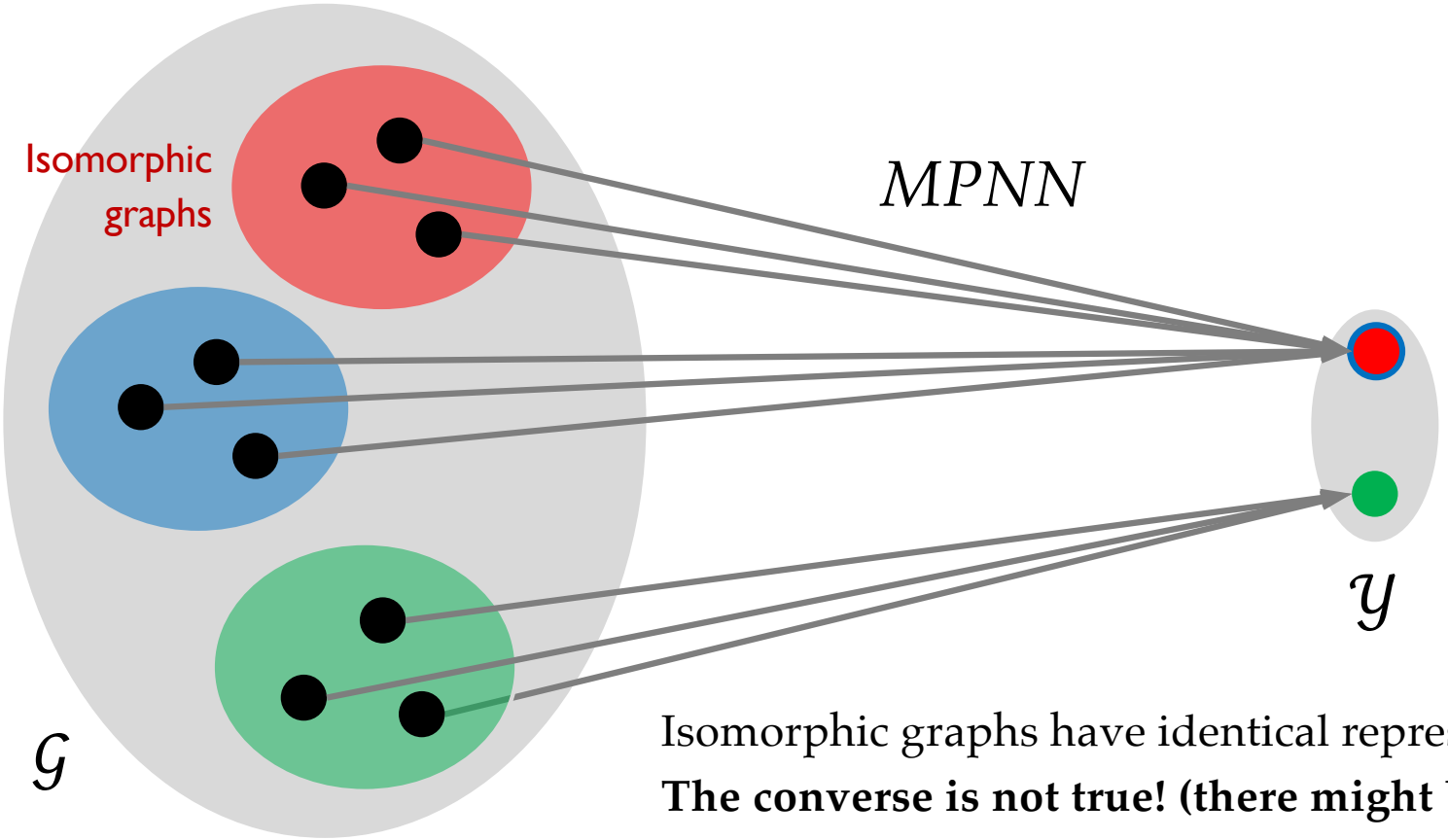
What graphs can MPNNs represent?



Isomorphic graphs have identical representations

Adapted from Bevilacqua et al.
(LoG Tutorial 2022)

What graphs can MPNNs represent?

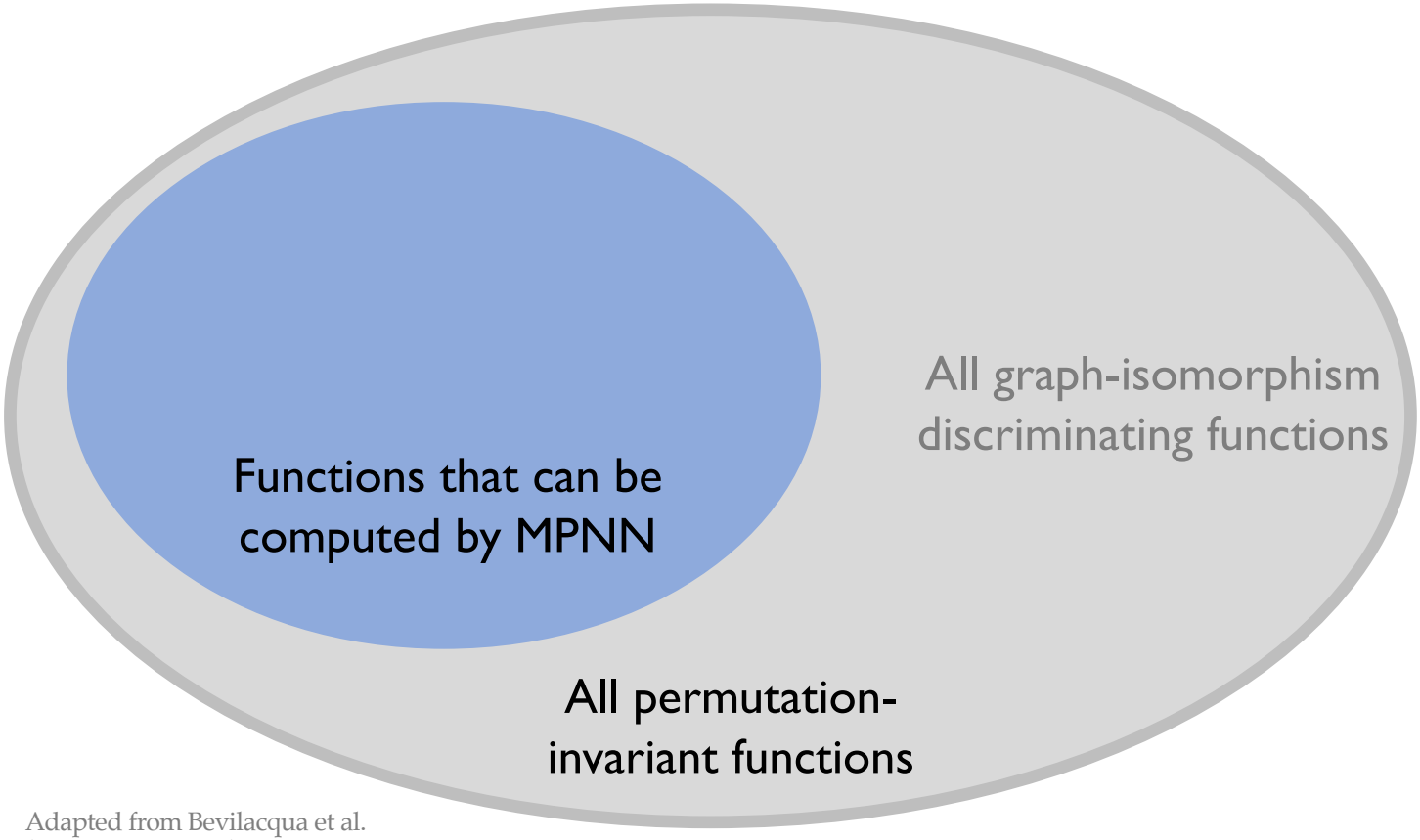


Adapted from Bevilacqua et al.
(LoG Tutorial 2022)

Isomorphic graphs have identical representations

The converse is not true! (there might be indistinguishable non-isomorphic graphs)

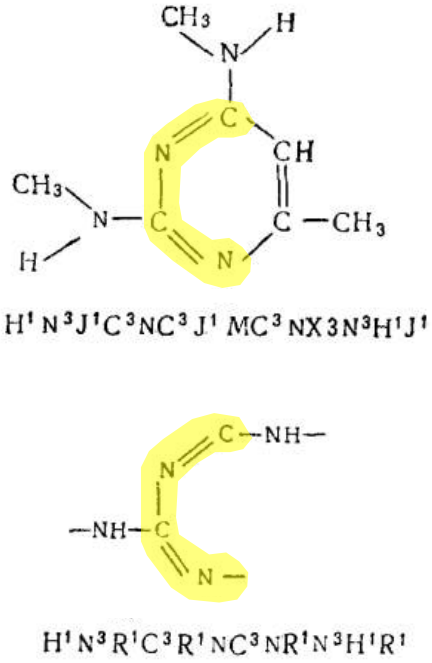
Expressive power of MPNNs



Adapted from Bevilacqua et al. (LoG Tutorial 2022)

Weisfeiler-Lehman Test & Chemical precursors of GNNs

George Vlăduţ

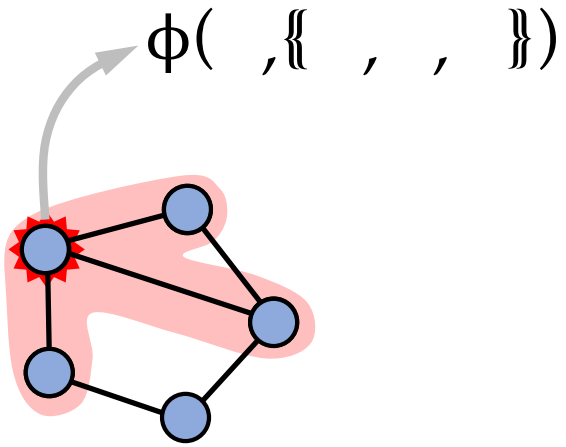


Andrey Lehman

Boris Weisfeiler

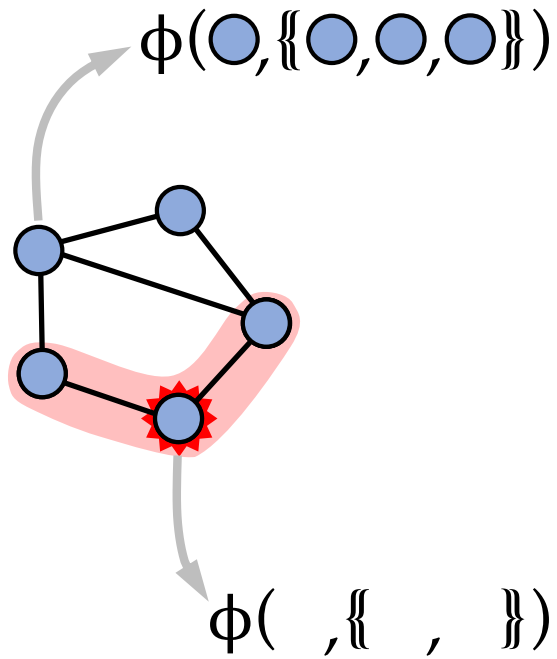
Vlăduţ et al. 1959; Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



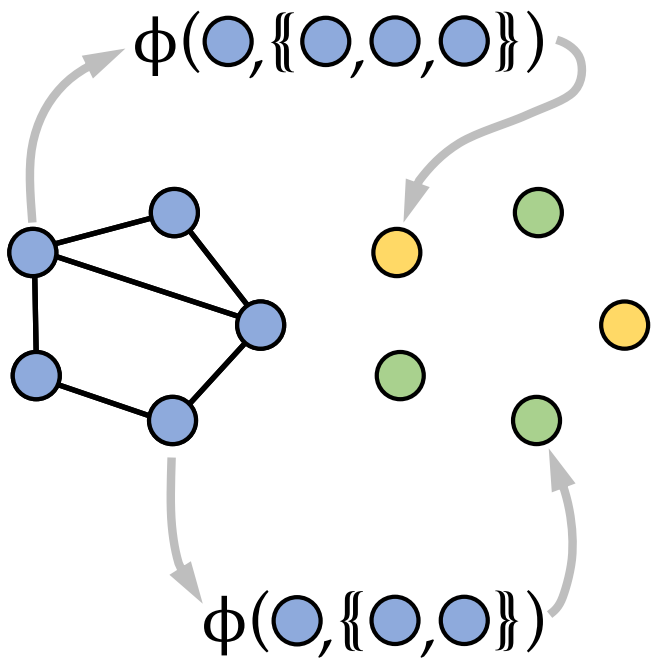
Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



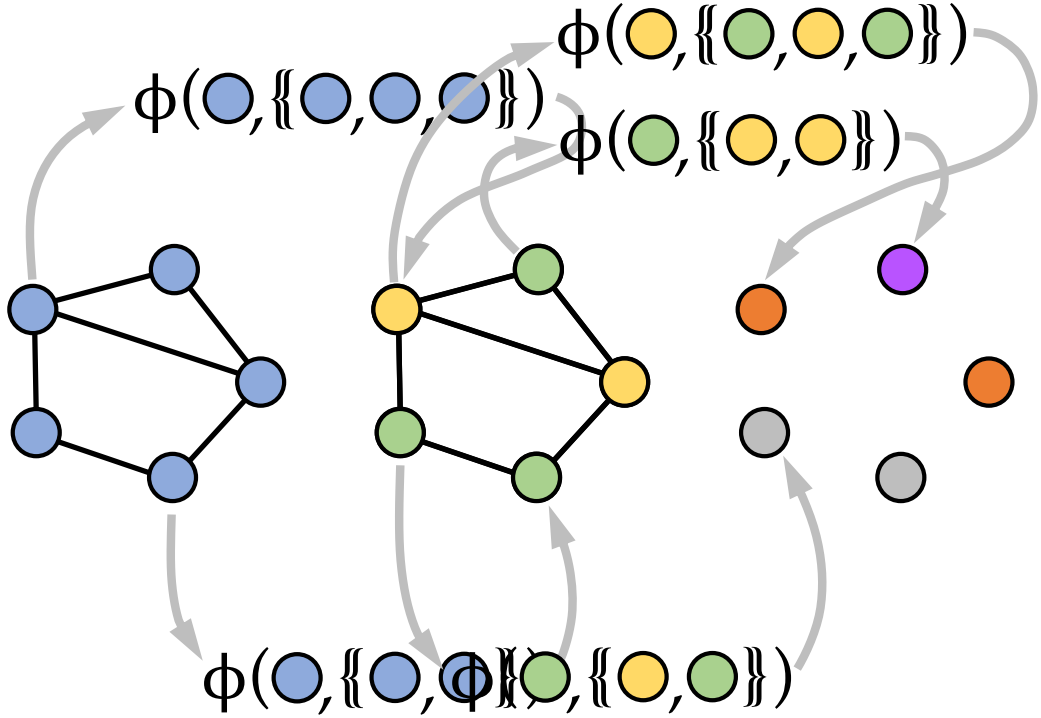
Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



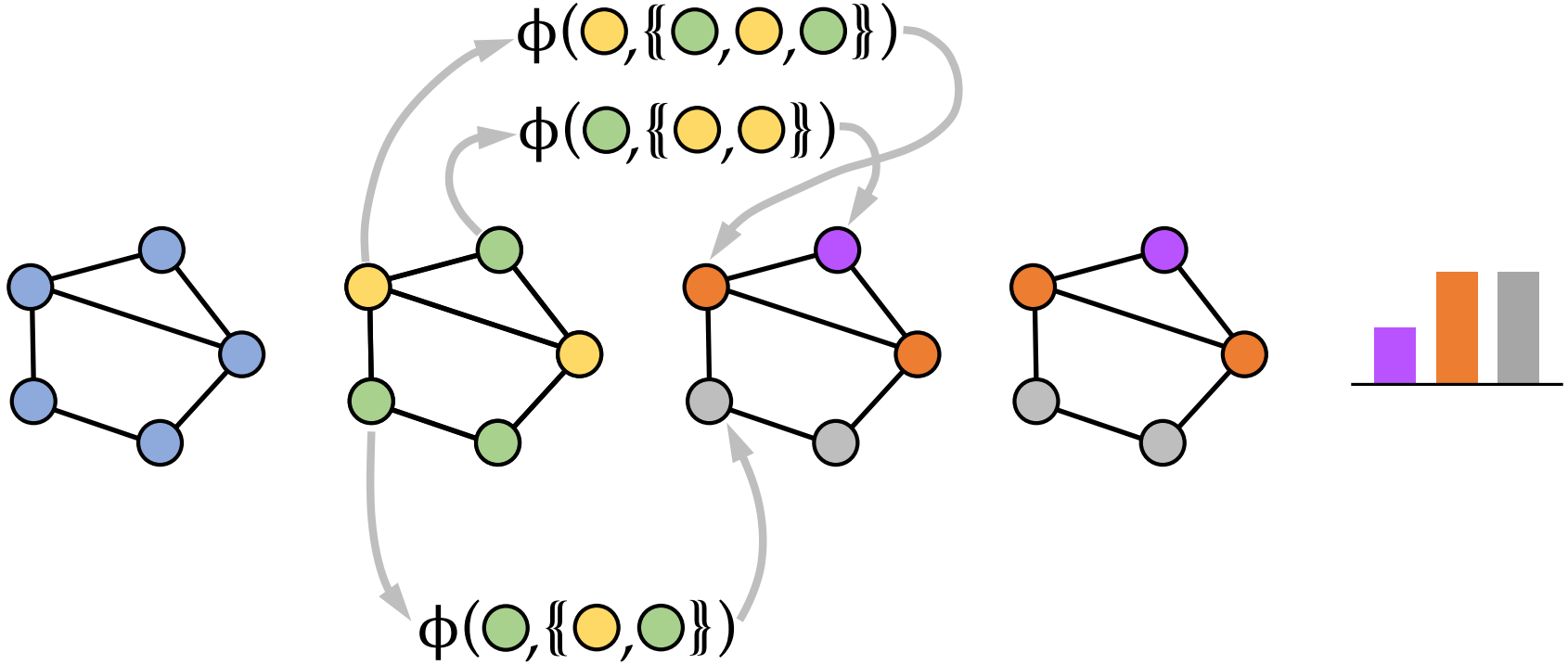
Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



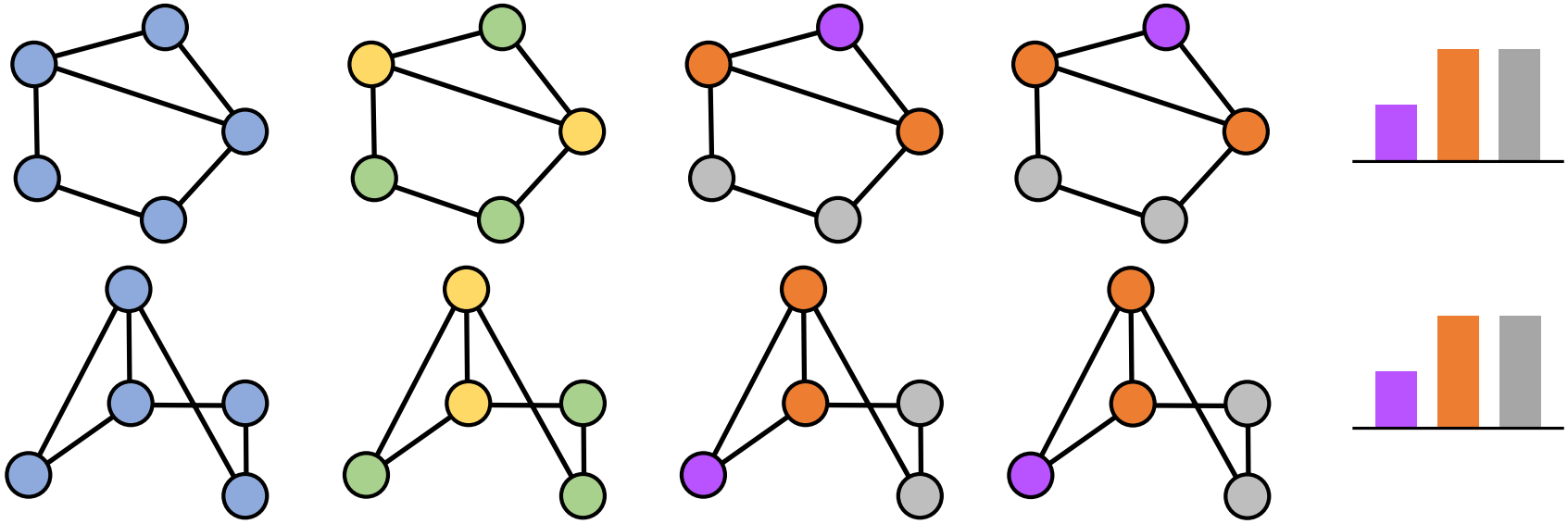
Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



Weisfeiler, Lehman 1968

Weisfeiler-Lehman Test



Weisfeiler, Lehman 1968

non-isomorphic graphs that are WL-equivalent

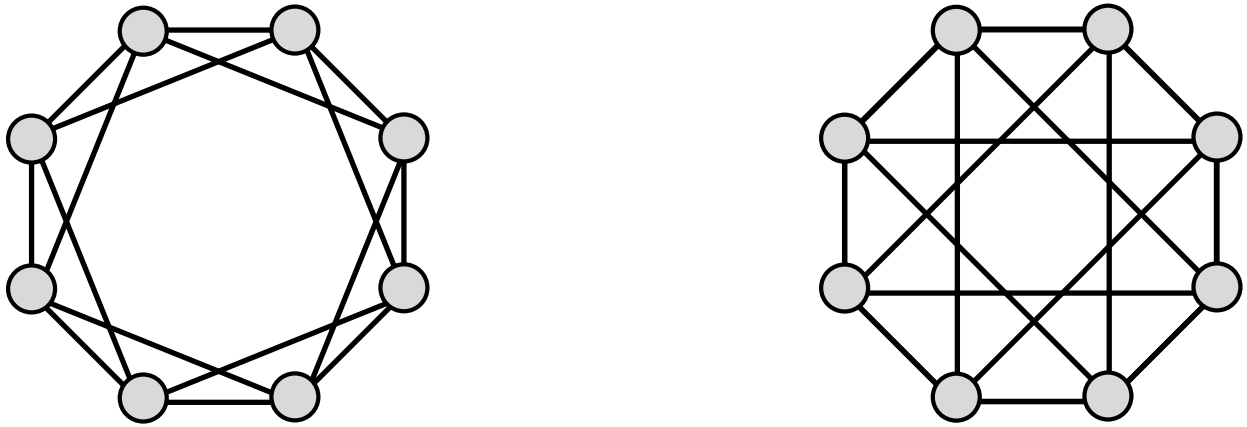
Necessary but insufficient condition for graph isomorphism!

What does WL test see?

=

What WL cannot test?

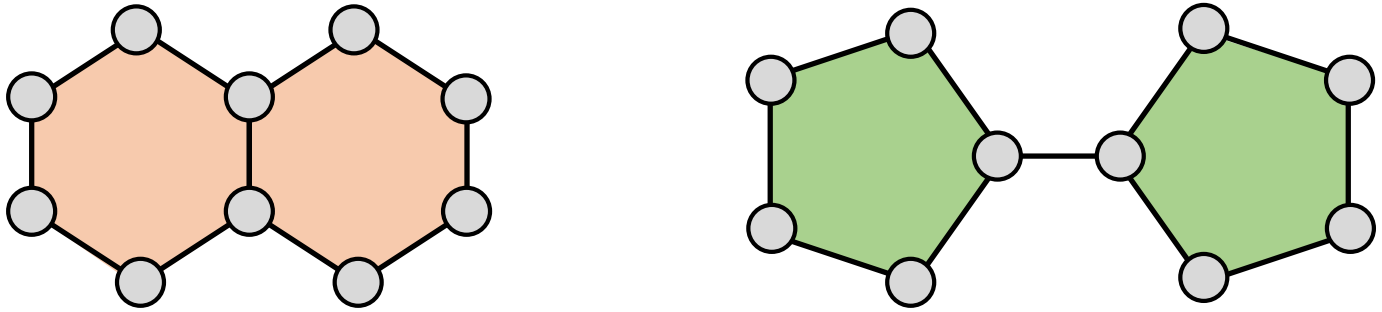
Example of non-isomorphic graphs that cannot be distinguished by Weisfeiler-Lehman test (outputs “possibly isomorphic”)



r-regular graphs (deg=r at every node) with the same number of nodes

What WL cannot test?

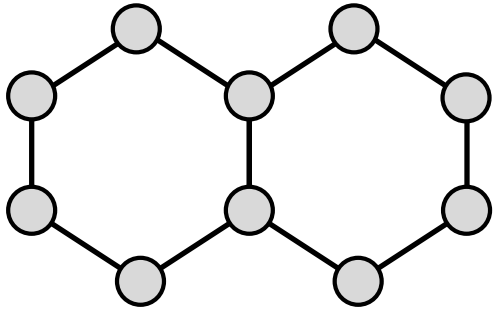
Example of non-isomorphic graphs that cannot be distinguished by Weisfeiler-Lehman test (outputs "possibly isomorphic")



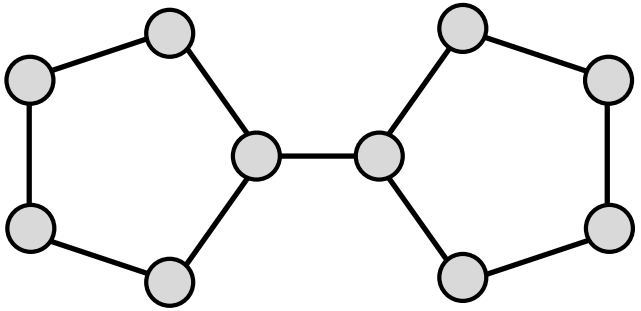
Any induced connected pattern with ≥ 3 *nodes (triangles, cycles, etc.)*

What WL cannot test?

Example of non-isomorphic graphs that cannot be distinguished by Weisfeiler-Lehman test (outputs “possibly isomorphic”)



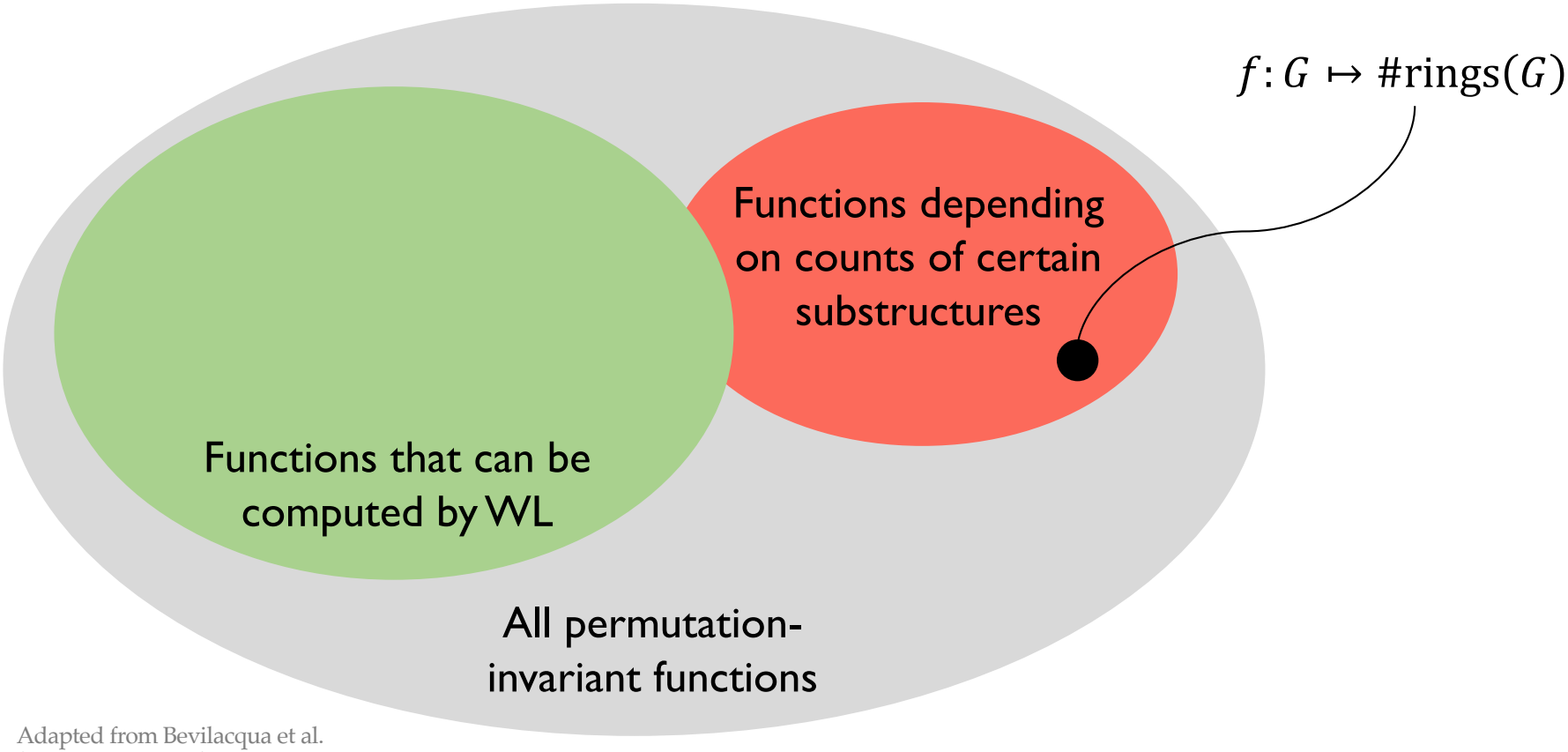
decalin



bicyclopnetyl

Important implications e.g. in chemistry!

Expressive power of Weisfeiler-Lehman

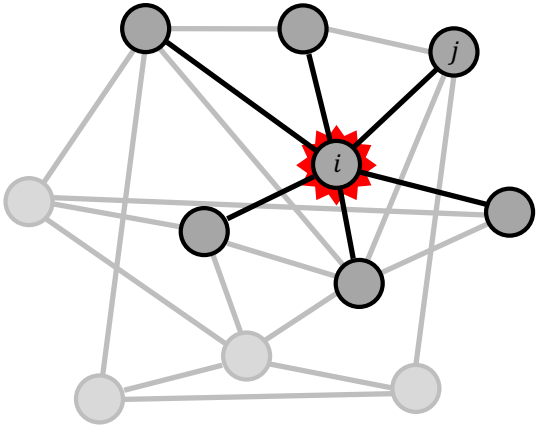


Adapted from Bevilacqua et al. (LoG Tutorial 2022)

MPNNs vs WL

- **WL-test:** $\mathbf{x}_i \leftarrow \phi\left(\mathbf{x}_i, \{\!\{\mathbf{x}_j : j \in \mathcal{N}_i\}\!\}\right)$
  where ϕ is *injective* (hash function)

- **MPNN:** $\mathbf{x}_i \leftarrow \phi\left(\mathbf{x}_i, \square_{j \in \mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$

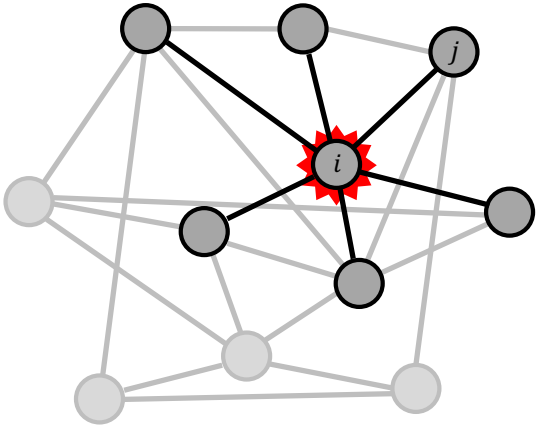


MPNN expressive power is upper-bounded by the Weisfeiler-Lehman test

MPNNs vs WL

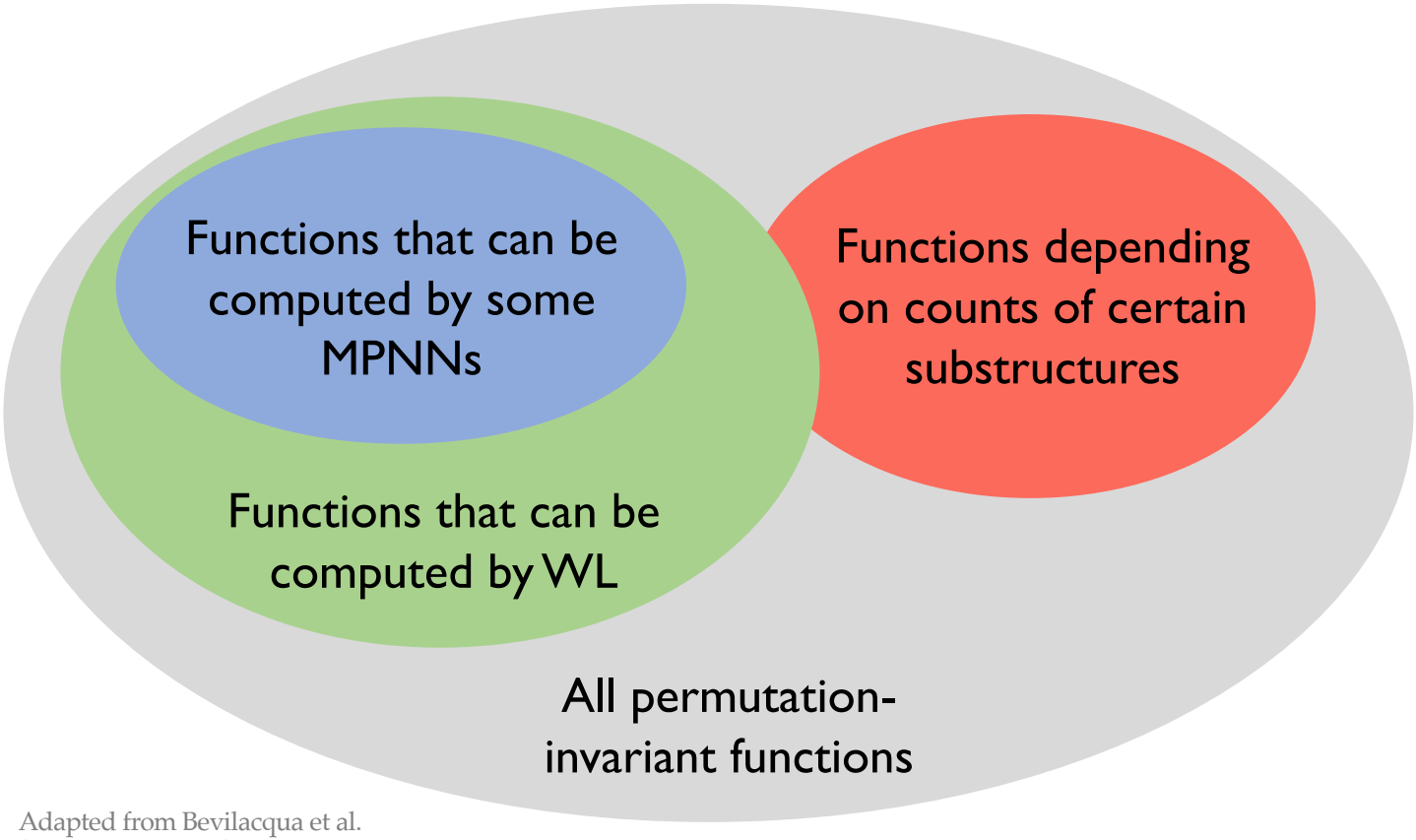
- **WL-test:** $\mathbf{x}_i \leftarrow \phi\left(\mathbf{x}_i, \{\!\{\mathbf{x}_j : j \in \mathcal{N}_i\}\!\}\right)$
  where ϕ is *injective* (hash function)

- **MPNN:** $\mathbf{x}_i \leftarrow \phi\left(\mathbf{x}_i, \square_{j \in \mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$



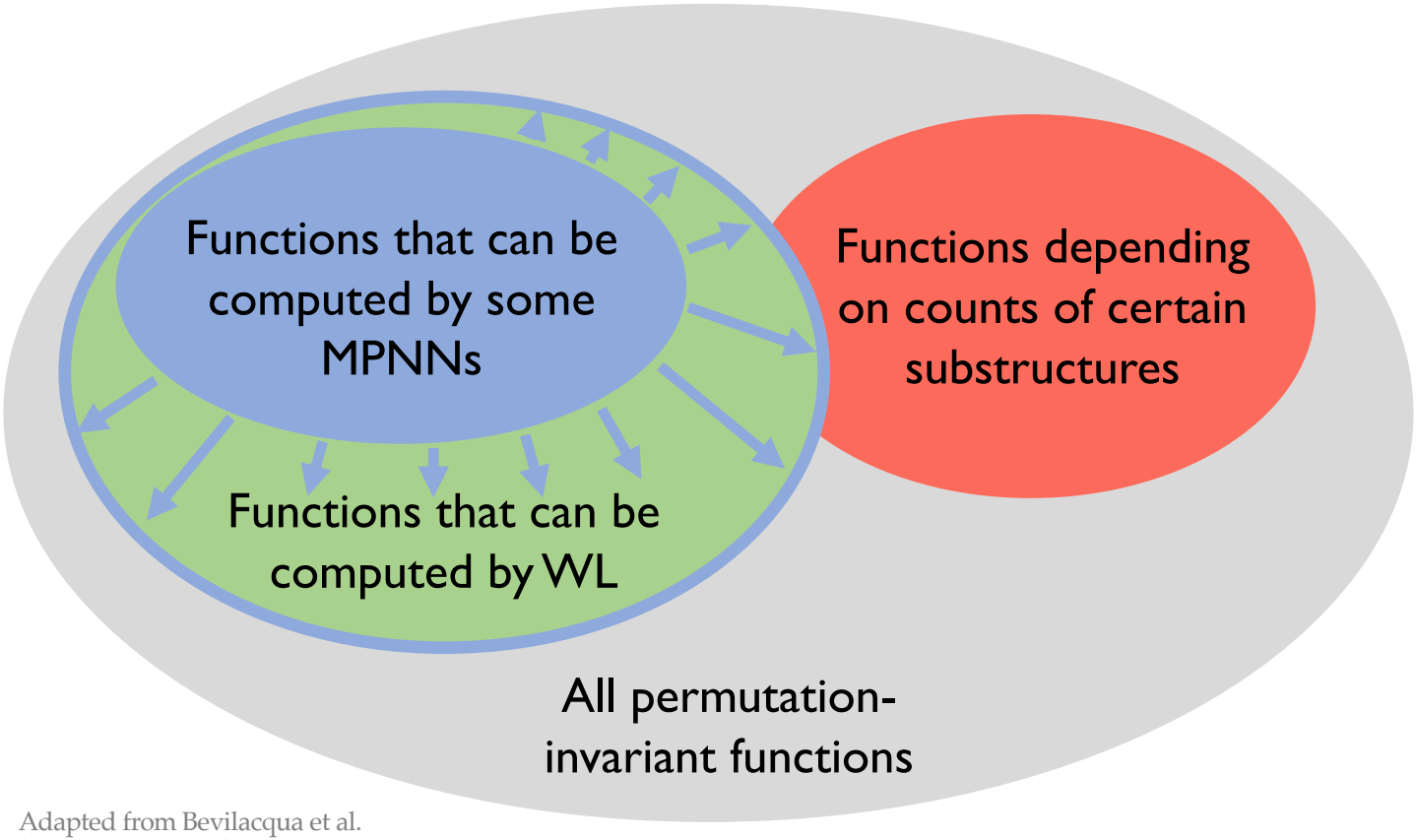
When is MPNN as expressive as the Weisfeiler-Lehman test?

Expressive power of MPNNs



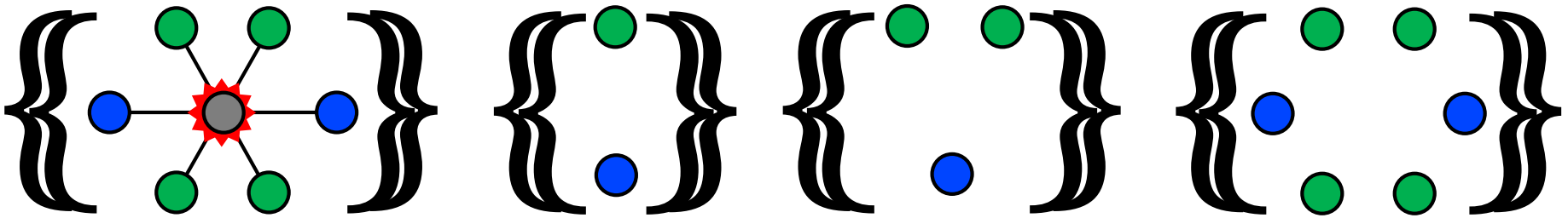
Adapted from Bevilacqua et al. (LoG Tutorial 2022)

Expressive power of MPNNs



Adapted from Bevilacqua et al. (LoG Tutorial 2022)

Are all Aggregators the same?

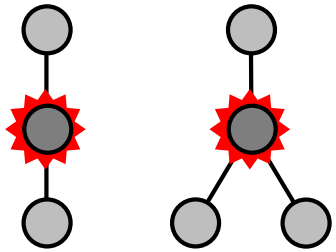


Input **max** **mean** **sum**

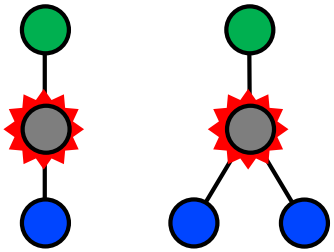
"skeleton"
of the multiset

Distribution
of the multiset

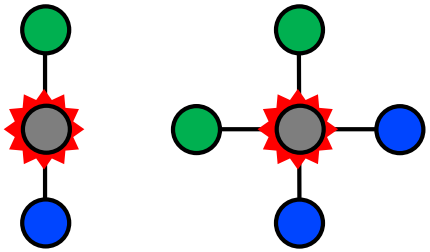
Are all Aggregators the same?



mean and **max**
fail to distinguish



max
fails to distinguish



mean and **max**
fail to distinguish

Graph Isomorphism Network (GIN)

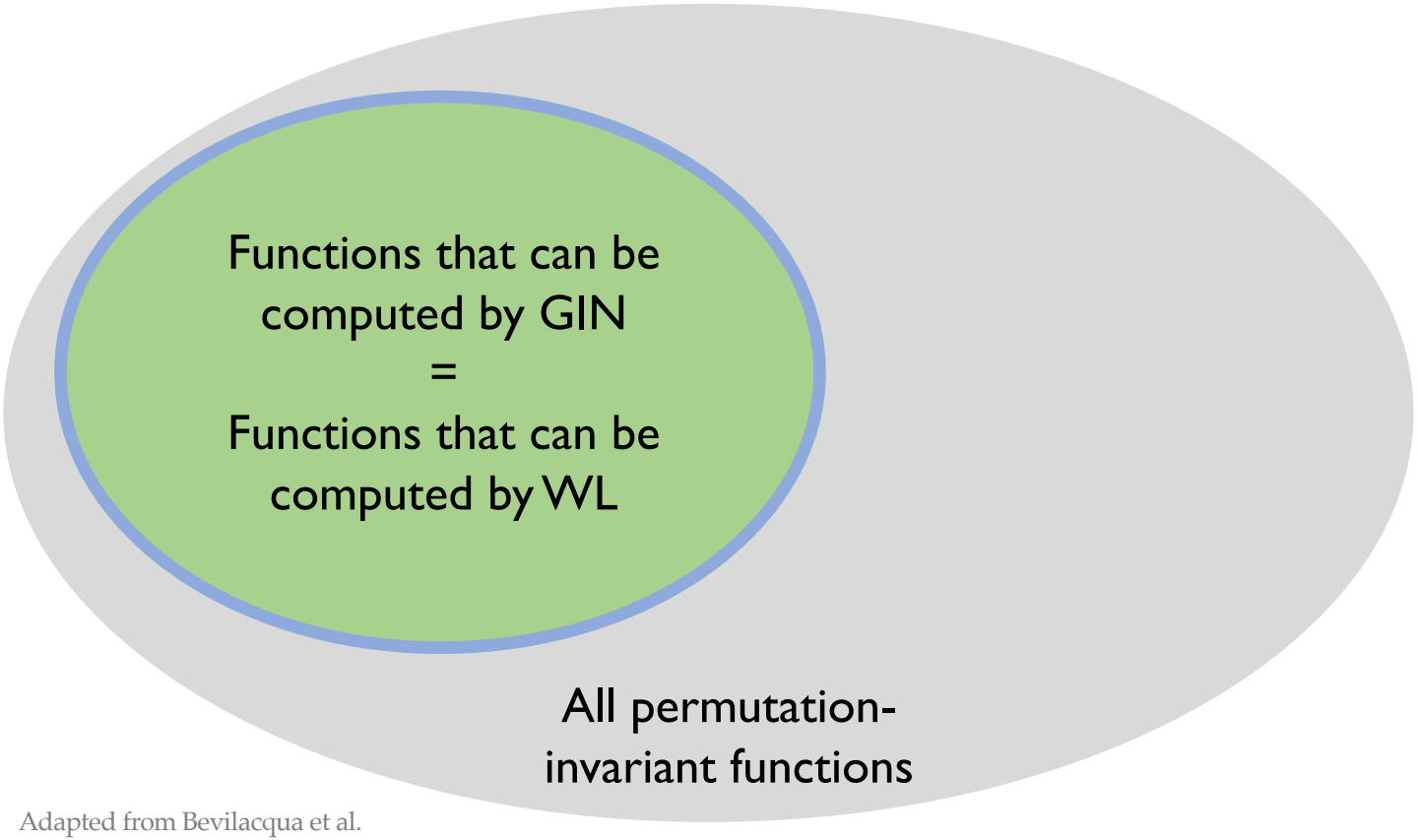
Theorem: Assume graph node features are from a countable set. Then, an MPNN with with injective aggregator $\square$, update function ϕ, and graph-wise readout function, is as powerful as the Weisfeiler-Lehman test.

- Assumption of discrete countable features (often not the case in practice)
- GIN: uses an injective multiset function of the form

$$\text{MLP}\left((1+\epsilon)\mathbf{x}_i + \sum_{j \in \mathcal{N}_i} \mathbf{x}_j\right)$$

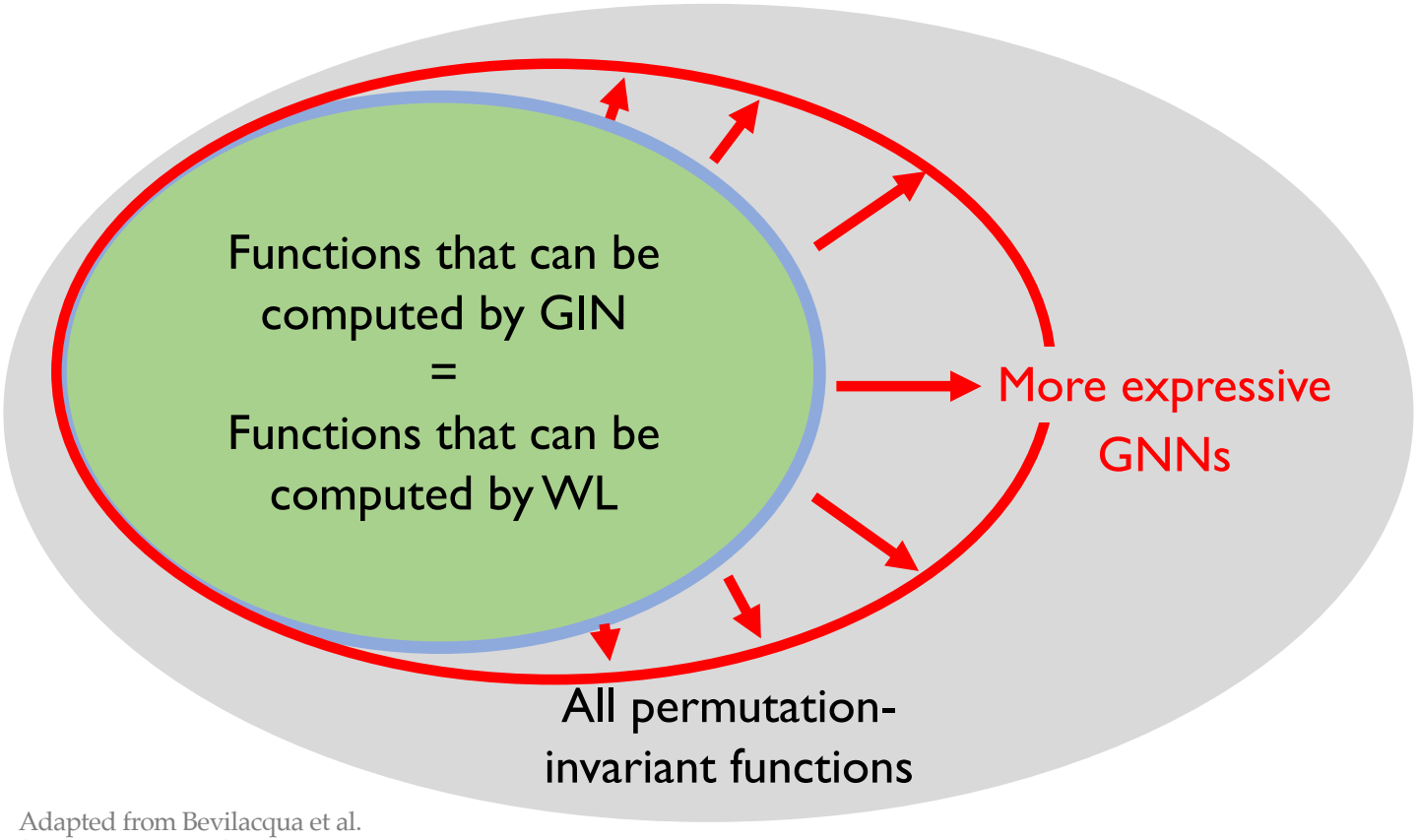
Xu 2019; Morris 2019; Zaheer et al. 2017 (DeepSets)

Expressive power of GIN ("best MPNN")



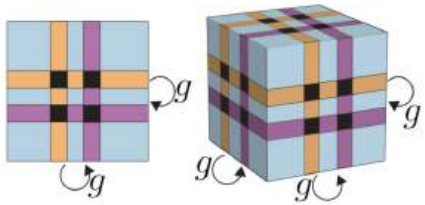
Adapted from Bevilacqua et al. (LoG Tutorial 2022)

Expressive power of GIN ("best MPNN")



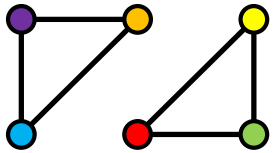
Adapted from Bevilacqua et al. (LoG Tutorial 2022)

Towards More Expressive GNNs



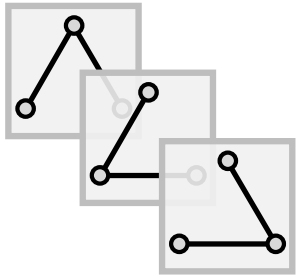
Higher-order
WL tests

Maron et al. 2019
Morris et al. 2019



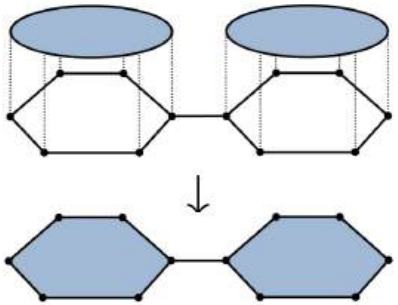
Positional &
Structural encoding

Monti, Otness et B 2018
Sato 2020
Dwivedi et al. 2020
Bouritsas, Frasca et B 2020
…many more



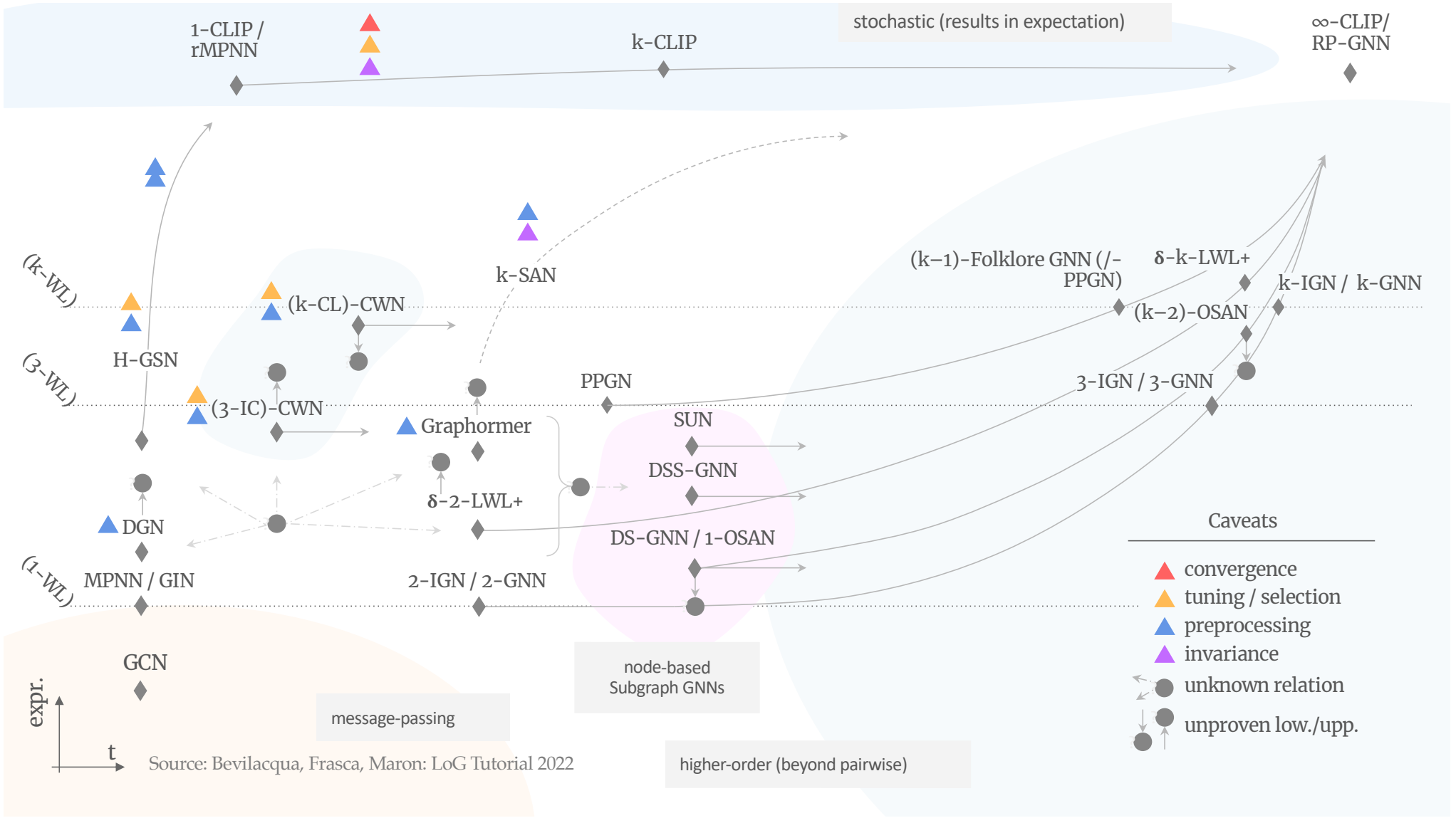
Subgraph
GNNs

Papp et al. 2021
Cotta et al. 2021
Zhao et al. 2021
Bevilacqua, Frasca et B, Maron 2021
Frasca et B, Maron 2022



Topological
message passing

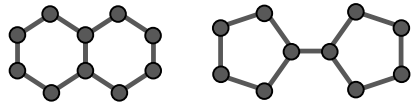
Bodnar, Frasca et B 2021


stochastic (results in expectation)
1-CLIP / rMPNN
k-CLIP
∞-CLIP/ RP-GNN
(k-WL)
(3-WL)
(1-WL)
H-GSN
(k-CL)-CWN
k-SAN
(3-IC)-CWN
Graphormer
PPGN
SUN
DSS-GNN
δ-2-LWL+
DS-GNN / 1-OSAN
DGN
MPNN / GIN
2-IGN / 2-GNN
(k−1)-Folklore GNN (/-PPGN)
δ-k-LWL+
k-IGN / k-GNN
(k−2)-OSAN
3-IGN / 3-GNN
Caveats
convergence
tuning / selection
preprocessing
invariance
unknown relation
unproven low./upp.
GCN
expr.
t
message-passing
node-based Subgraph GNNs
higher-order (beyond pairwise)
Source: Bevilacqua, Frasca, Maron: LoG Tutorial 2022

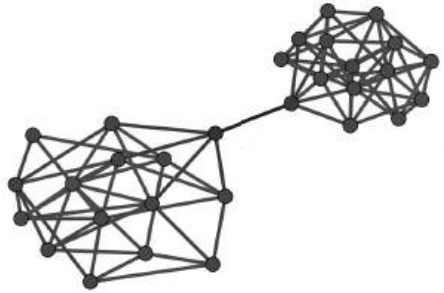
Theory

WL test = expressive power

Practice

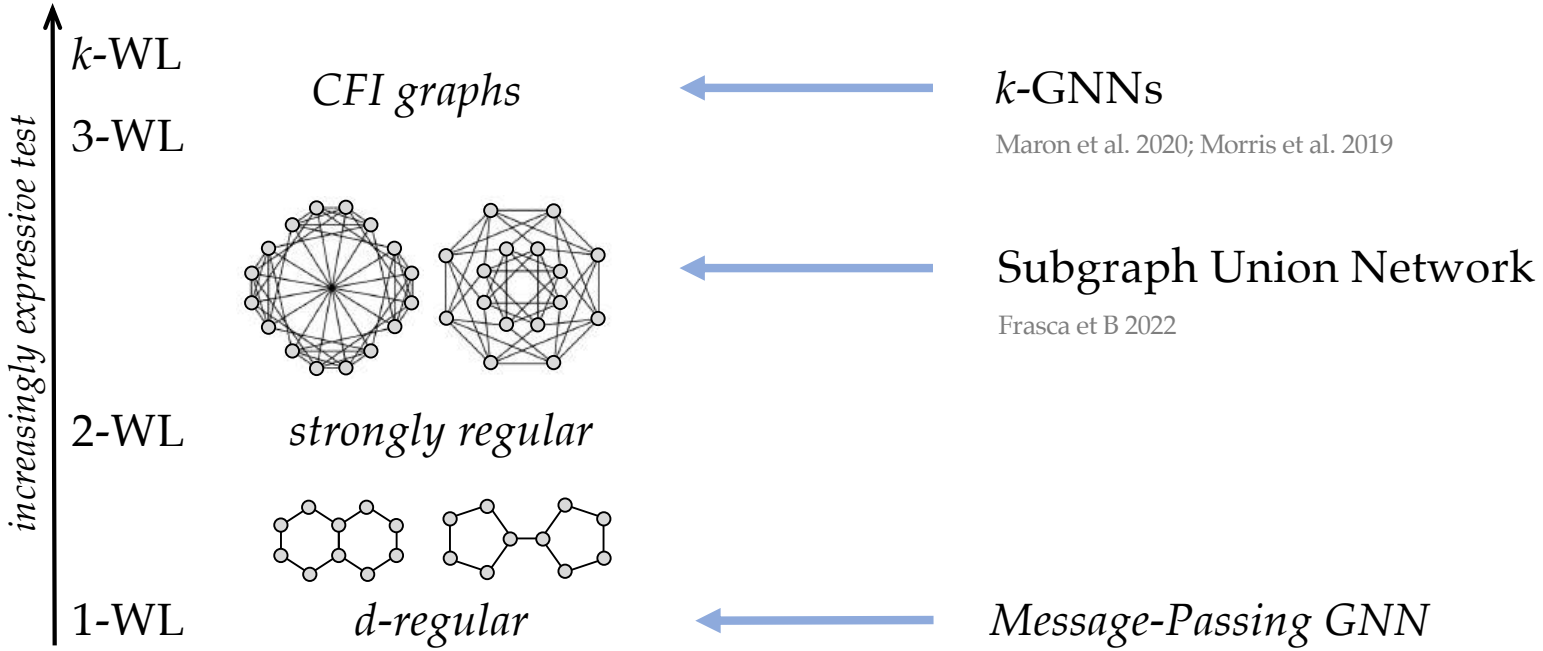


Some non-isomorphic graphs
cannot be tested by WL



Some graphs may be
unfriendly for message passing

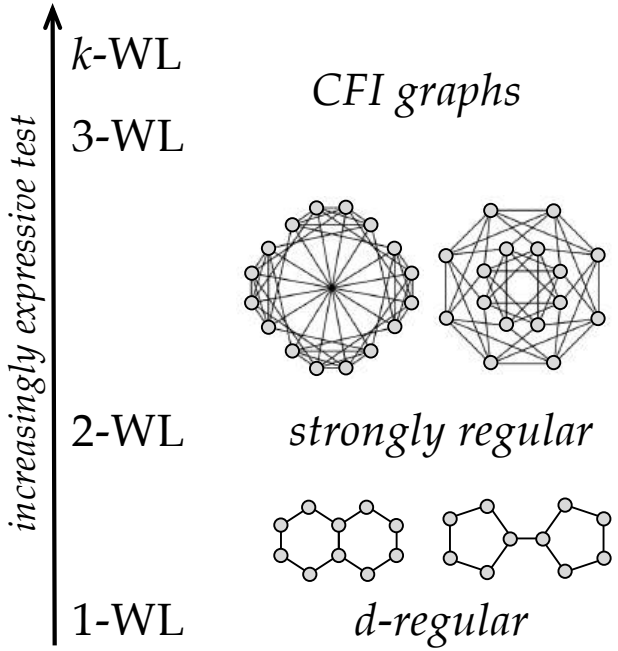
More expressive isomorphism tests (*k*-WL hierarchy)



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)

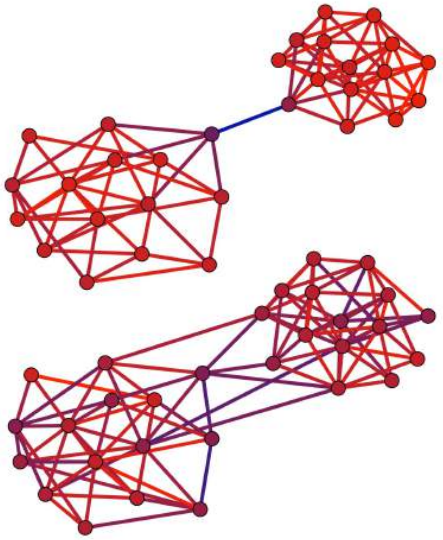


GNN expressive power

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

Gap between Theory & Practice

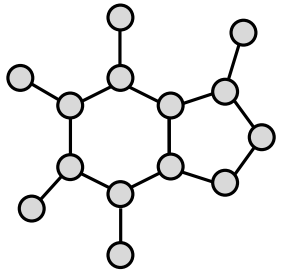
Decouple input graph from the computational graph



Graph rewiring

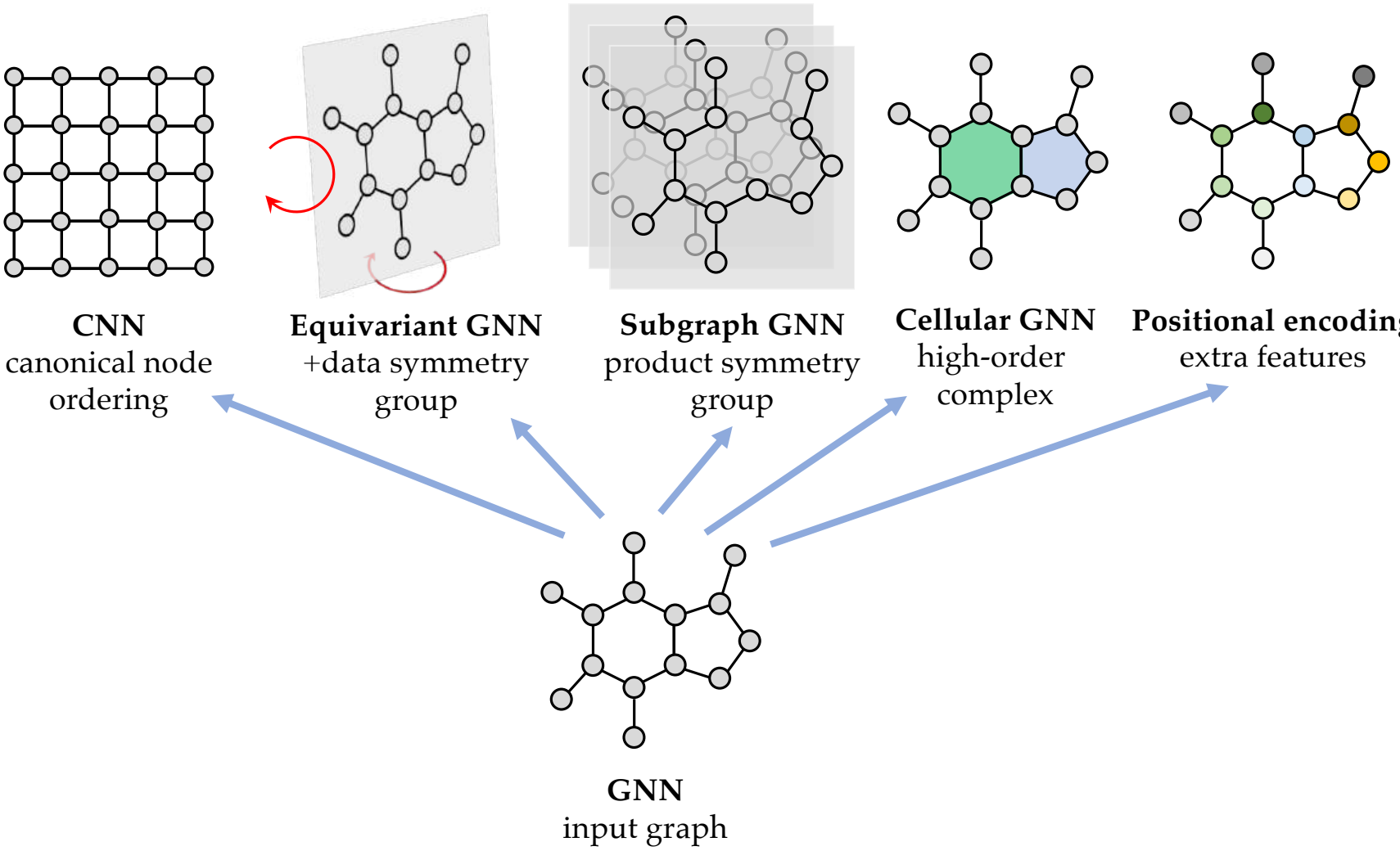
Alon, Yahav 2020 (bottlenecks); Hamilton et al. 2017 (neighbour sampling); Klicpera et al. 2019 (diffusion); Topping, Di Giovanni et B 2022 (Ricci flow); Deac et al. 2022 (expanders); Barbero et B, Di Giovanni 2024 (LASER)

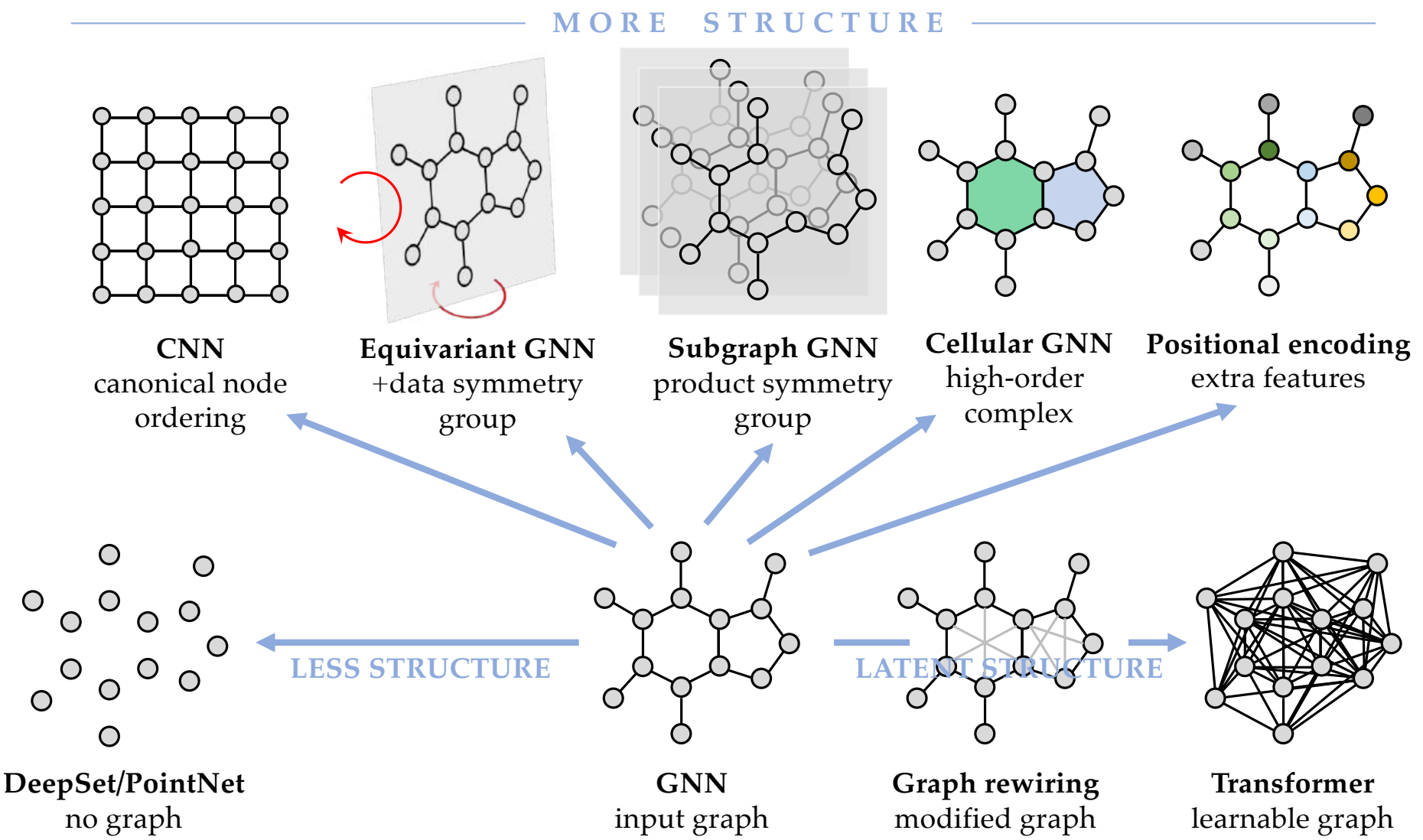
Classical GNNs: propagate information on the input graph



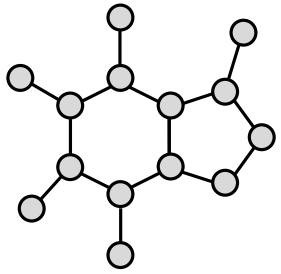
GNN
input graph

MORE STRUCTURE

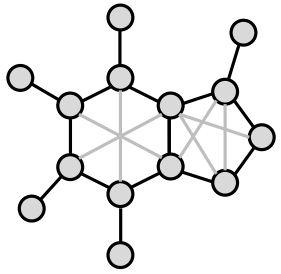



MORE STRUCTURE
CNN
canonical node ordering
Equivariant GNN
+data symmetry group
Subgraph GNN
product symmetry group
Cellular GNN
high-order complex
Positional encoding
extra features
LESS STRUCTURE
LATENT STRUCTURE
DeepSet/PointNet
no graph
GNN
input graph
Graph rewiring
modified graph
Transformer
learnable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

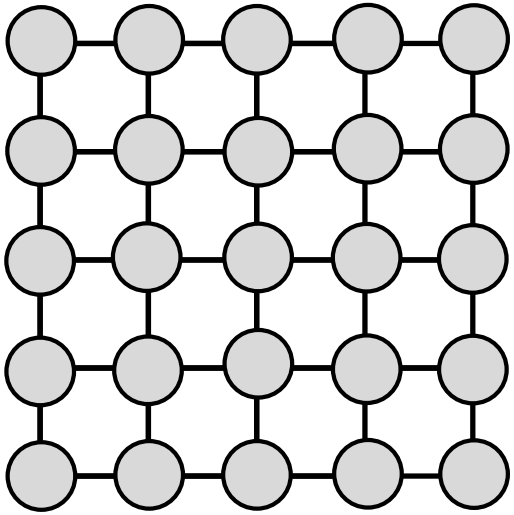
Classical GNNs: propagate information on the input graph



“Modern” GNNs: propagate information on some other “better” graph

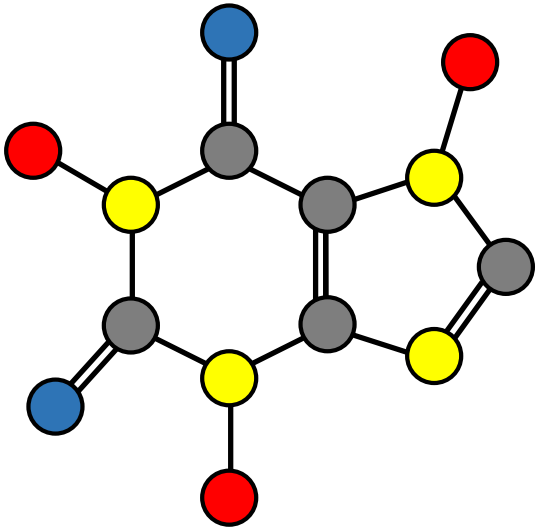


Graphs vs Meshes vs Grids



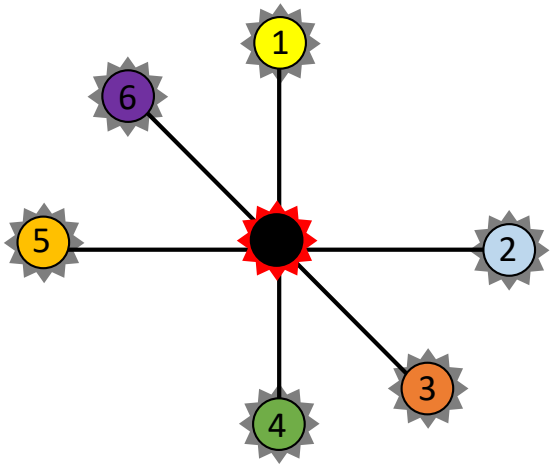
Grid

Mesh



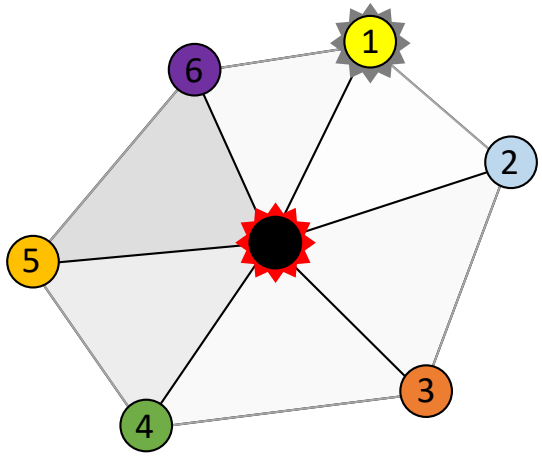
Graph

Graphs vs Meshes vs Grids

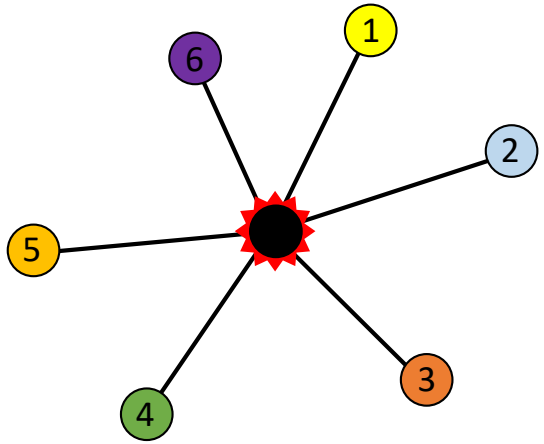


Grid

Fixed

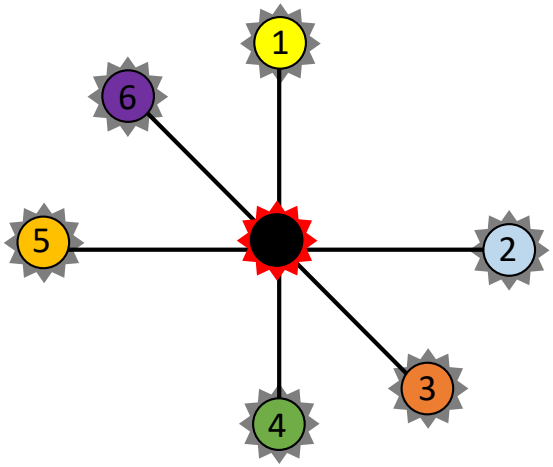


Mesh



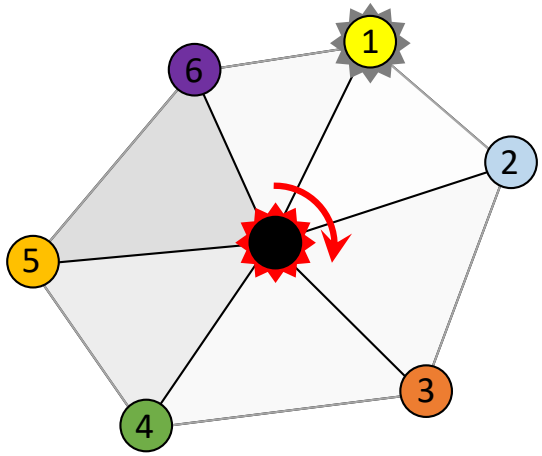
Graph

Graphs vs Meshes vs Grids



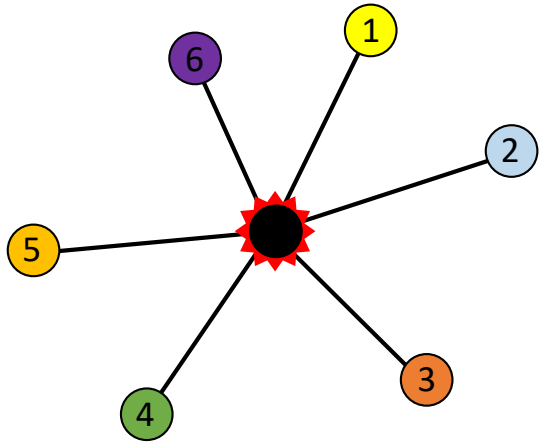
Grid

Fixed



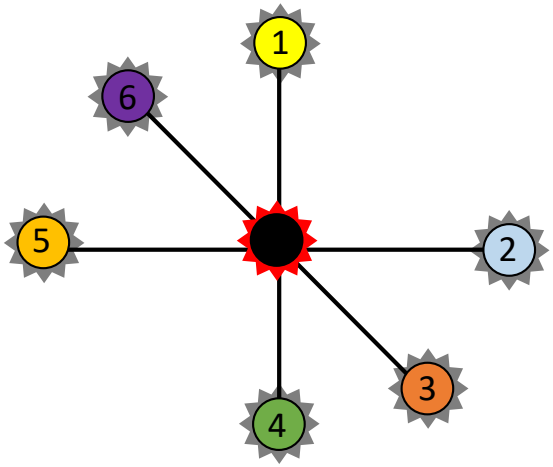
Mesh

Rotation



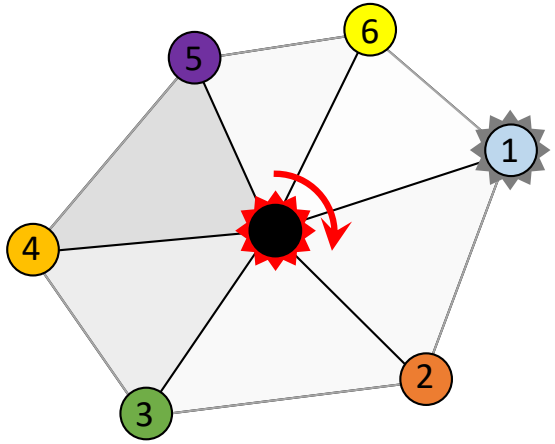
Graph

Graphs vs Meshes vs Grids



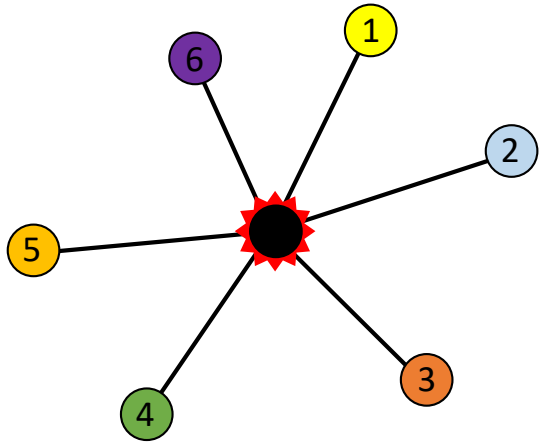
Grid

Fixed



Mesh

Rotation

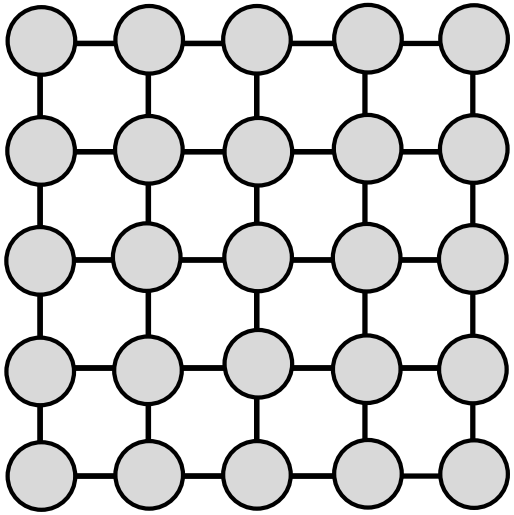


Graph

Permutation

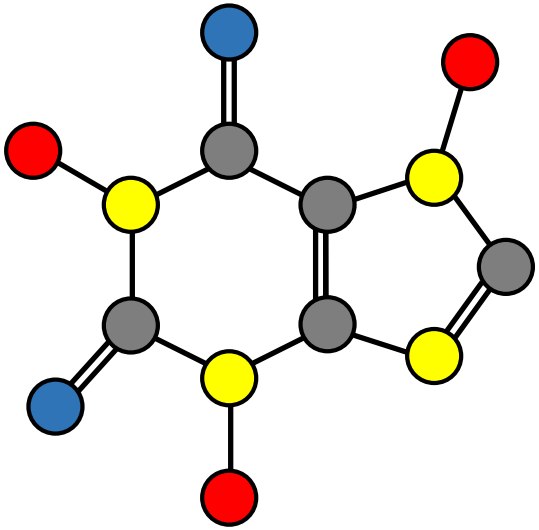
Graphs have the least structure

Graphs vs Meshes vs Grids



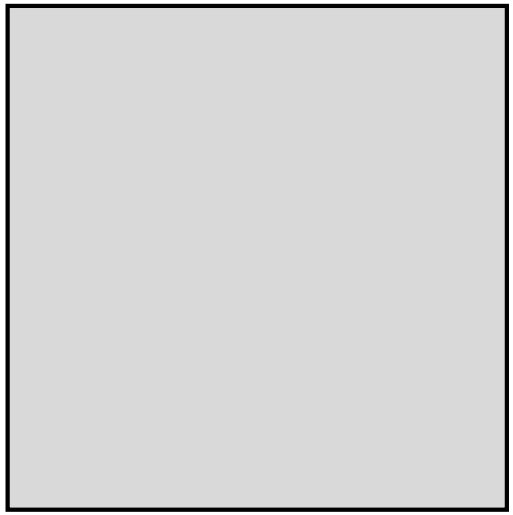
Grid

Mesh



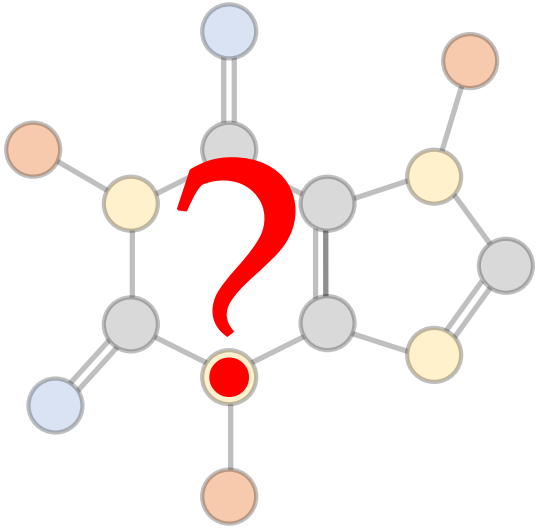
Graph

Graphs vs Meshes vs Grids



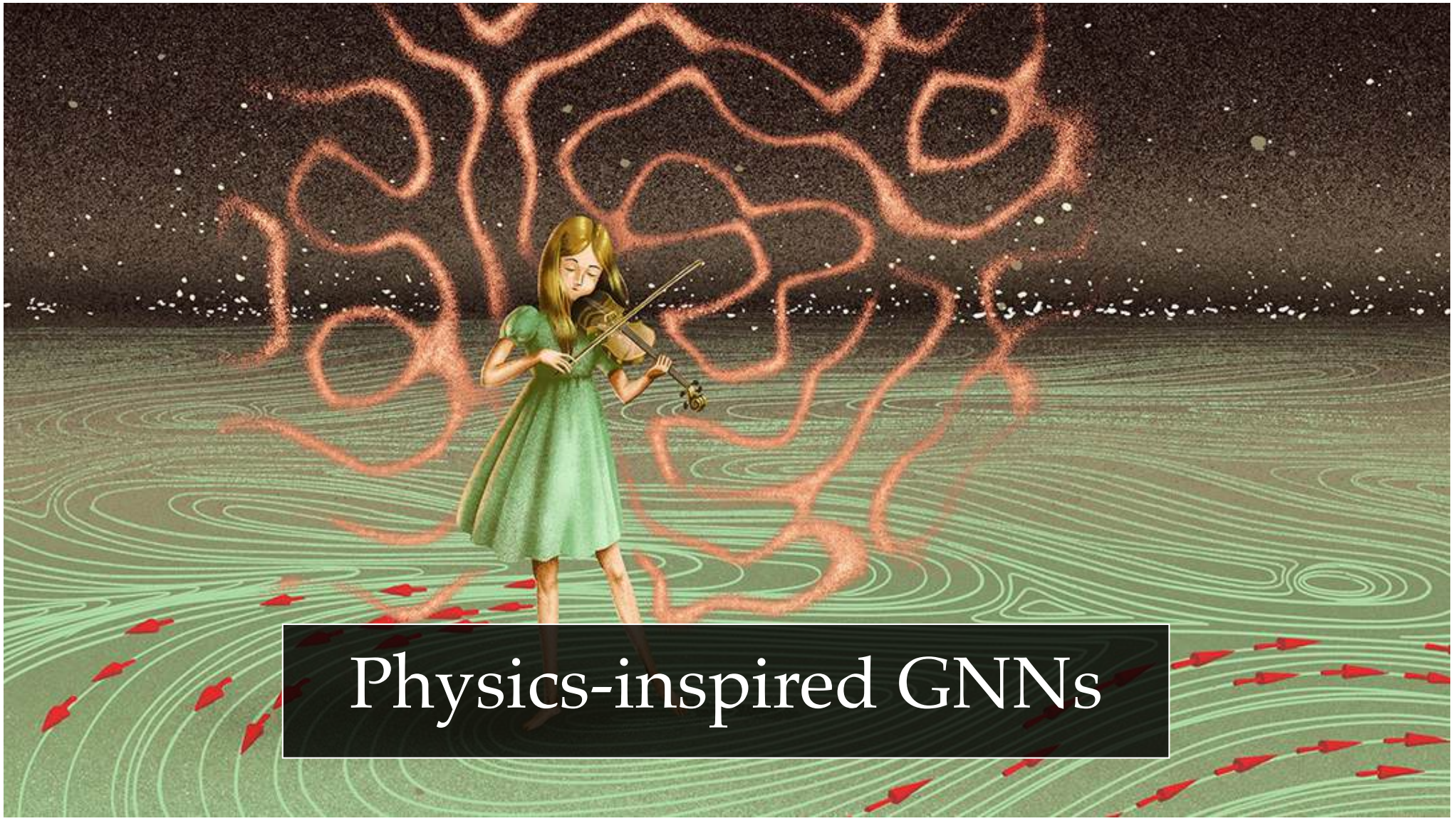
Grid

Mesh



Graph

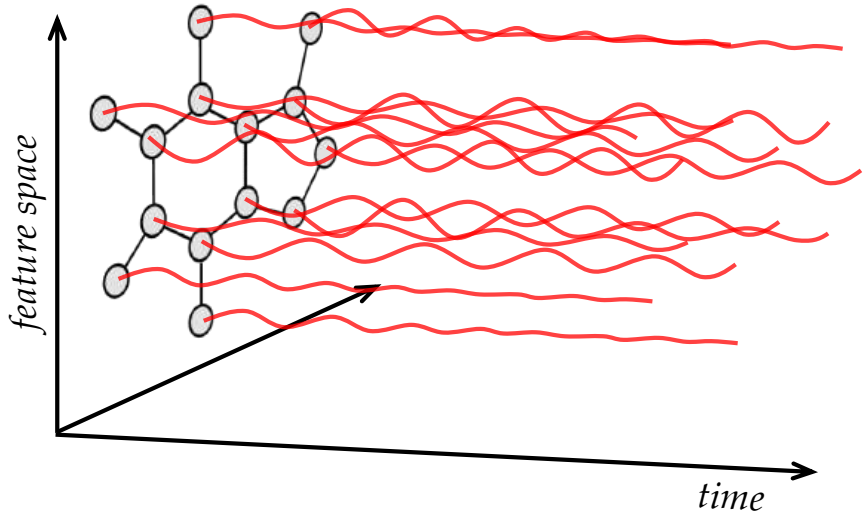
Continuous models for GNNs?

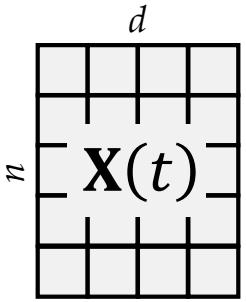


Physics-inspired GNNs

Physical metaphor of Graph ML

- GNN = dynamic system

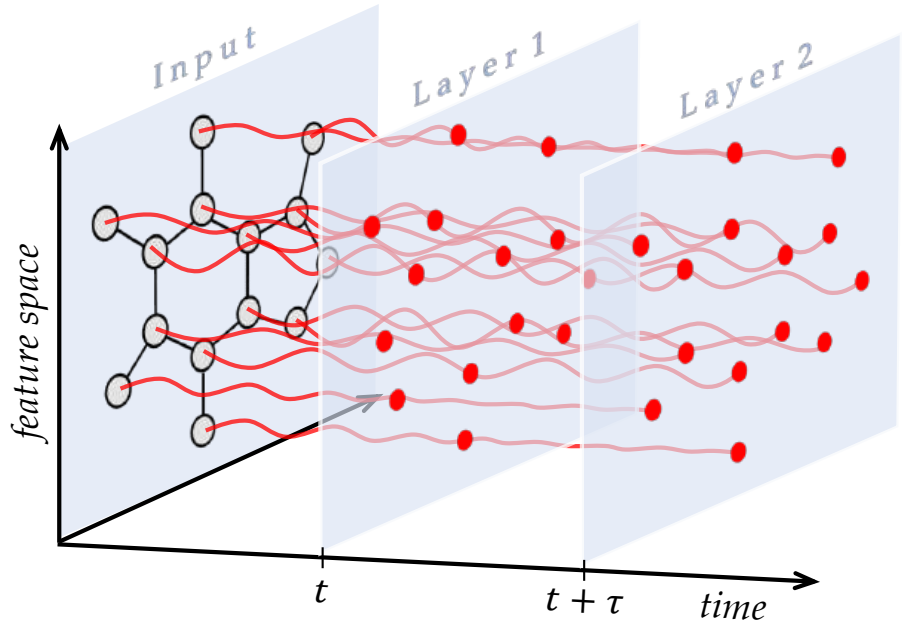




$$\dot{\mathbf{X}}(t) = \mathbf{F}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

Haber, Ruthotto 2017; Chen et al. 2019 (Neural ODEs); Xhonneau et al. 2020 (CGNN); Chamberlain et B. 2021 (GRAND, BLEND); Eliasof, Haber 2021 (PDE-GCN); Di Giovanni, Rowbottom et B 2022 (GRAFF), Rusch et B 2022 (GraphCON); Wu et B, Yan 2023; Eliasof et al. 2023; 2024 (TDE-GNN)

Physical metaphor of Graph ML



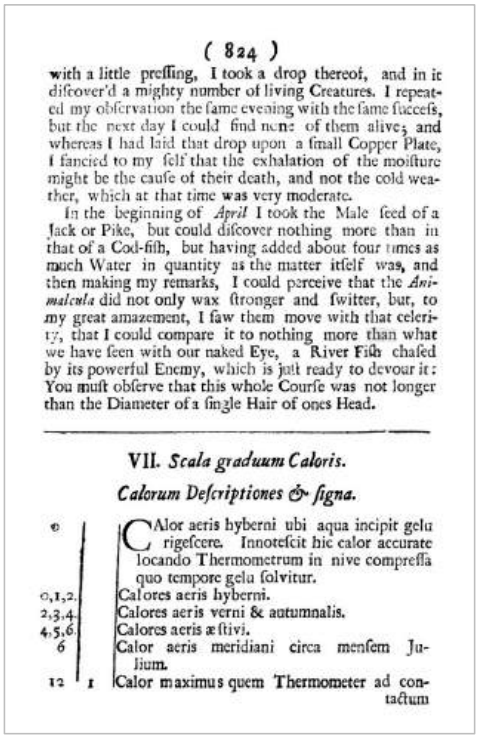
$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \tau \mathbf{F}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

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

Haber, Ruthotto 2017; Chen et al. 2019 (Neural ODEs); Xhonneau et al. 2020 (CGNN); Chamberlain et B. 2021 (GRAND, BLEND); Eliasof, Haber 2021 (PDE-GCN); Di Giovanni, Rowbottom et B 2022 (GRAFF), Rusch et B 2022 (GraphCON); Wu et B, Yan 2023; Eliasof et al. 2023; 2024 (TDE-GNN)

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.*

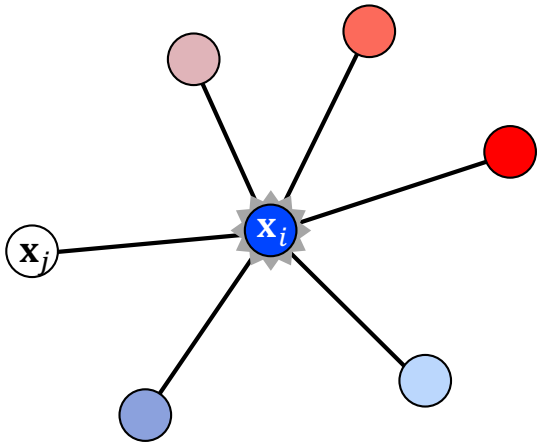
Calorum Deſcriptiones & ſigna.

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

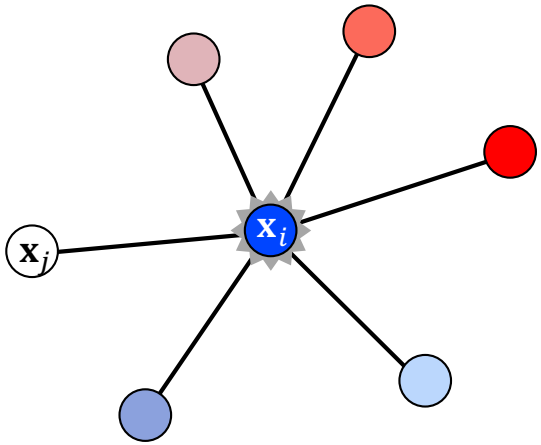
Anonymous 1701

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



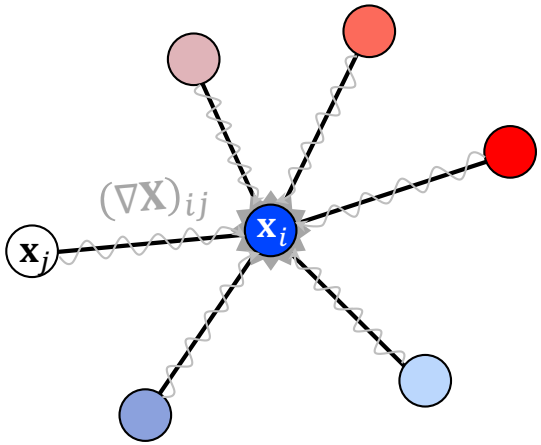
$$\dot{\mathbf{x}}_i(t) = \mathbf{x}_i(t) - \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} a_{ij} \mathbf{x}_j(t)$$

rate of temperature change

self temperature

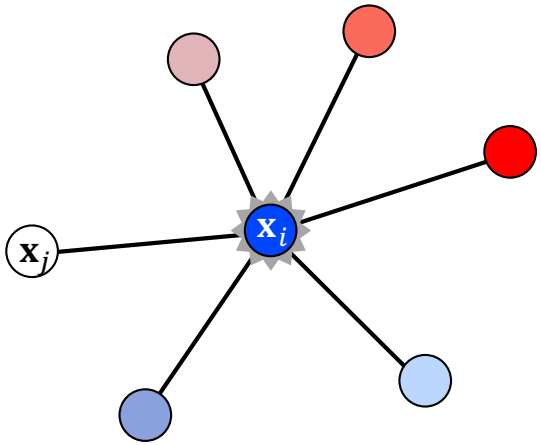
temperature of the environment

Heat Diffusion Equation on Graphs



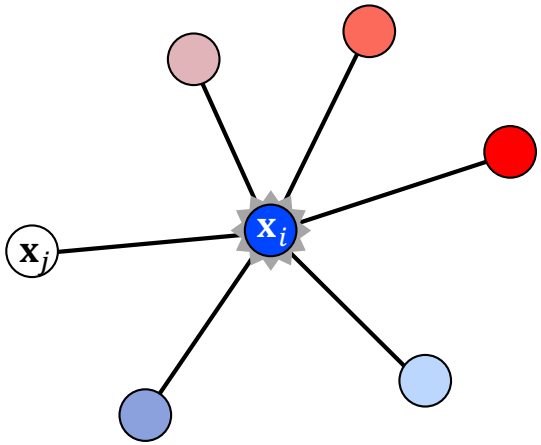
$$\dot{\mathbf{x}}_i(t) = \underbrace{\frac{1}{d_i}\sum_{j\in\mathcal{N}_i} a_{ij}}_{\text{divergence}\ \text{div}} \underbrace{\left(\mathbf{x}_i(t) - \mathbf{x}_j(t)\right)}_{\text{gradient}\ -(\nabla \mathbf{X})_{ij}}$$

Heat Diffusion Equation on Graphs



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

Heat Diffusion Equation on Graphs



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

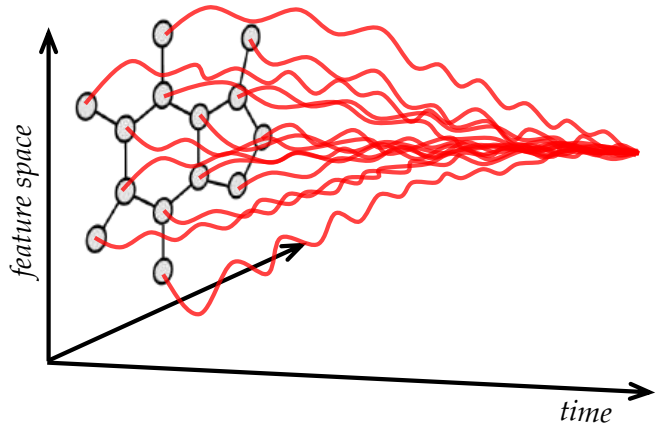
Heat Diffusion Equation as a prototypical Gradient Flow

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

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

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"



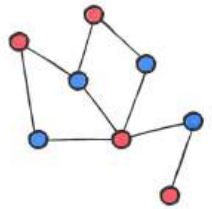
Heat Diffusion Equation as a prototypical Gradient Flow

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

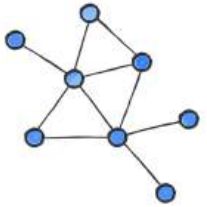
$$\mathcal{E}_{\mathrm{DIR}}(\mathbf{X}) = \frac{1}{2}\sum_{j \in \mathcal{N}_i} \left\|(\nabla \mathbf{X})_{ij}\right\|^2 = \frac{1}{2}\,\mathrm{trace}\big(\mathbf{X}^{\mathrm{T}} \Delta \mathbf{X}\big)$$

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")



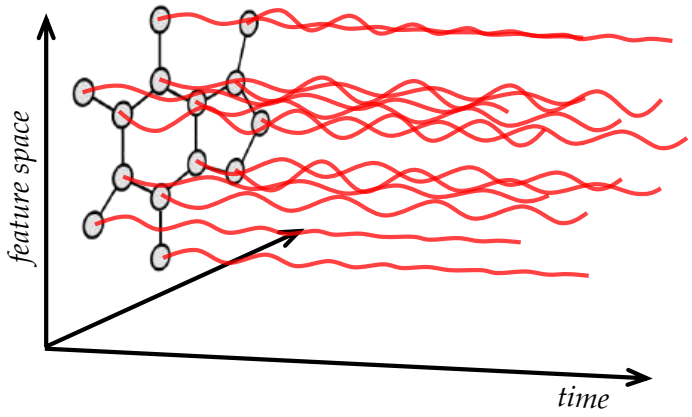
heterophilic



homophilic

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

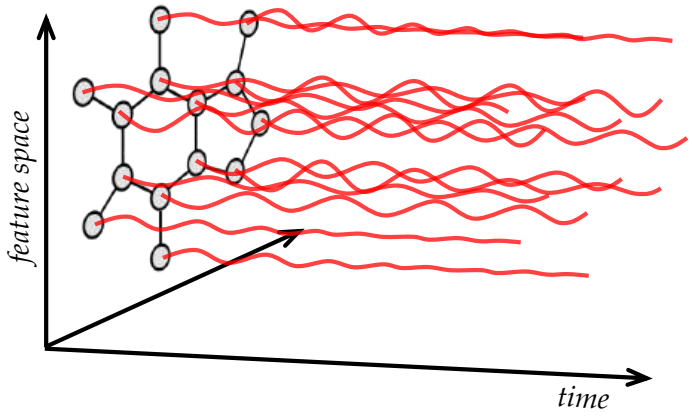
Gradient Flow Framework (GRAFF)



Traditional GNNs

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

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

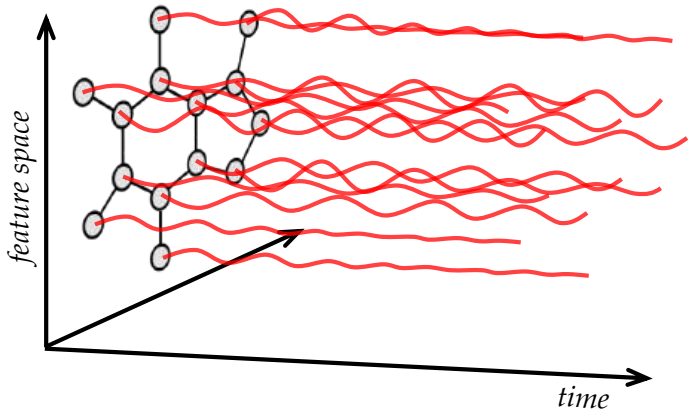
GRAFF

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

- Parametrize energy

Di Giovanni, Rowbottom et B 2022

Gradient Flow Framework (GRAFF)



Traditional GNNs

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

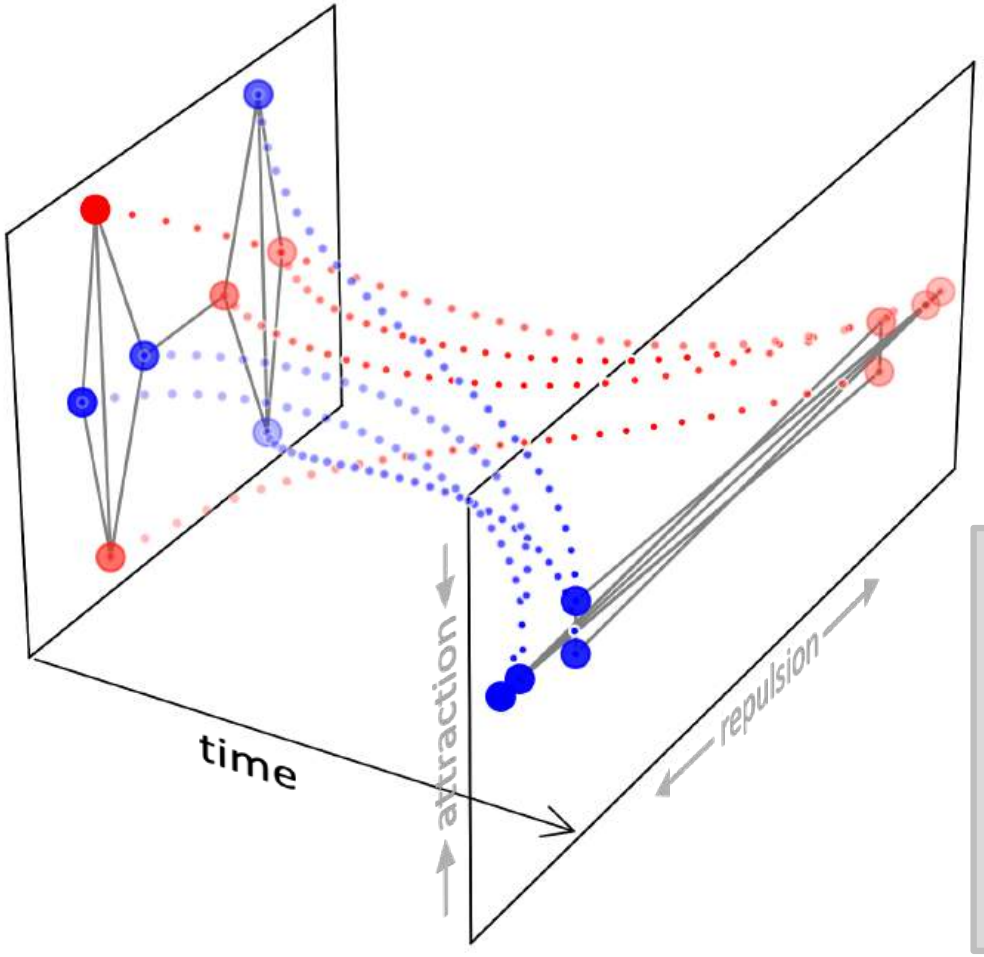
- Parametrize evolution equations

GRAFF

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

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

Di Giovanni, Rowbottom et B 2022



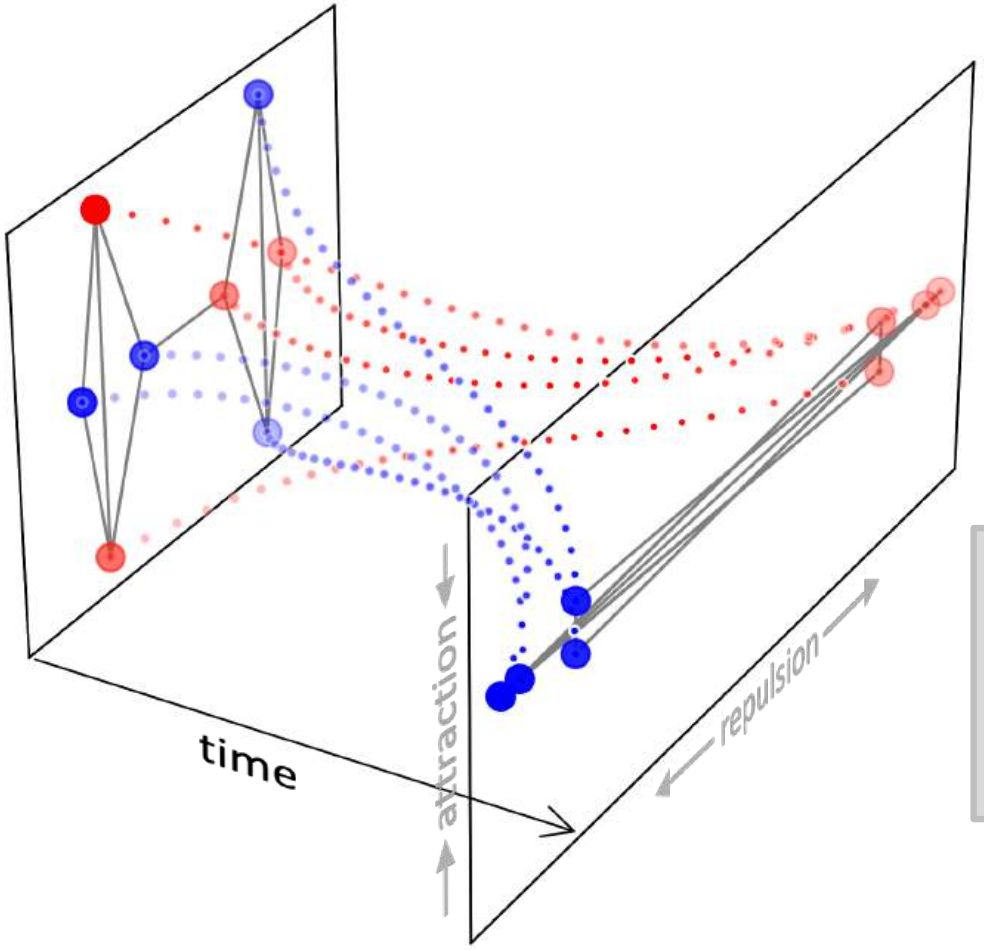
Di Giovanni, Rowbottom et B 2022

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

- **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

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

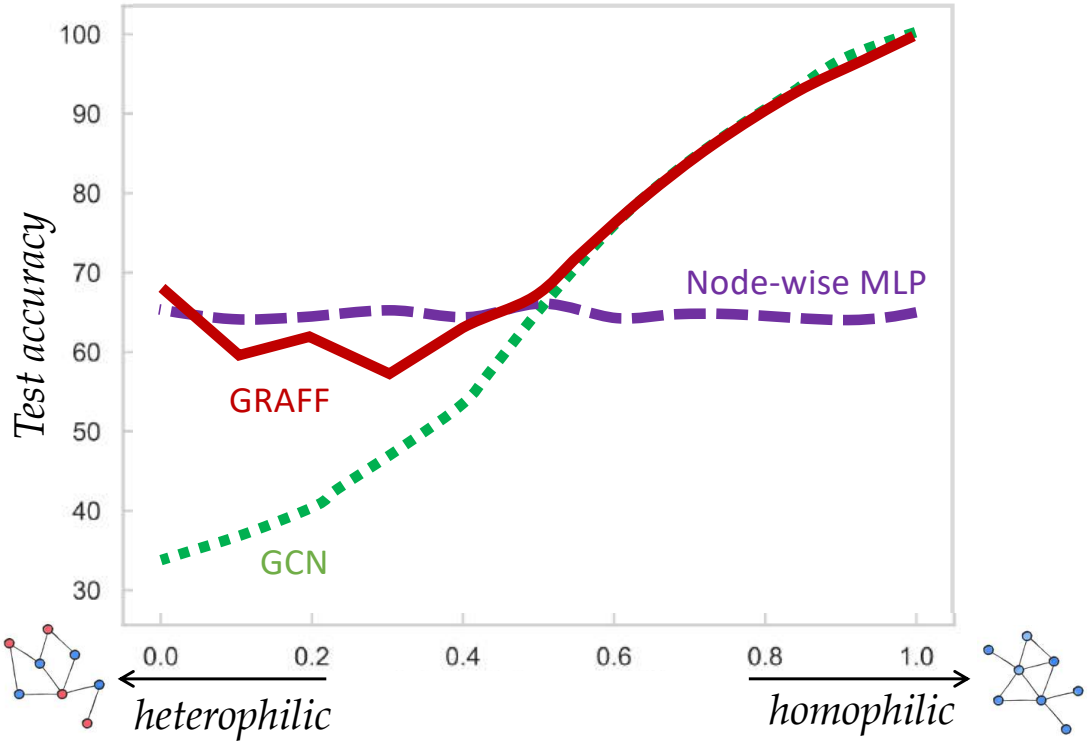
- **Attraction** along positive eigenvectors of **W**
- **Repulsion** along negative eigenvectors of **W**

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

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

Contradicts GNN "folklore"!

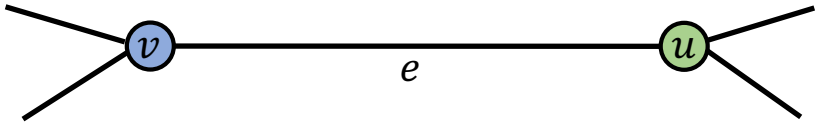
Homophily vs Heterophily



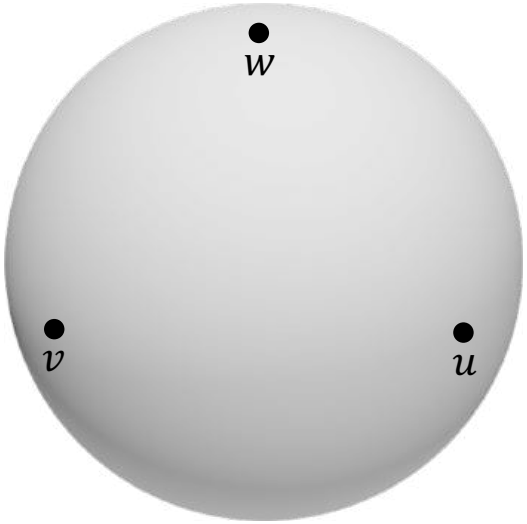
Synthetic Cora node classification task

Di Giovanni, Rowbottom et B 2022

Cellular Sheaves

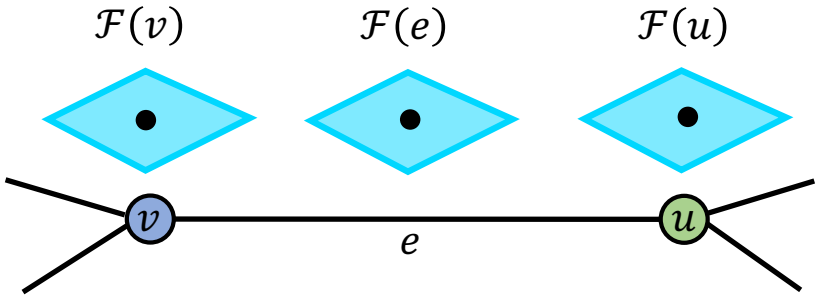


Graph

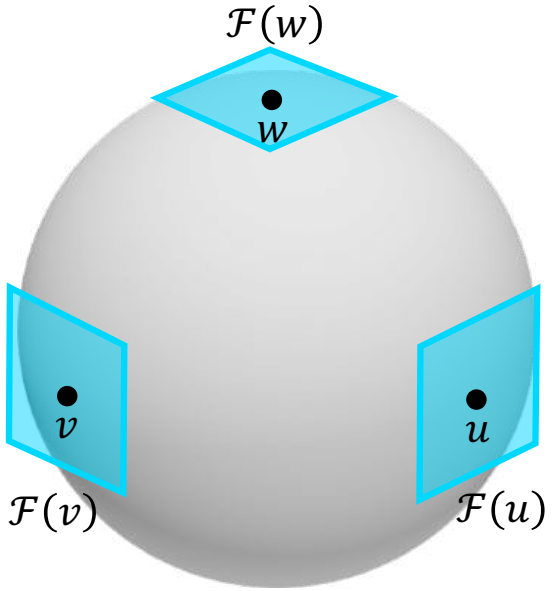


Manifold

Cellular Sheaves

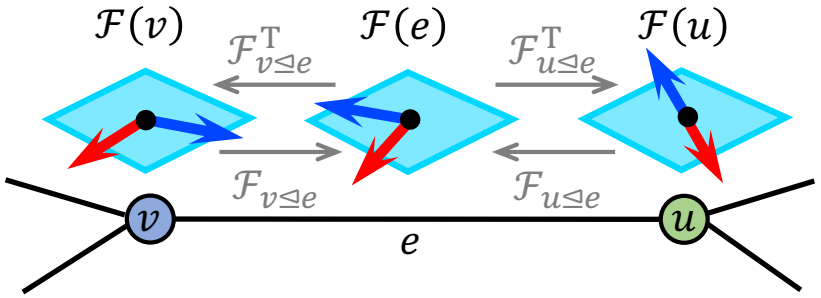


Cellular sheaf $\mathcal{F}$

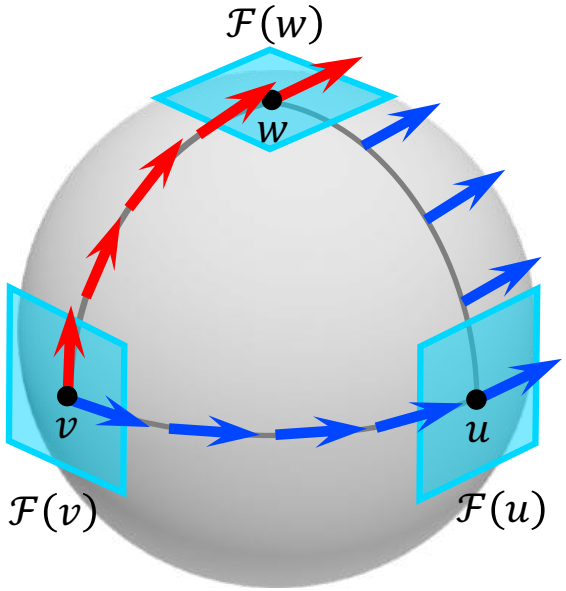


Manifold + Connection

Cellular Sheaves



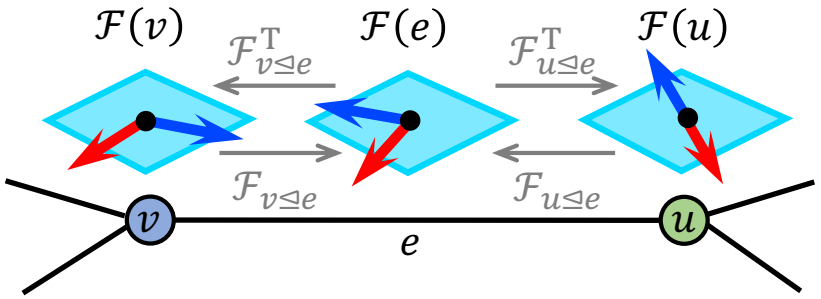
Cellular sheaf $\mathcal{F}$



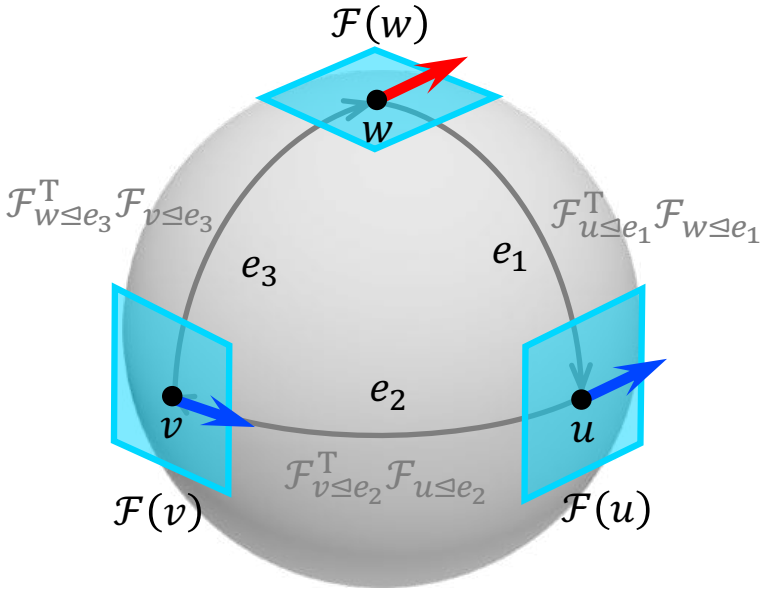
Manifold + Connection

Bodnar, di Giovanni, et B. 2022

Cellular Sheaves

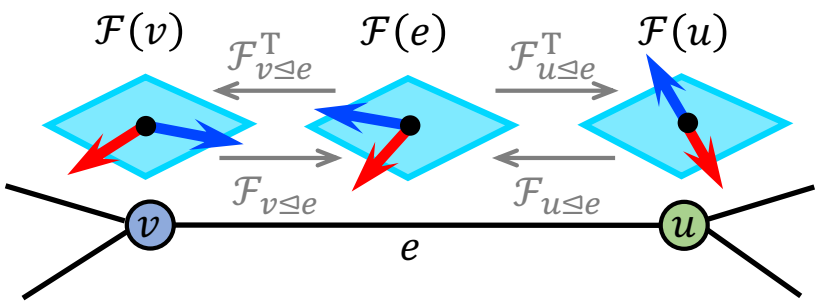


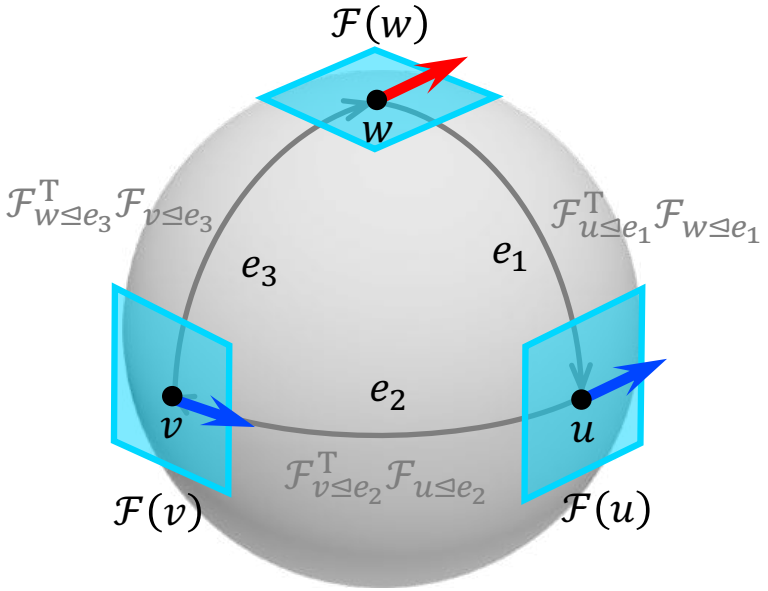
Cellular sheaf $\mathcal{F}$



Analogy to parallel transport on manifolds

Cellular Sheaves

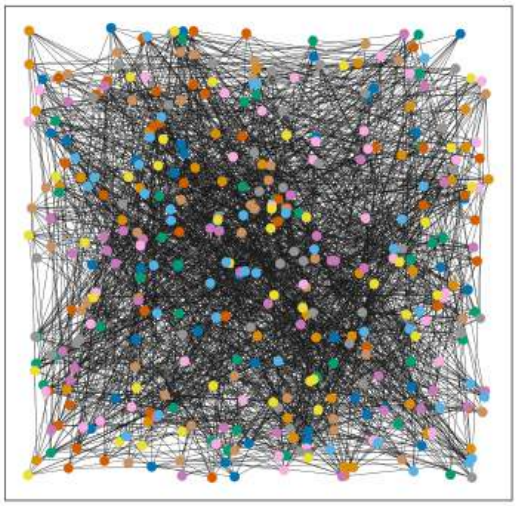




Endow graph with "geometry" leading to richer diffusion with better separation, ability to cope with heterophily, and no oversmoothing

Bodnar, di Giovanni, et B. 2022

Diffusion on Cellular Sheaves



Graph type	**# Classes**	**Sheaf class $\mathcal{F}$, dim=d**	
Homophilic	2	Symmetric d=1	✓
Heterophilic	2	Symmetric d=1	✗
	2	Non-symmetric d=1	✓
	≥3	Non-symmetric d=1	✗
	≤2d	Orthogonal, d={2,4}	✓

$$\dot{\mathbf{X}}(t) = \Delta_{\mathcal{F}}\mathbf{X}(t)$$

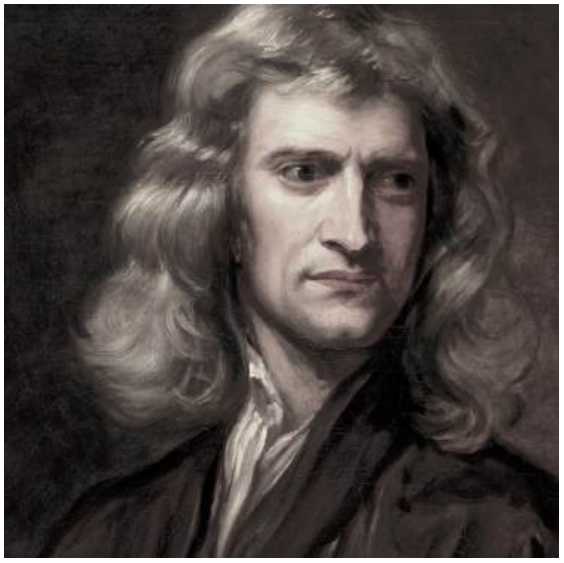
Node classification = limit of sheaf diffusion equation with an appropriate sheaf class

Bodnar, di Giovanni, et B. 2022


GNN community
toy
GCN knife
simplex
cells
Ricci flows
sheaf
GNN community
toy
GCN knife
simplex
cells
sheaf
Michael Galkin

Homogeneous Diffusion in Image Processing

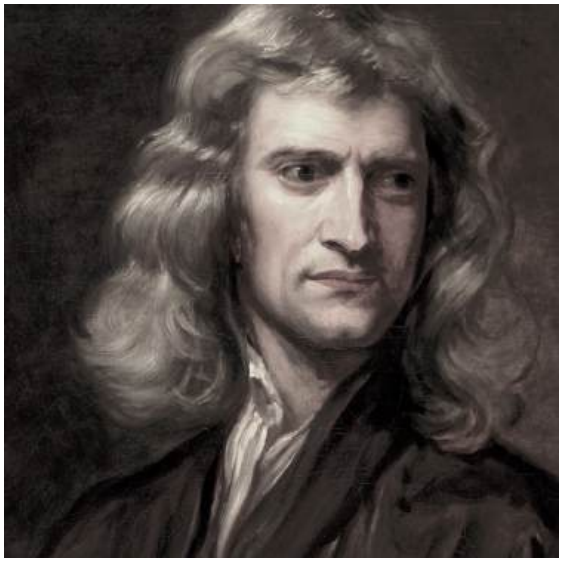
$$\dot{\mathbf{X}}(t) = -\mathrm{div}\big(c\nabla\mathbf{X}(t)\big)$$



$\mathbf{X}(0)$

Homogeneous Diffusion in Image Processing

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



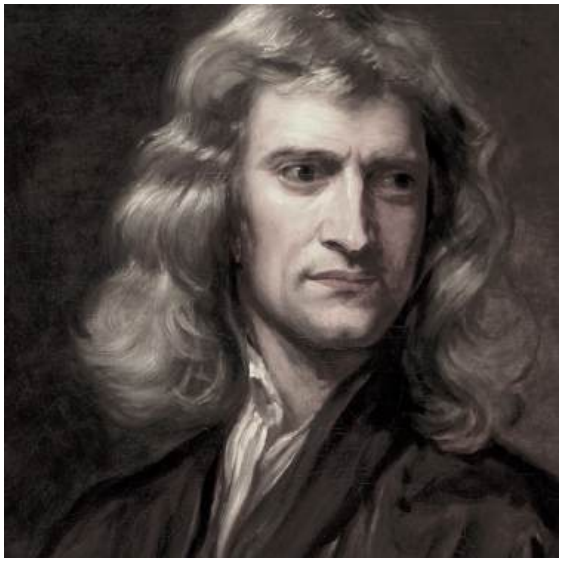
$\mathbf{X}(0)$

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

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"

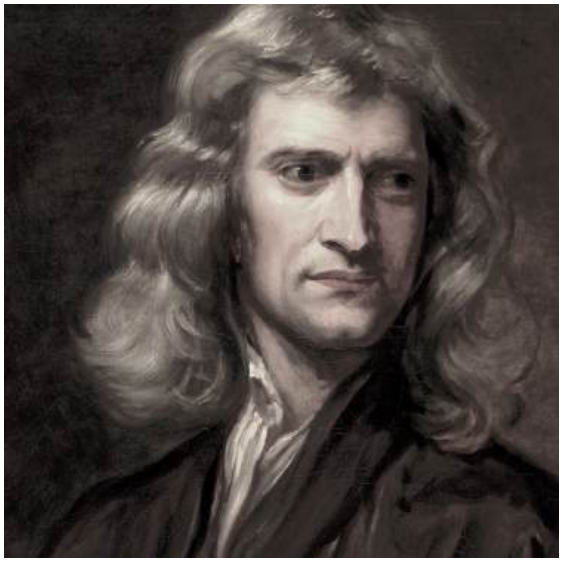
$\mathbf{X}(t) = \mathbf{X}(0) \star \mathbf{G}_{\sigma \propto t}$

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

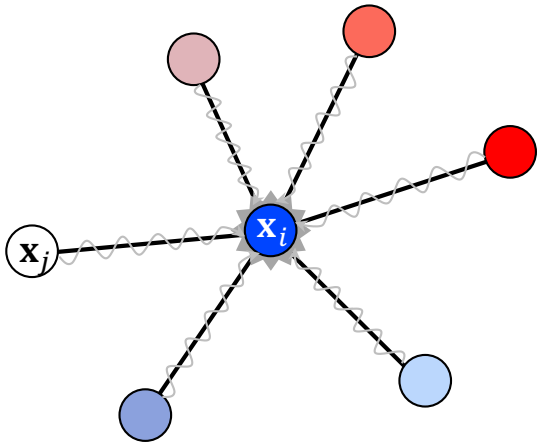


Non-homogeneous

Homogeneous

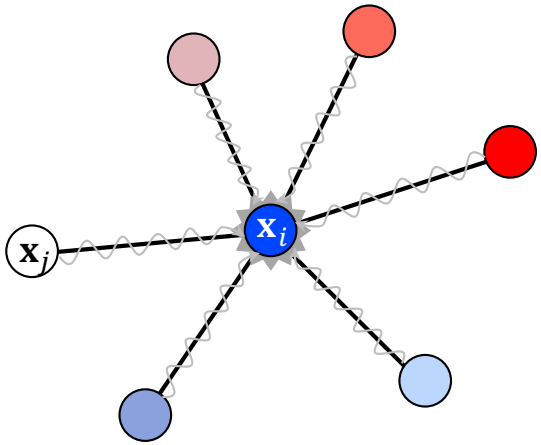
Perona, Malik 1990

Non-homogeneous Diffusion on Graphs



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

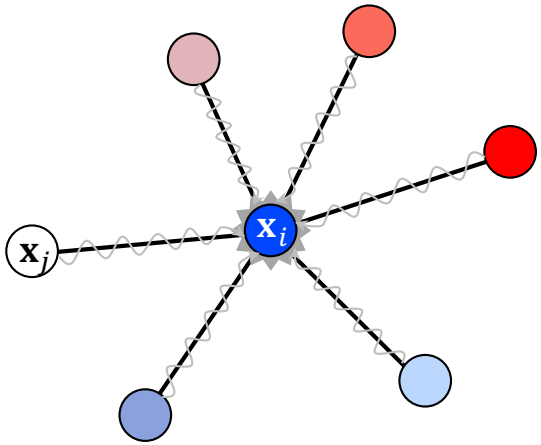
Non-homogeneous Diffusion on Graphs



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

learnable diffusivity

Non-homogeneous Diffusion on Graphs



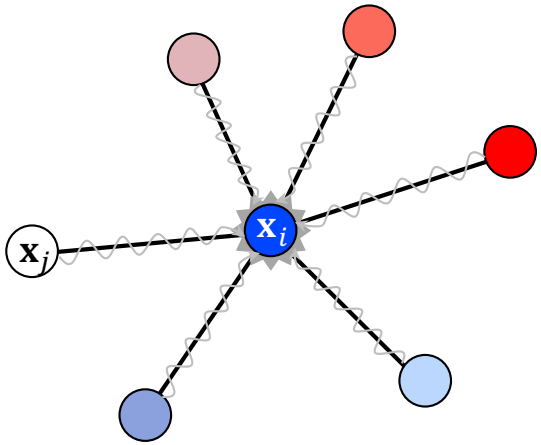
$$\mathbf{x}_i(t+\tau) = \mathbf{x}_i(t) + \tau \sum_{j \in \mathcal{N}_i} a\left(\mathbf{x}_i(t), \mathbf{x}_j(t)\right)\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)$$

Explicit (Forward Euler) discretization

time step

learnable diffusivity

Non-homogeneous Diffusion on Graphs



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

GAT!

normalised $\sum_j a_{ij} = 1$

unit step $\tau = 1$

Advective Diffusion Transformer

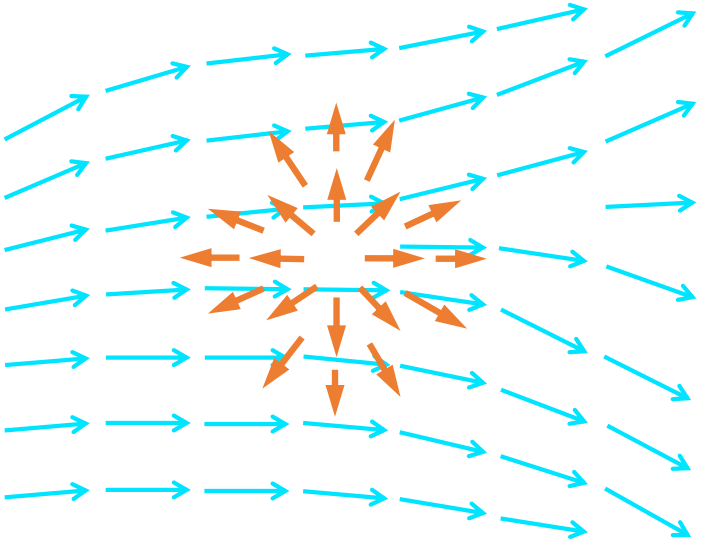
$$\dot{\mathbf{X}}(t) = \mathrm{div}\big(\mathbf{S}(t)\nabla\mathbf{X}(t)\big) + \beta\,\mathrm{div}\big(\mathbf{V}(t)\mathbf{X}(t)\big)$$

Wu et B, Yan 2023

Advective Diffusion Transformer

$$\dot{\mathbf{X}}(t) = \mathrm{div}\big(\mathbf{S}(t)\nabla\mathbf{X}(t)\big) + \beta\,\mathrm{div}\big(\mathbf{V}(t)\mathbf{X}(t)\big)$$

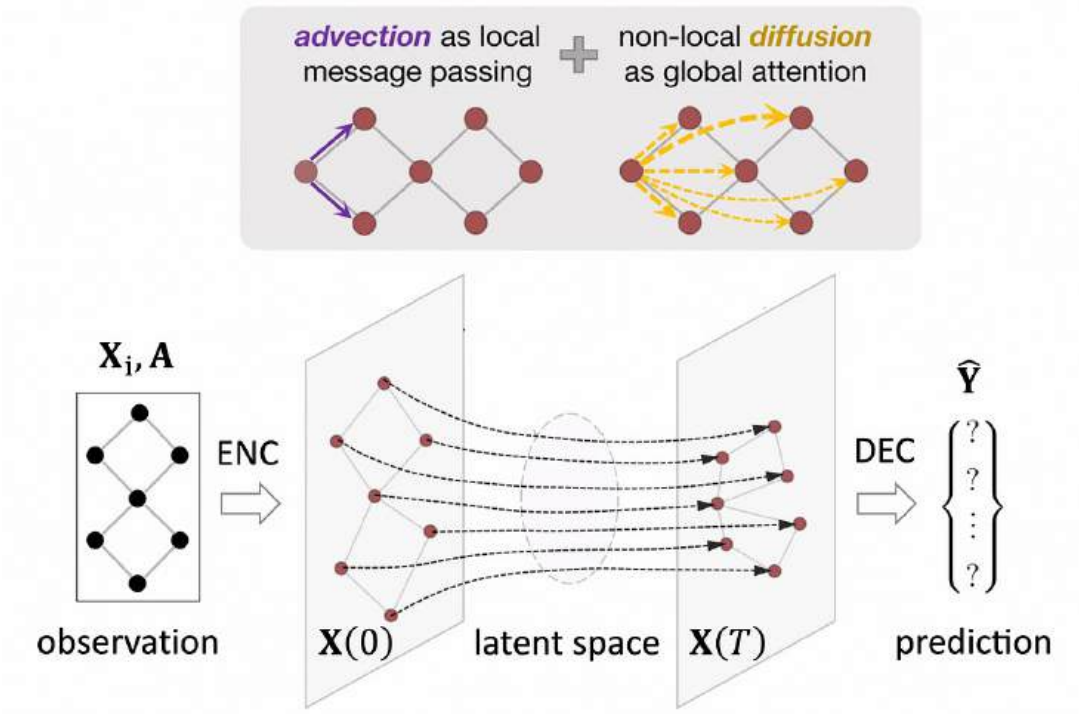
diffusion advection



Wu et B, Yan 2023

Advective Diffusion Transformer

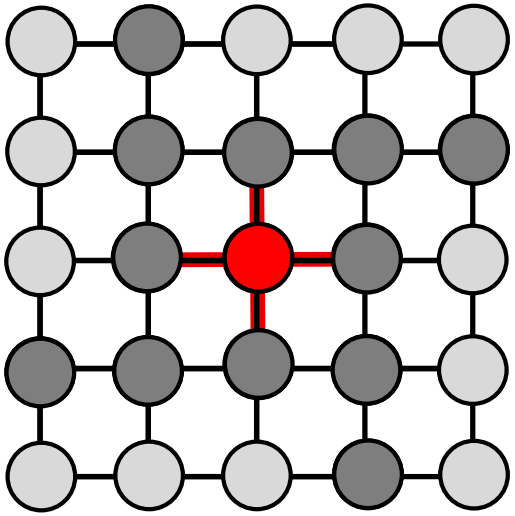
$$\dot{\mathbf{X}}(t) = \mathrm{div}\big(\mathbf{S}(t)\nabla\mathbf{X}(t)\big) + \beta\,\mathrm{div}\big(\mathbf{V}(t)\mathbf{X}(t)\big)$$



Wu et B, Yan 2023

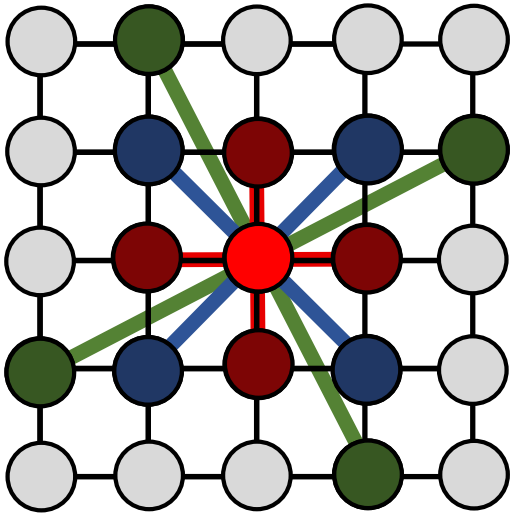
Spatial Derivative: Graph Rewiring?

$$\dot{x}(t) = \Delta x(t)$$



Spatial Derivative: Graph Rewiring?

$$\dot{x}(t) = \Delta x(t)$$

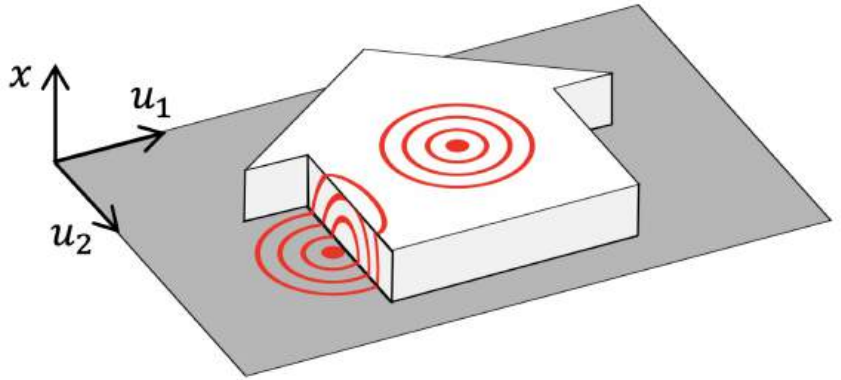


Different discretisations of 2D Laplacian

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

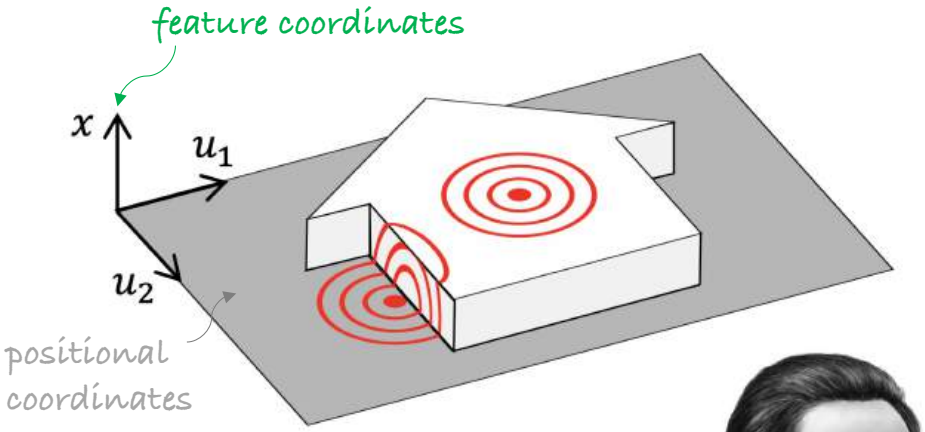
- Consider image as embedded 2-*manifold*

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

- *Pullback metric:* 2×2 matrix

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

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



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

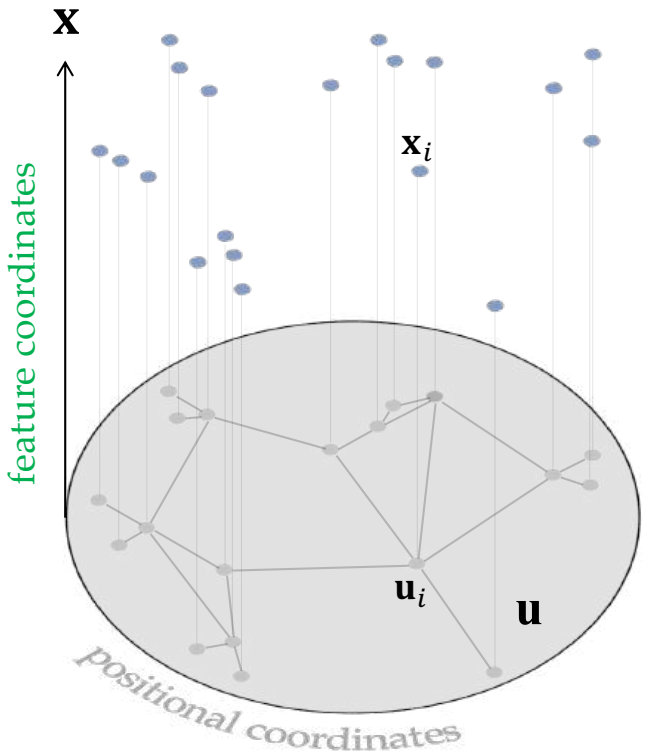
E. Beltrami

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**

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



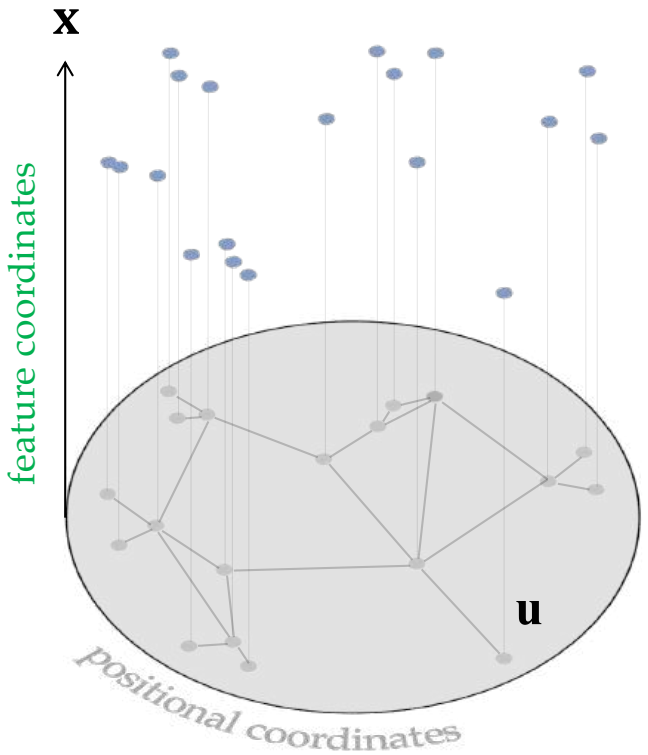
Chamberlain, Rowbottom, et B. 2021

Graph Beltrami flow

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

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

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



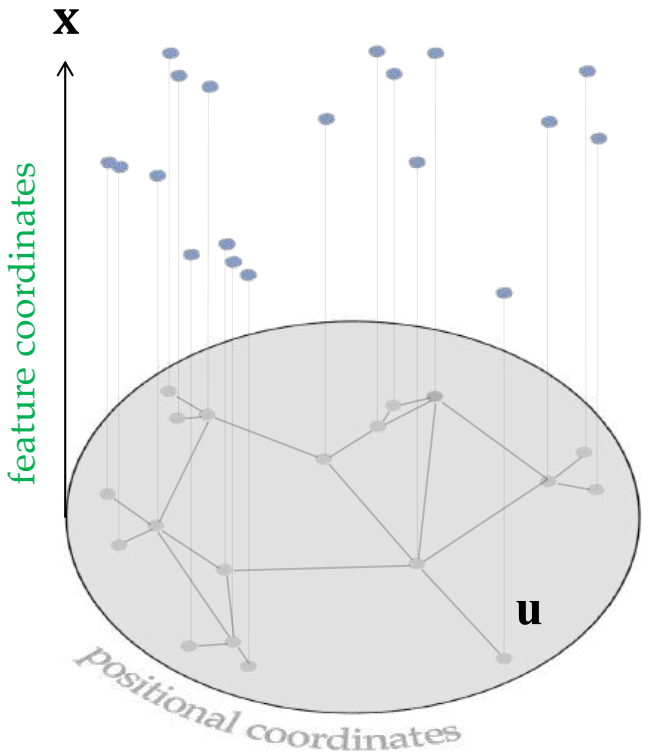
Chamberlain, Rowbottom, et B. 2021

Graph Beltrami flow

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

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

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



Chamberlain, Rowbottom, et B. 2021

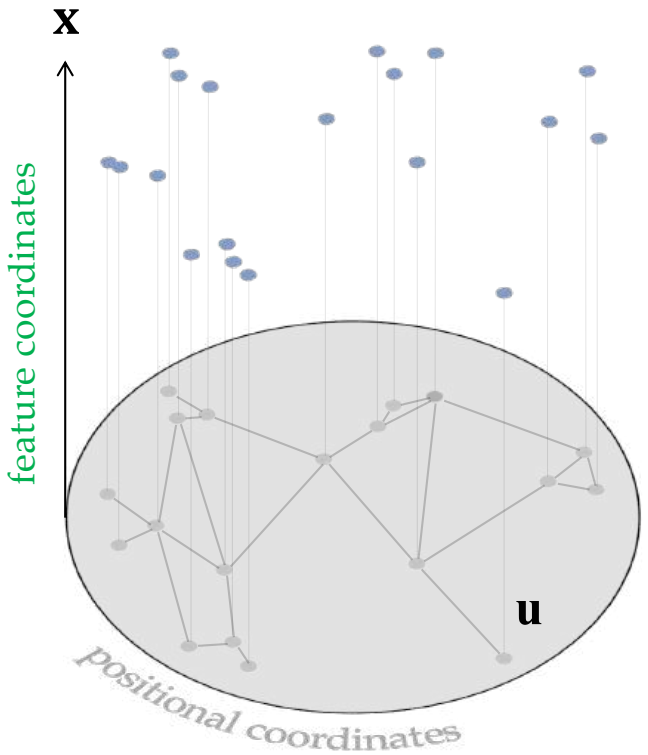
Graph Beltrami flow

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

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

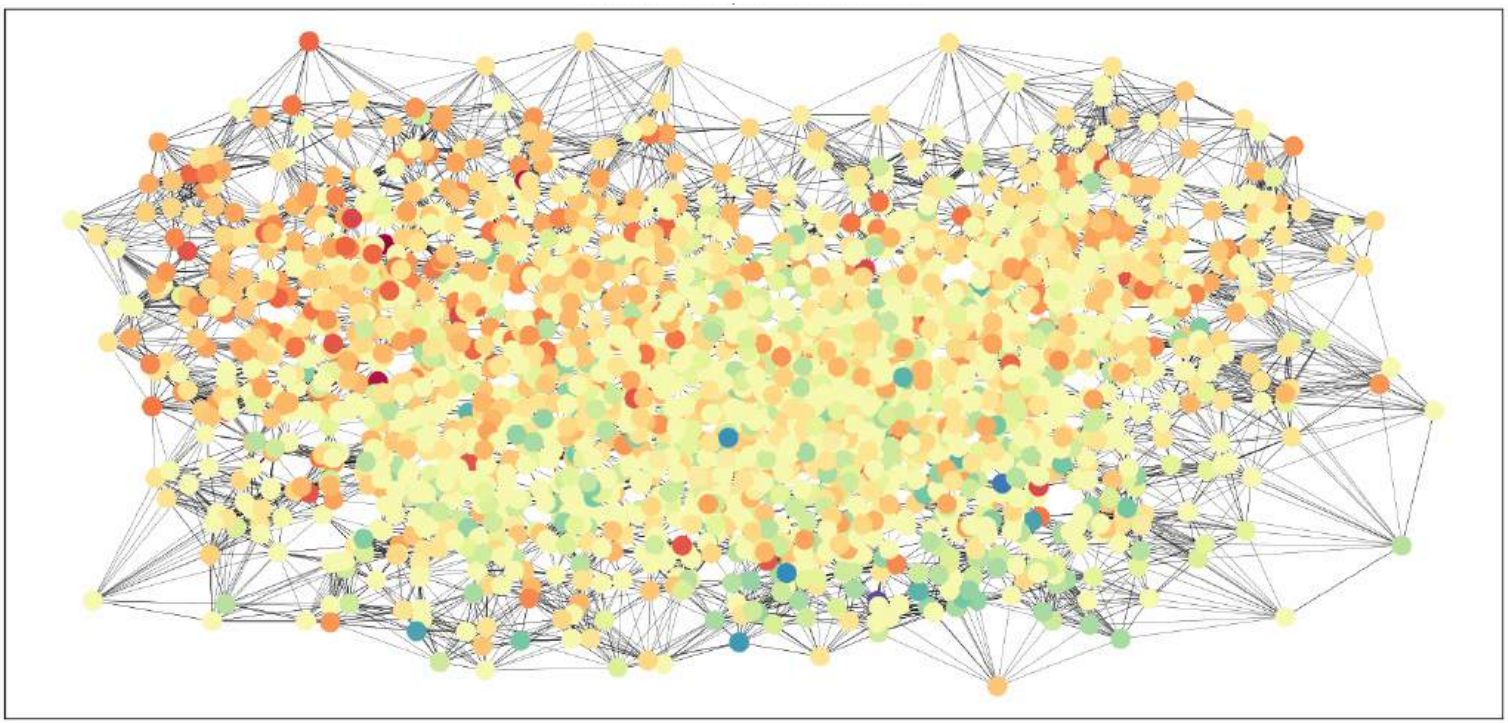
rewired graph

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



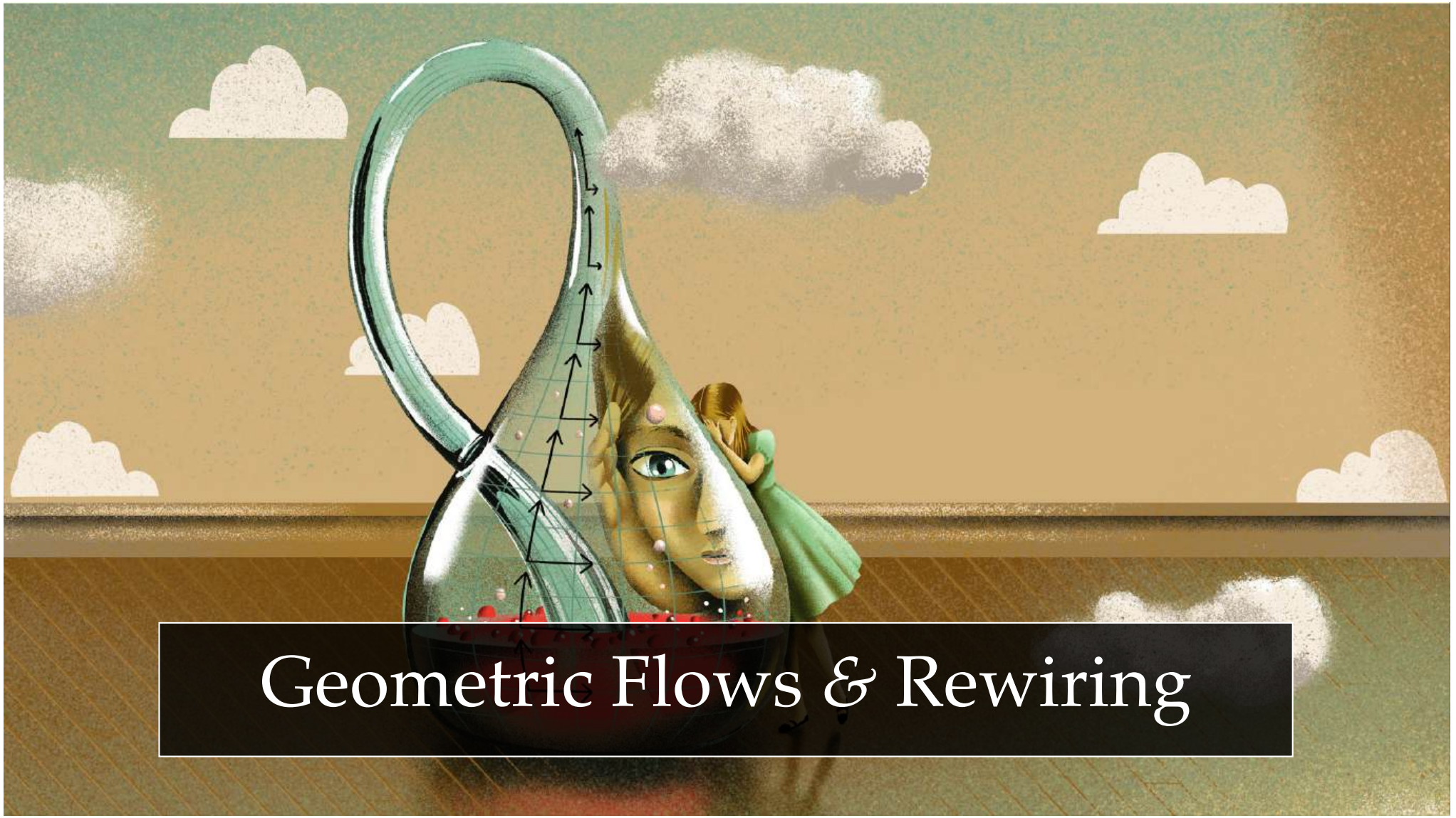
Chamberlain, Rowbottom, et B. 2021

Graph Beltrami flow



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

Chamberlain, Rowbottom, et B. 2021

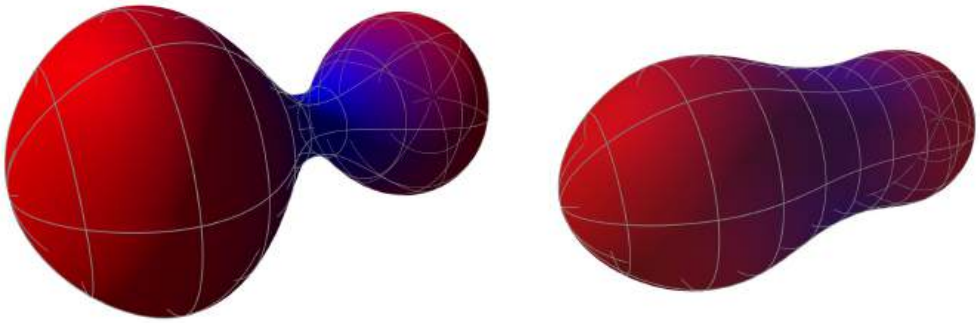


Geometric Flows & Rewiring

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;

G. Perelman

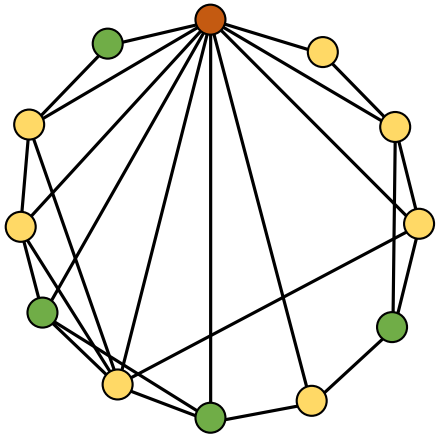
G. Ricci-Curbastro

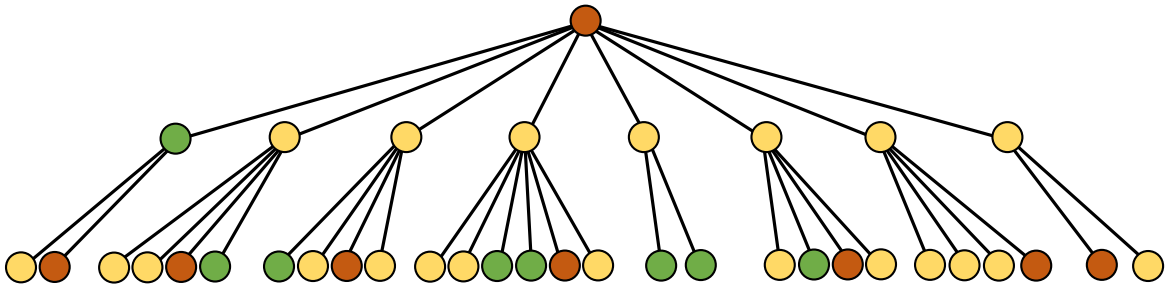
R. Hamilton

Ricci 1903; Hamilton 1988; Perelman 2003

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

Over-squashing & Bottlenecks





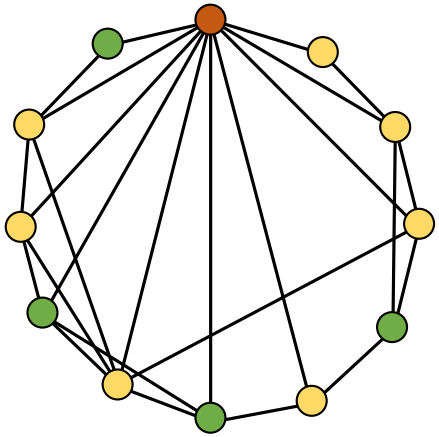
In some graphs metric ball volume grows exponentially with ball radius

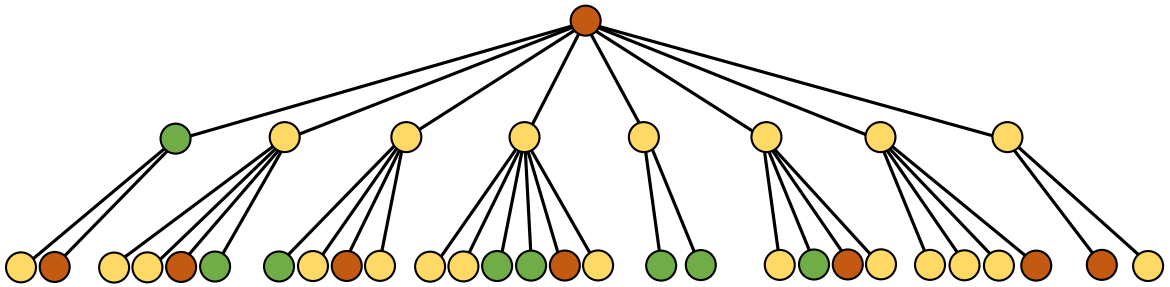
Over-squashing = Fast volume growth

+ 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

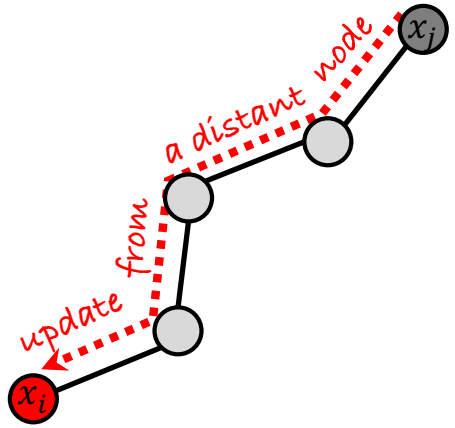
- Consider an MPNN of the form

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

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

Theorem (Sensitivity bound): For any i, j

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



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

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

Over-squashing

- Consider an MPNN of the form

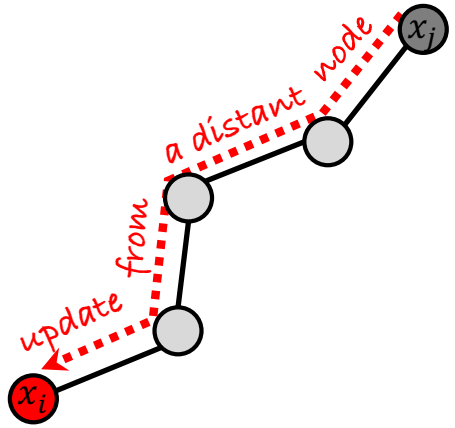
$$\mathbf{x}_i^{(k+1)} = \sigma\left(\mathbf{W}_1\mathbf{x}_i^{(k)} + \sum_j a_{ij}\,\mathbf{W}_2\mathbf{x}_j^{(k)}\right)$$

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

Theorem (Sensitivity bound): For any i, j

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

model topology



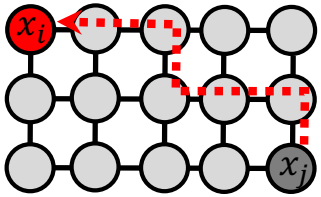
Over-squashing: small Jacobian $\left\|\partial \mathbf{x}_i^{(L)} / \partial \mathbf{x}_j^{(0)}\right\|$ indicates poor information propagation from input node

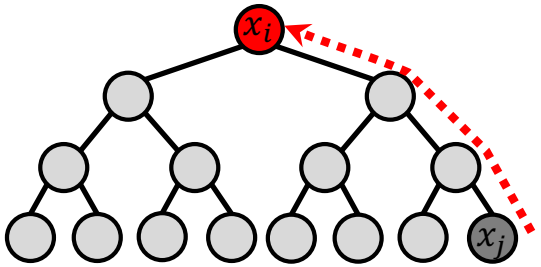
Topping, Di Giovanni et B 2021; Di Giovanni et B 2023

Preventing over-squashing

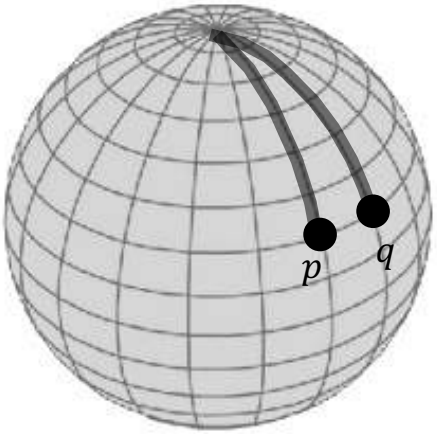
$$\left\| \frac{\partial \mathbf{x}_i^{(L)}}{\partial \mathbf{x}_j^{(0)}} \right\|_1 \leq (c_\sigma w p)^L (\mathbf{I} + \mathbf{A})^L_{ij}$$

model topology

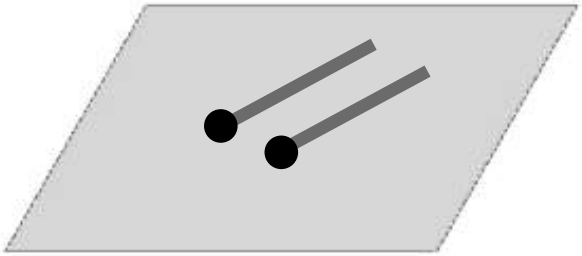




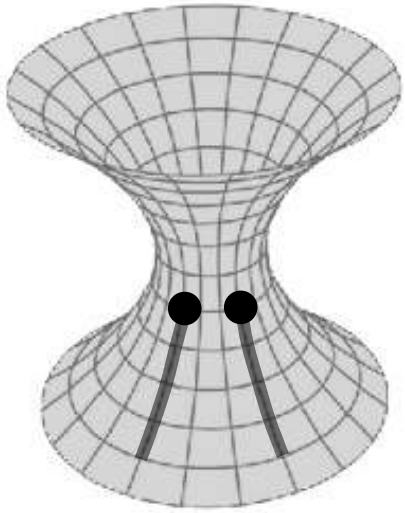
Ricci Curvature on Manifolds



Spherical (>0)



Euclidean (=0)

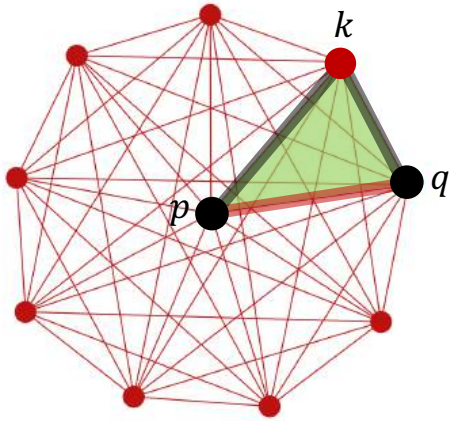


Hyperbolic (<0)

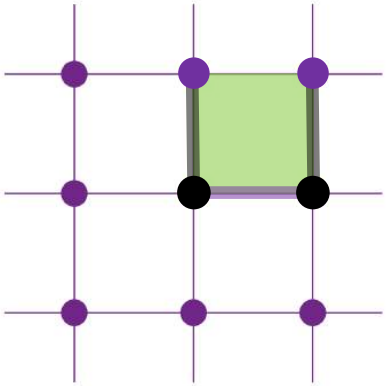
"geodesic dispersion"

Ricci 1903

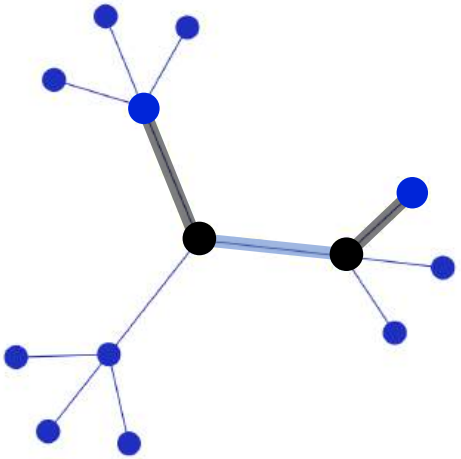
Discrete Ricci Curvature on Graphs



Clique (>0)



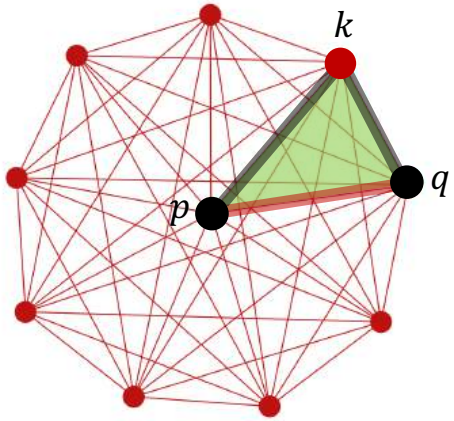
Grid (=0)



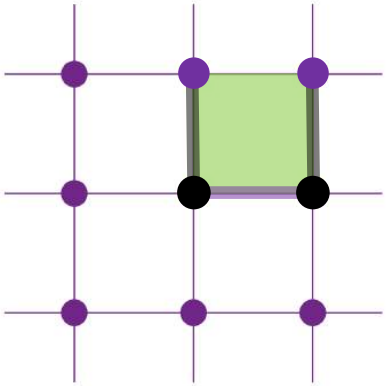
Tree (<0)

Topping, Di Giovanni et B 2021

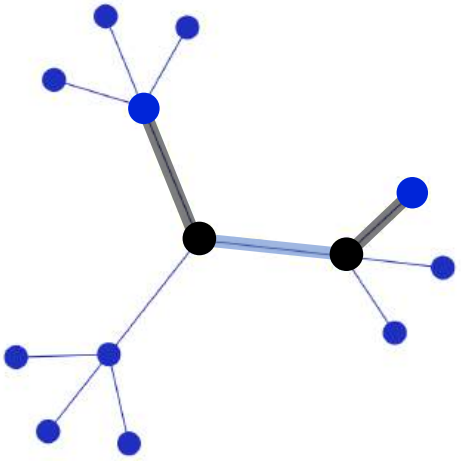
What contributes to over-squashing?



Clique ($\geqslant 0$)



Grid ($=0$)

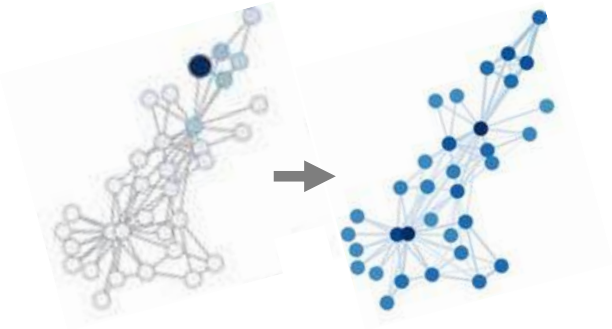


Tree ($\leqslant 0$)

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

Topping, Di Giovanni et B 2021

Graph Rewiring Approaches



Connectivity Diffusion
(DIGL)

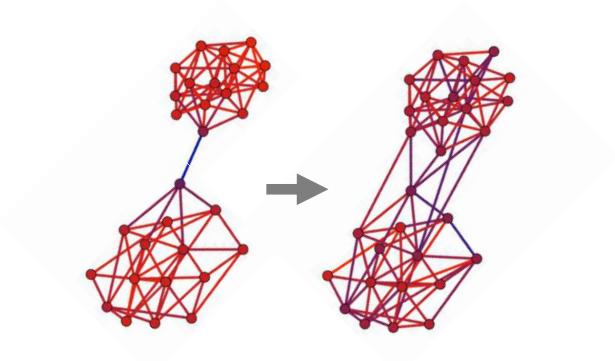
Gasteiger et al. 2019

- KNN graph on PPR embedding

+ Good for homophilic graphs

– Bad for heterophilic graphs

– Drastically different graph



Discrete Ricci flow
(SDRF)

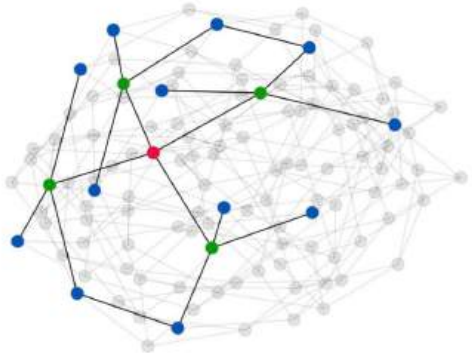
Topping, Di Giovanni et B. 2022

- Remove negatively-curved edges

+ Good for both homophilic and heterophilic graphs

+ "Surgical" rewiring

– Curvature is computationally expensive



Expander Propagation
(EGP)

Deac et al. 2022

- Fixed expander graph
- Alternate MP on original and new graph

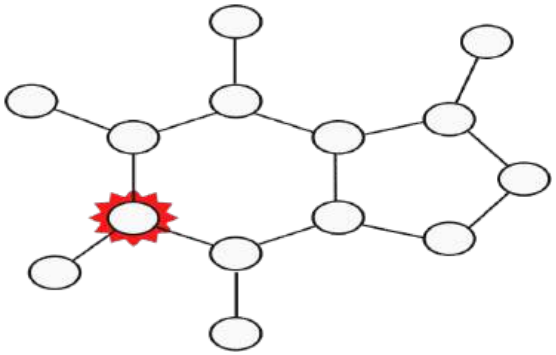
+ Optimal graph for MP

– No relation to input graph

– No permutation equivariance

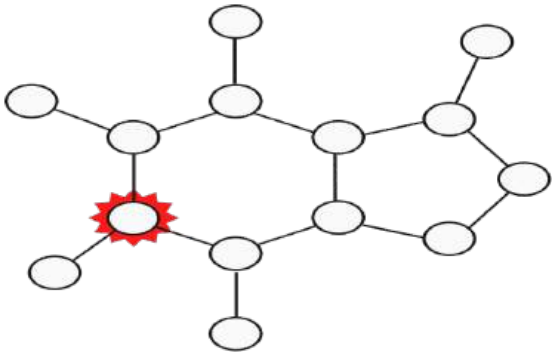
No relation to the task!

Why it is important to consider the task?



Van der Waals interactions

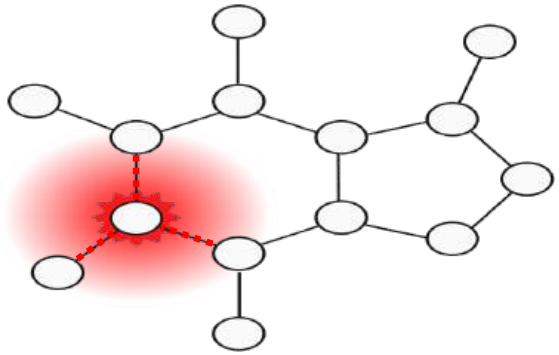
$\propto r^{-12}$



Coulomb interactions

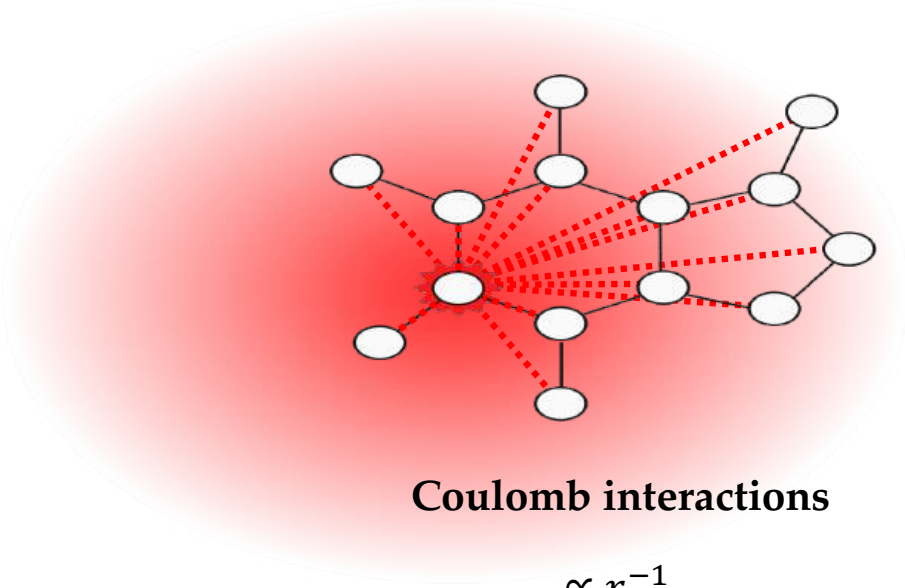
$\propto r^{-1}$

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!

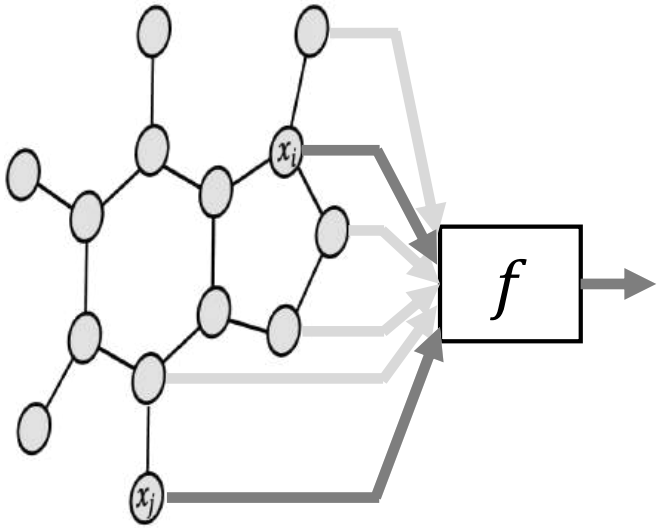
Long-range interactions & Expressivity

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

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 $\mathrm{mix}_f(i,j) = 0$

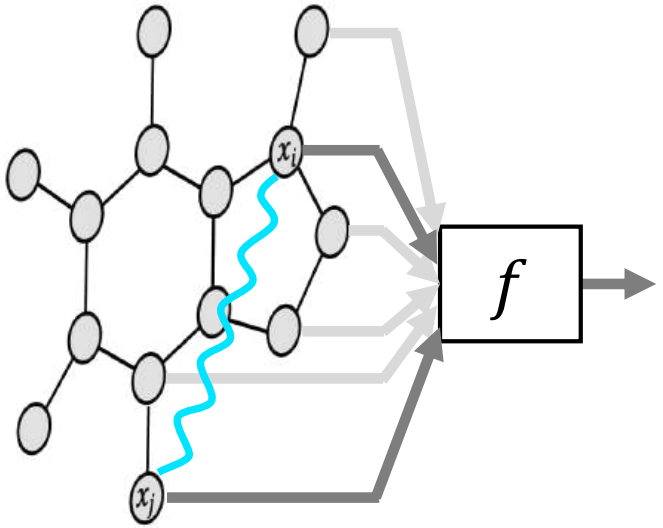


Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

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

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



- $f(\mathbf{X}) = \phi(\mathbf{x}_i) + \phi(\mathbf{x}_j)$ 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 ϕ is

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

Capacity bounds

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

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

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

Capacity bounds

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

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

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

Bound on weights w

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

- $d_{\min}$ =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

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

- d_i =degree of node i
- α, β =model-related constants
- $|E|$ =number of edges
- Bounded weights

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

Capacity bounds

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

Bound on weights w

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

"weights need to be large enough to allow mixing"

Bound on depth L

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

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

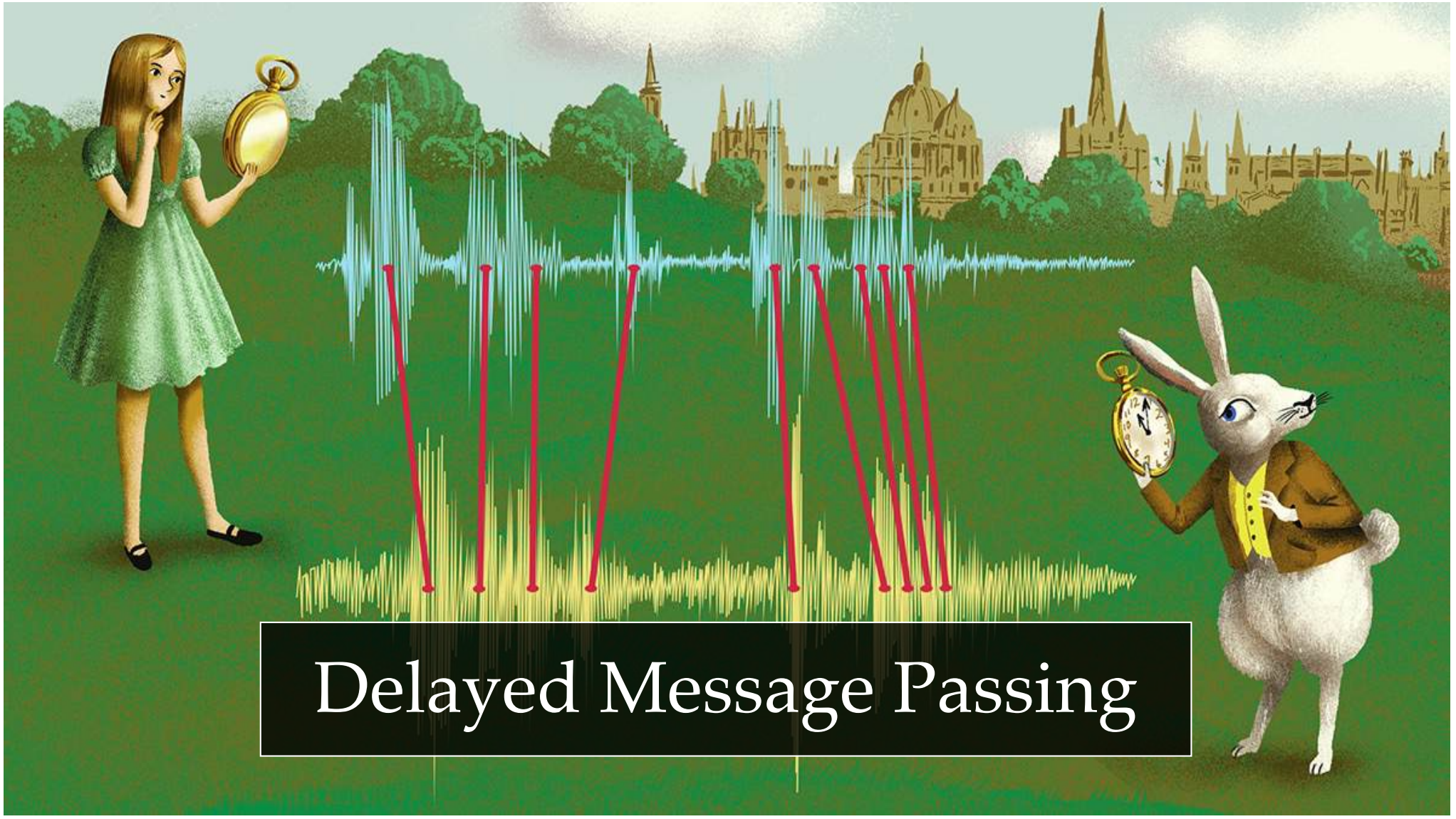
Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

Expressive power without Weisfeiler-Lehman

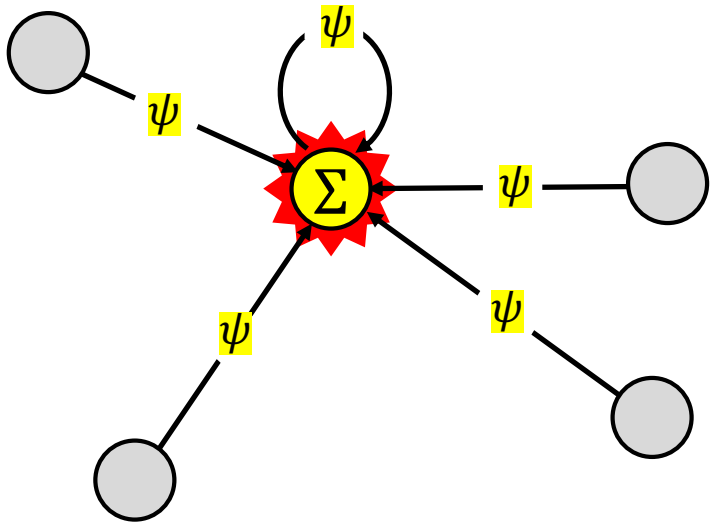
Expressive power (informal): MPNN with $L \leq n$ layers *cannot learn* tasks that require high mixing among features at nodes with large commute time.

A. Lehman **B. Weisfeiler**

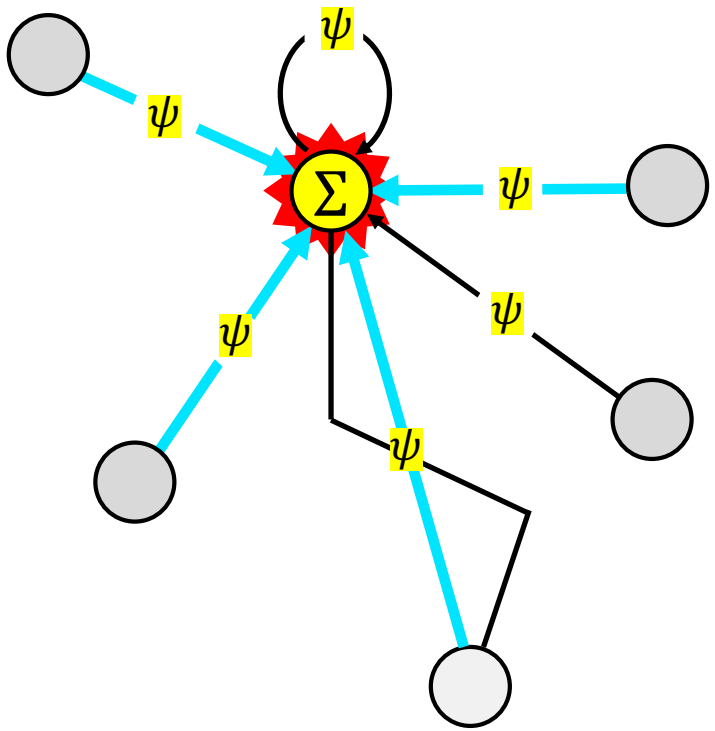
Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023


Delayed Message Passing

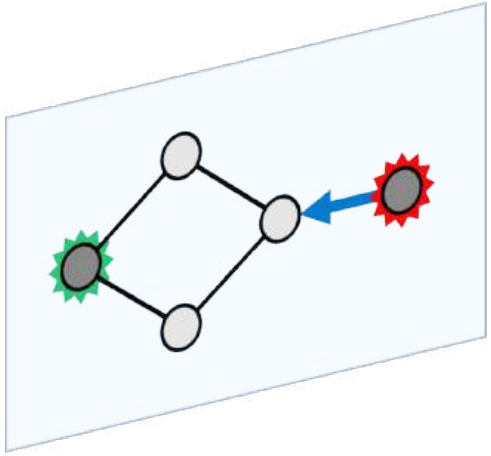
What



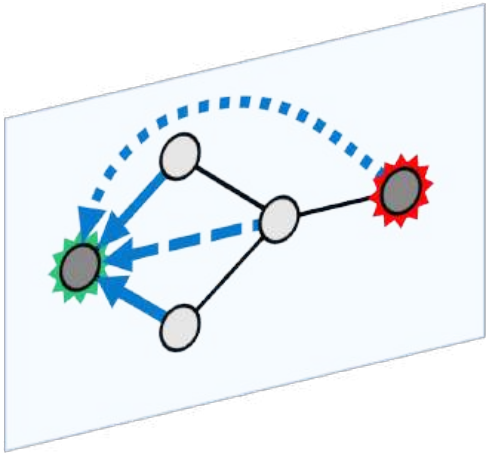
What + Where



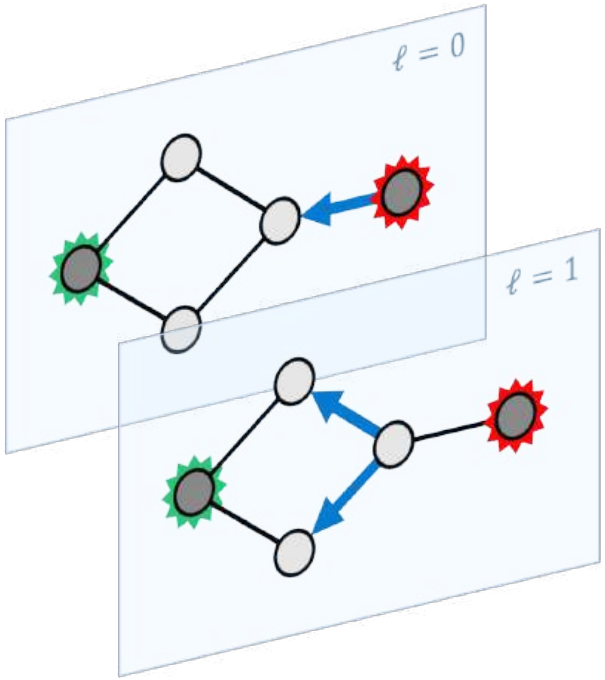
What + Where + When



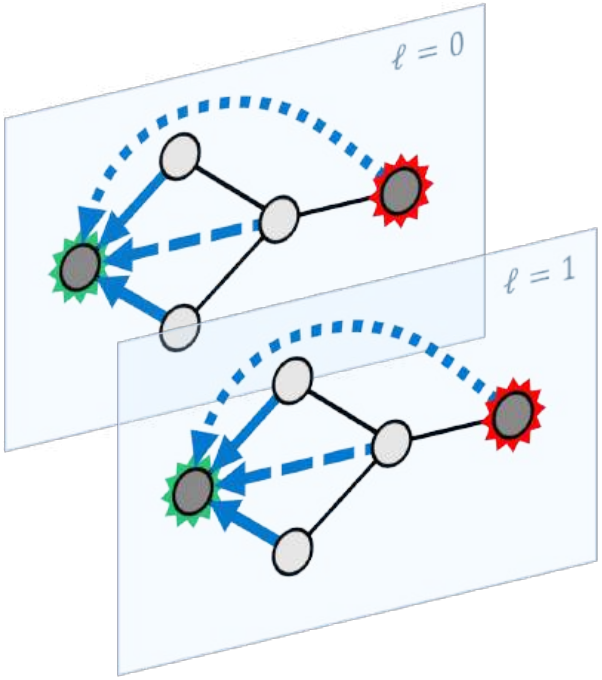
Classical MPNN



Graph Transformer

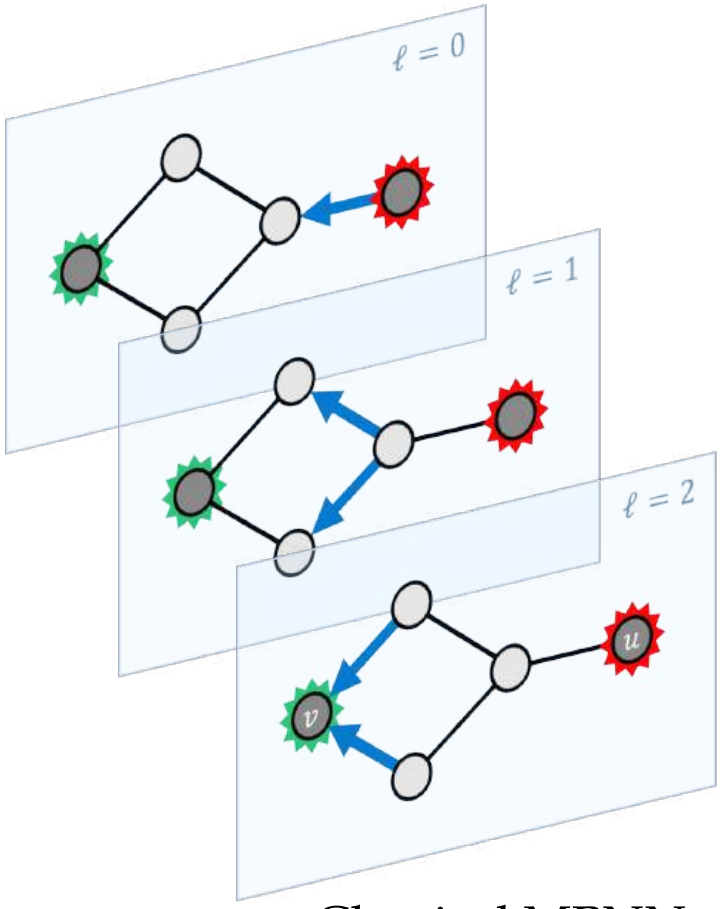


Classical MPNN

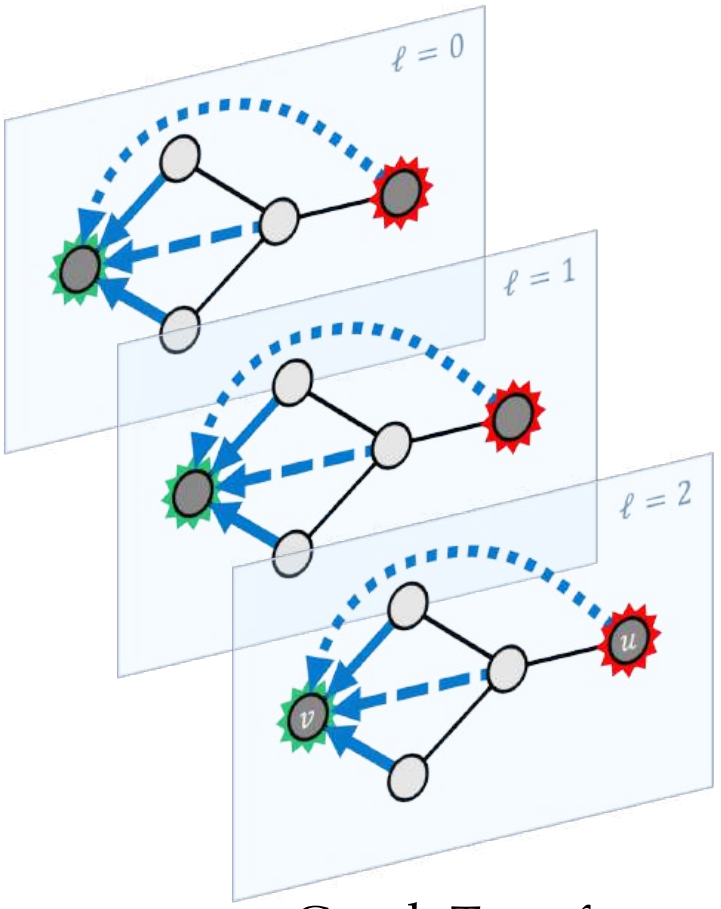


Graph Transformer

Gutteridge, Di Giovanni et B 2023

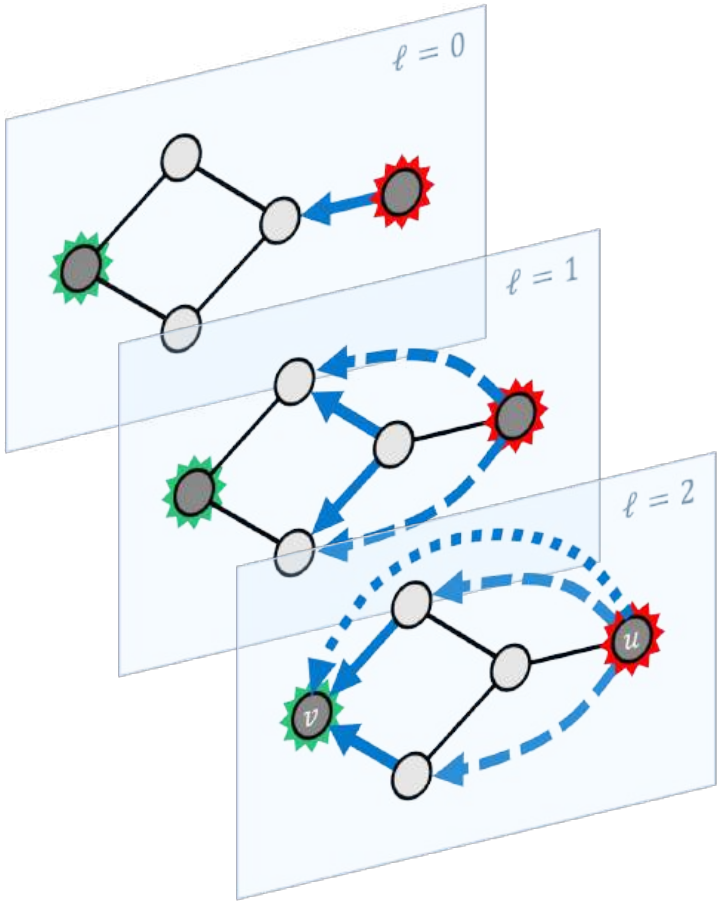


Classical MPNN

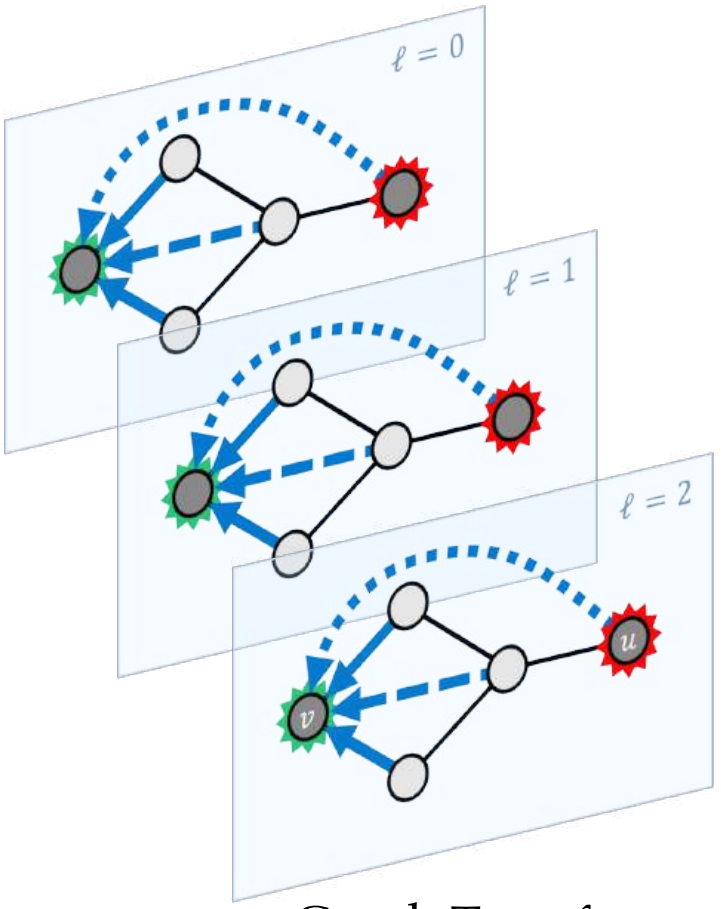


Graph Transformer

Gutteridge, Di Giovanni et B 2023

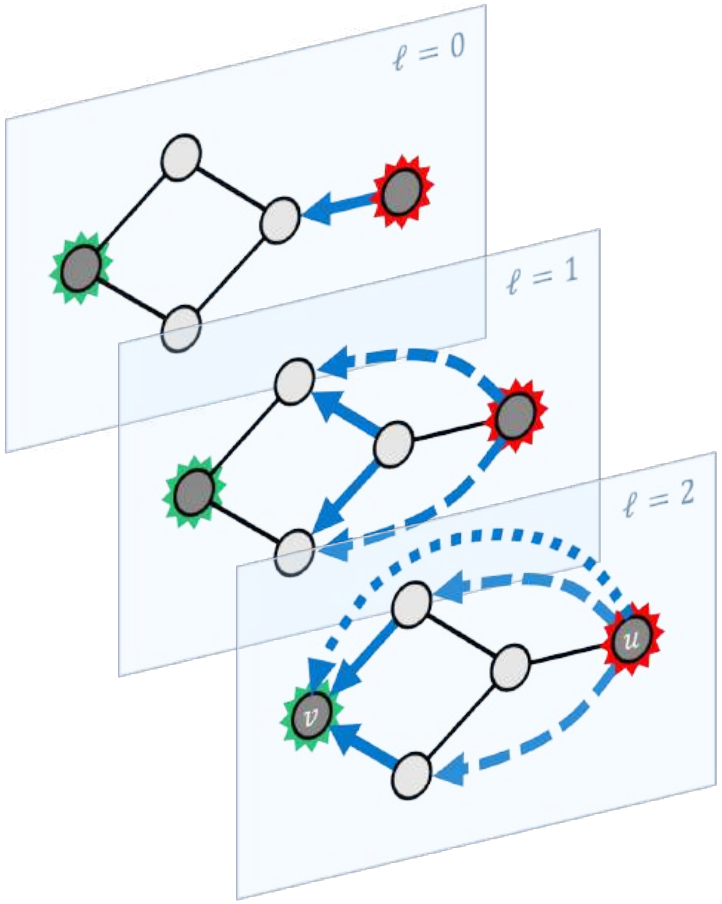


Dynamic Rewiring (DRew)

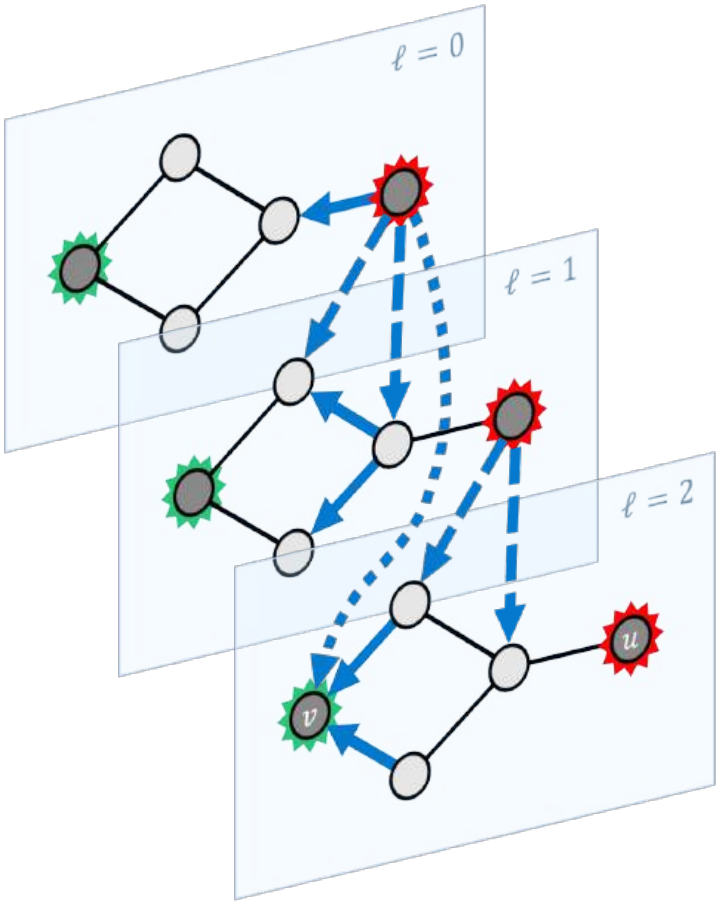


Graph Transformer

Gutteridge, Di Giovanni et B 2023



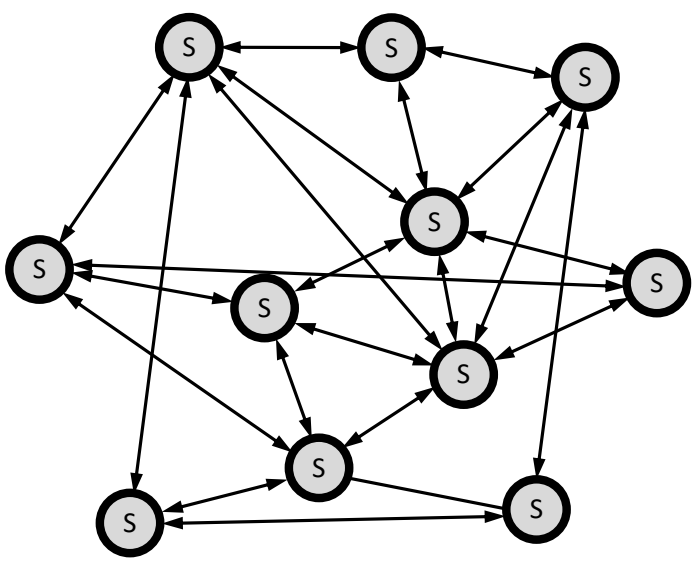
Dynamic Rewiring (DRew)



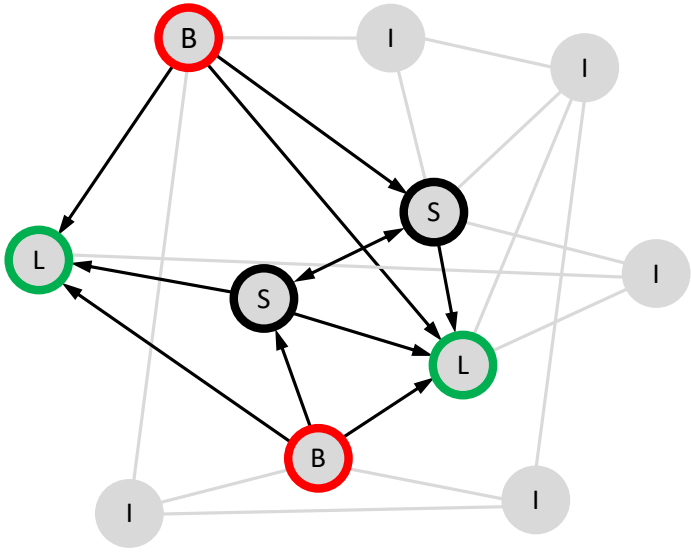
Dynamic Rewiring + delay (**v**DRew)

Gutteridge, Di Giovanni et B 2023

Cooperative Message Passing



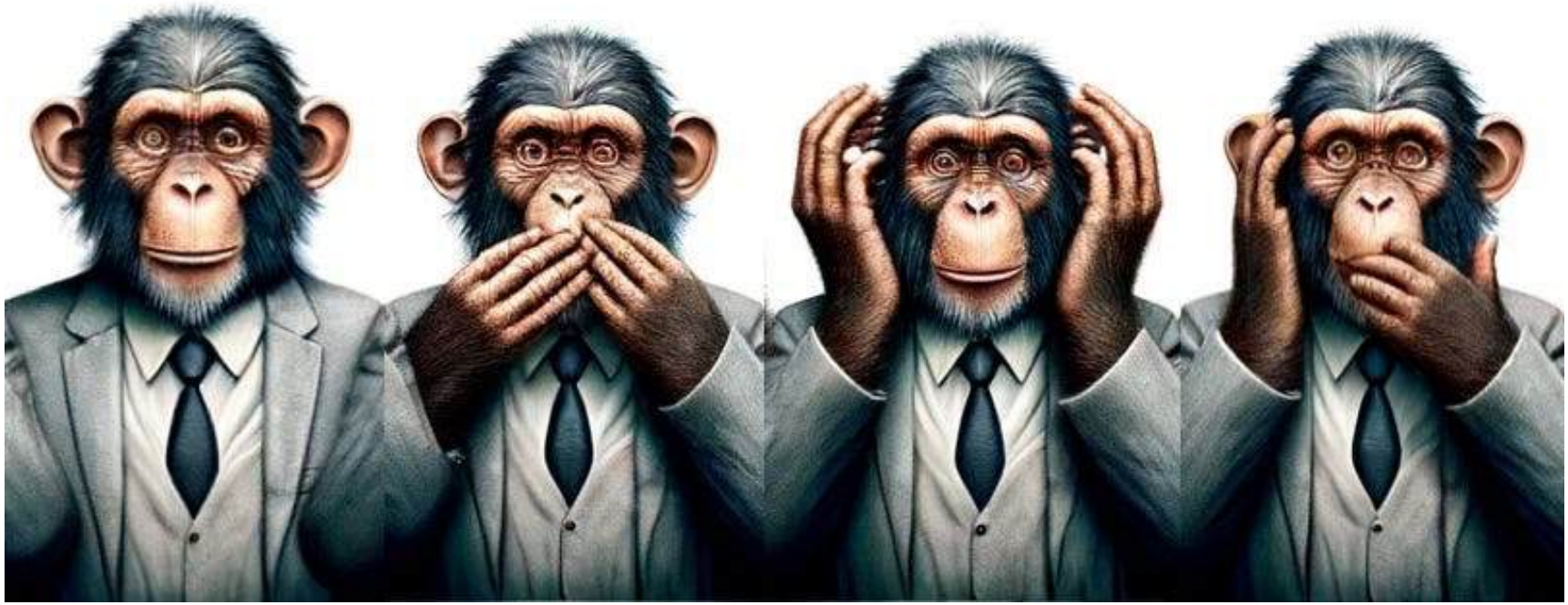
Standard Message Passing
each node Broadcasts & Listens



Cooperative Message Passing
each node individually decides

Finkelshtein, Huang, B, Ceylan 2023

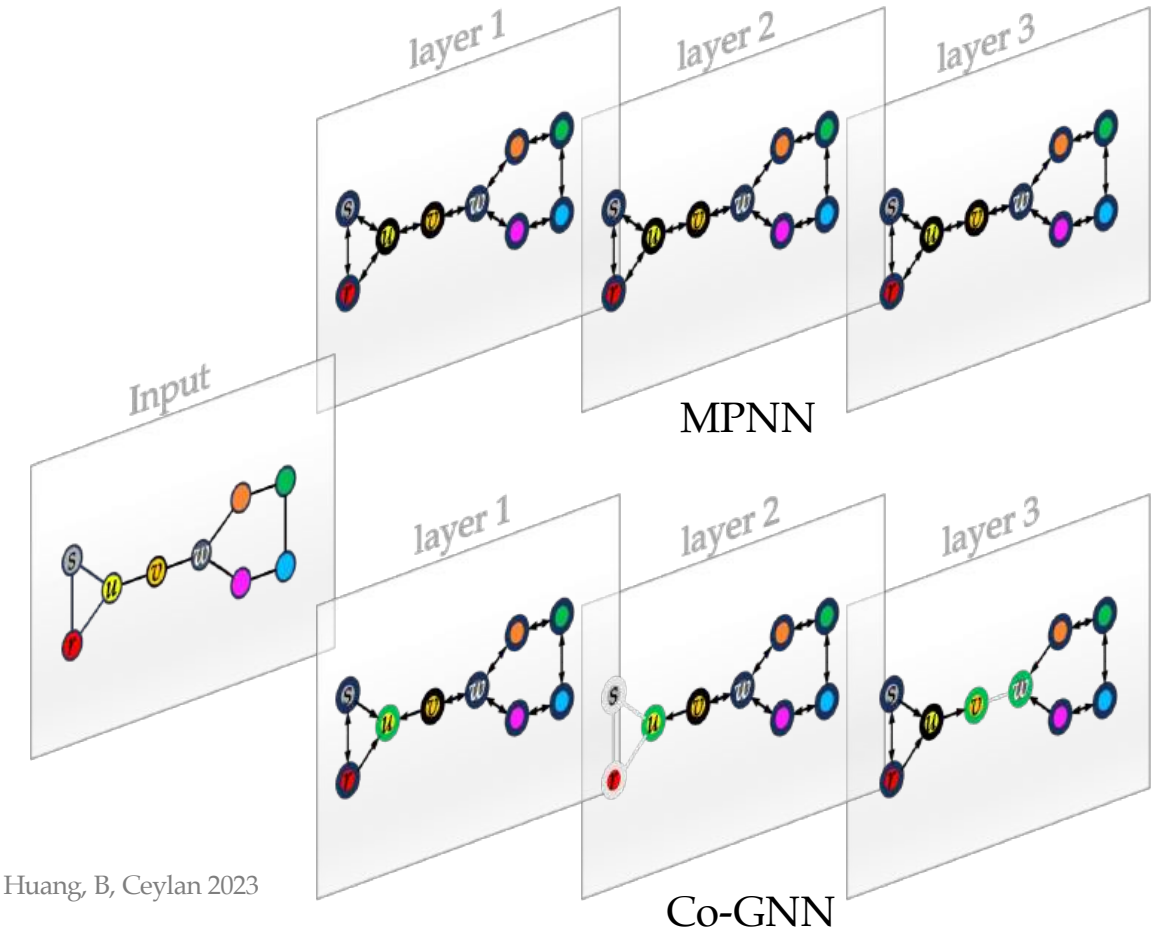
Cooperative Message Passing

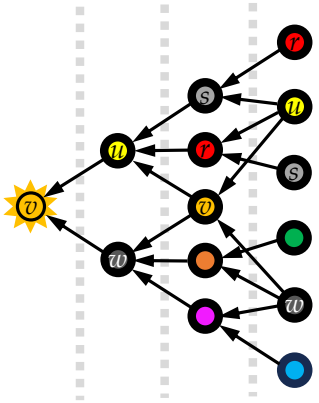


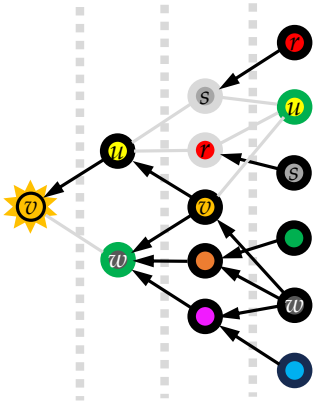
Broadcast & Listen | **Listen** | **Broadcast** | **Isolate**

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

Cooperative Message Passing



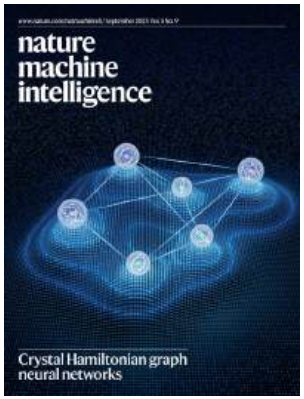


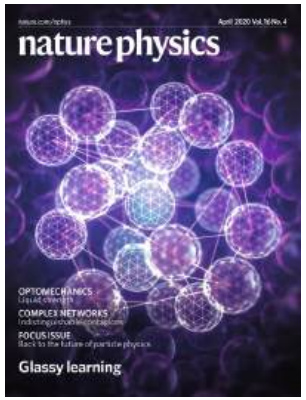


Finkelshtein, Huang, B, Ceylan 2023

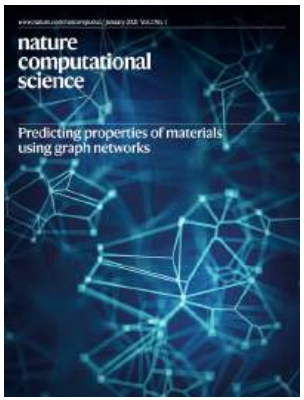
What do we gain from physics-inspired GNNs?

- 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*

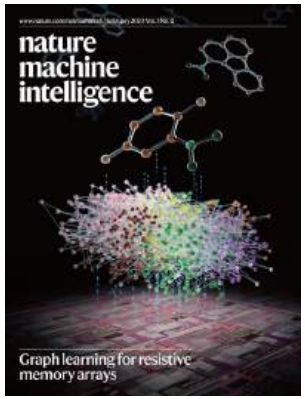




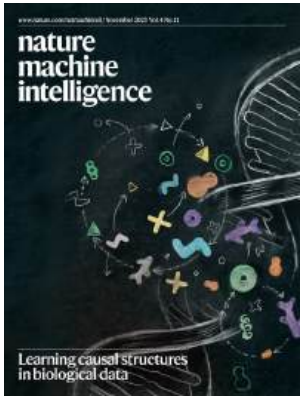
Physics

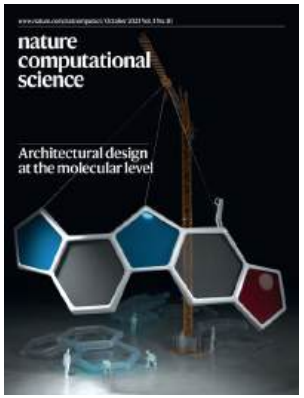


New materials

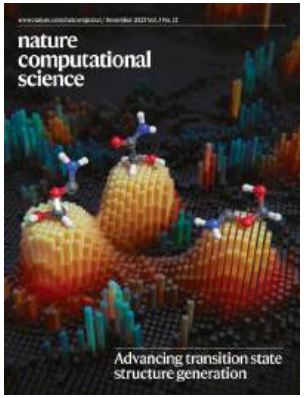


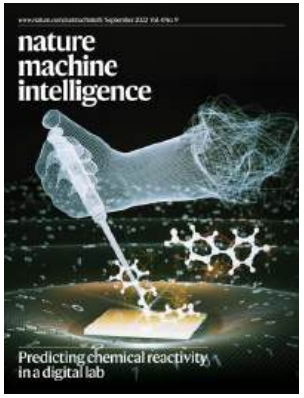
Chip design





Biology



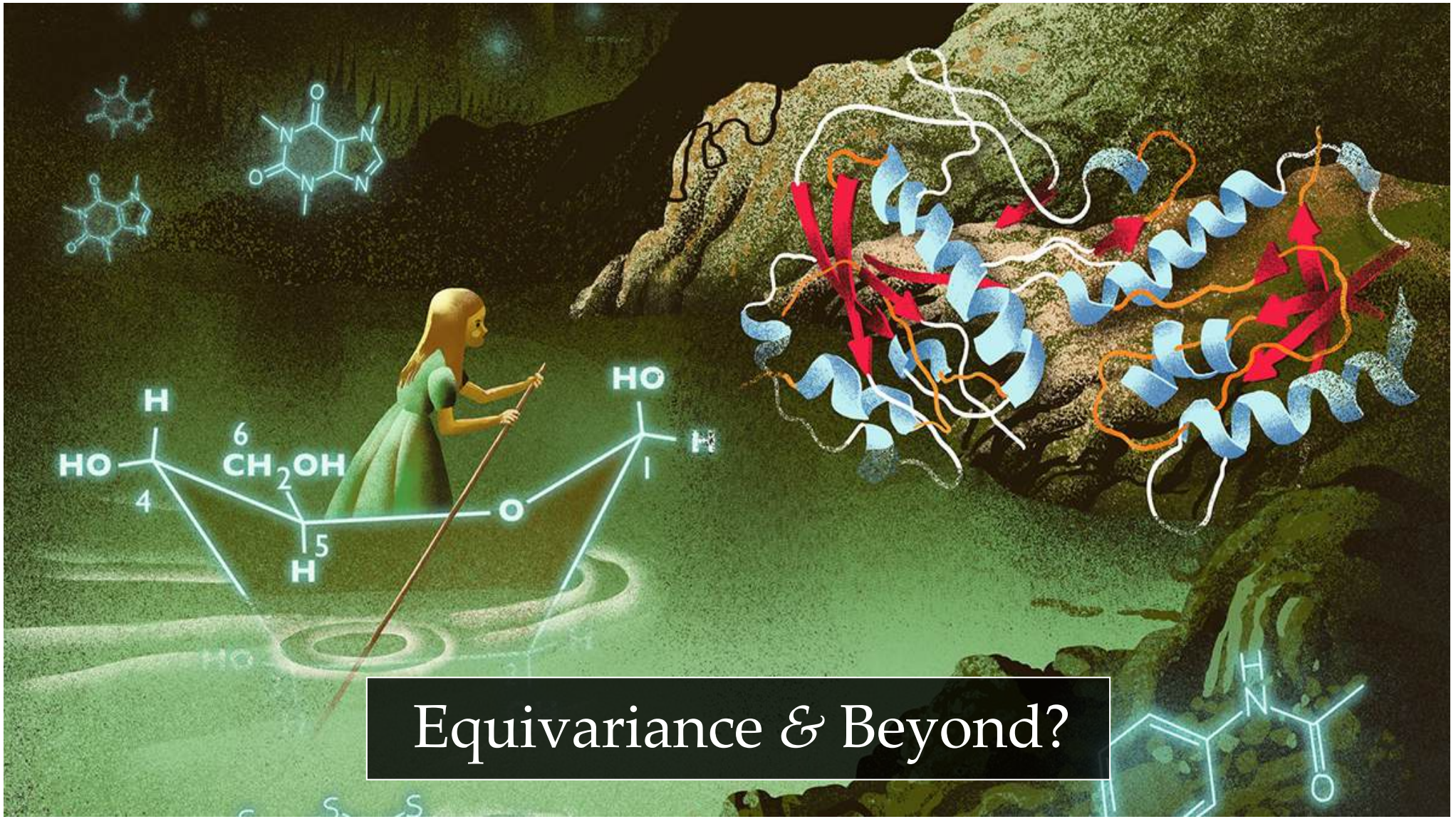


Chemistry

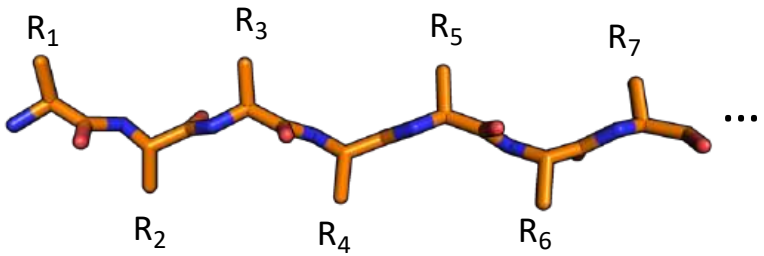
Urban planning

Pure math

Weather

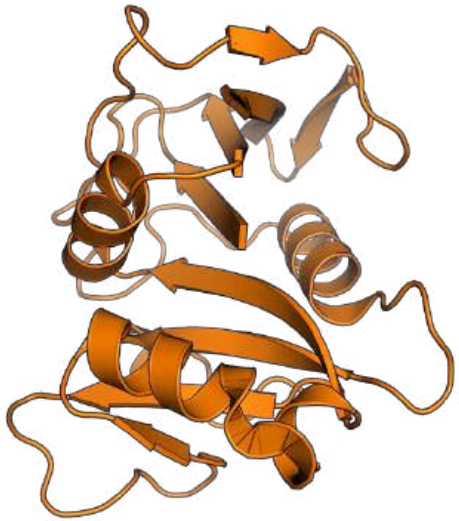

Equivariance & Beyond?

Protein Folding



Sequence of amino acids

Protein folding



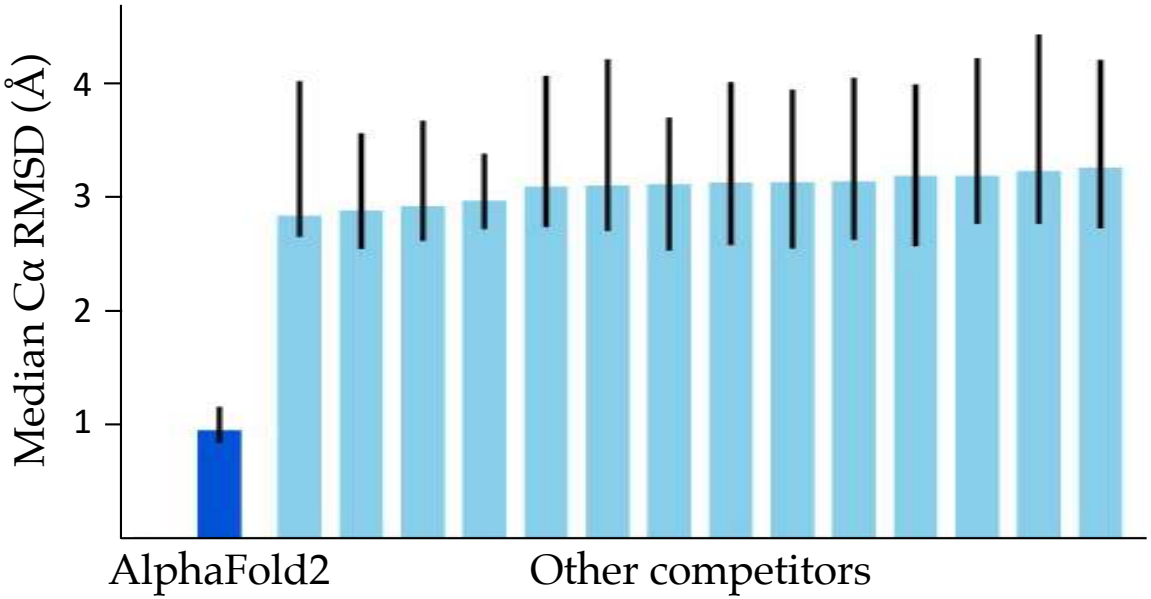
3D structure

"[protein] conformation is determined by the totality of interatomic interactions and hence by the amino acid sequence"

C. Anfinsen

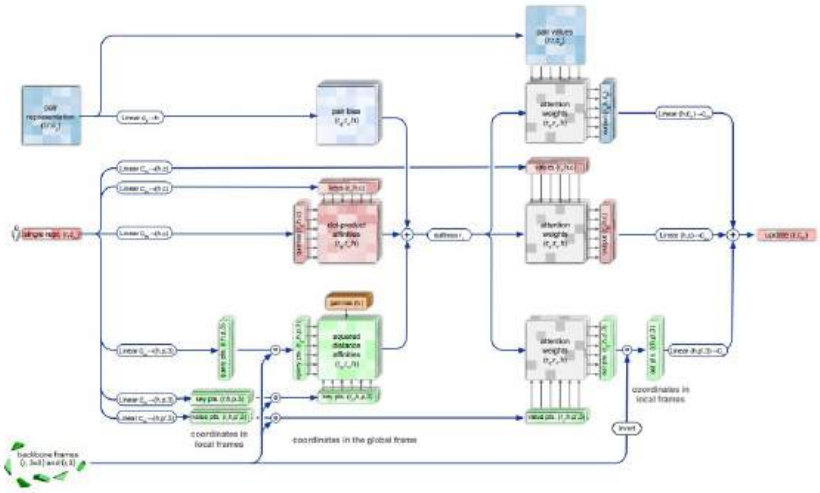
Anfinsen 1972 Nobel Lecture; Portrait: Ihor Gorskyi

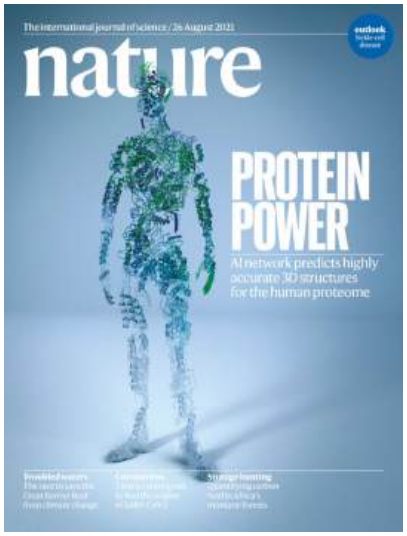
AlphaFold2: the “ImageNet moment” of Structural Biology



Jumper et al. 2022; CASP14

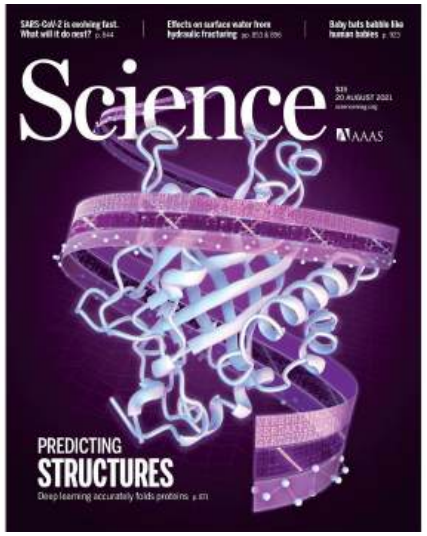
Revolution in Structural Biology





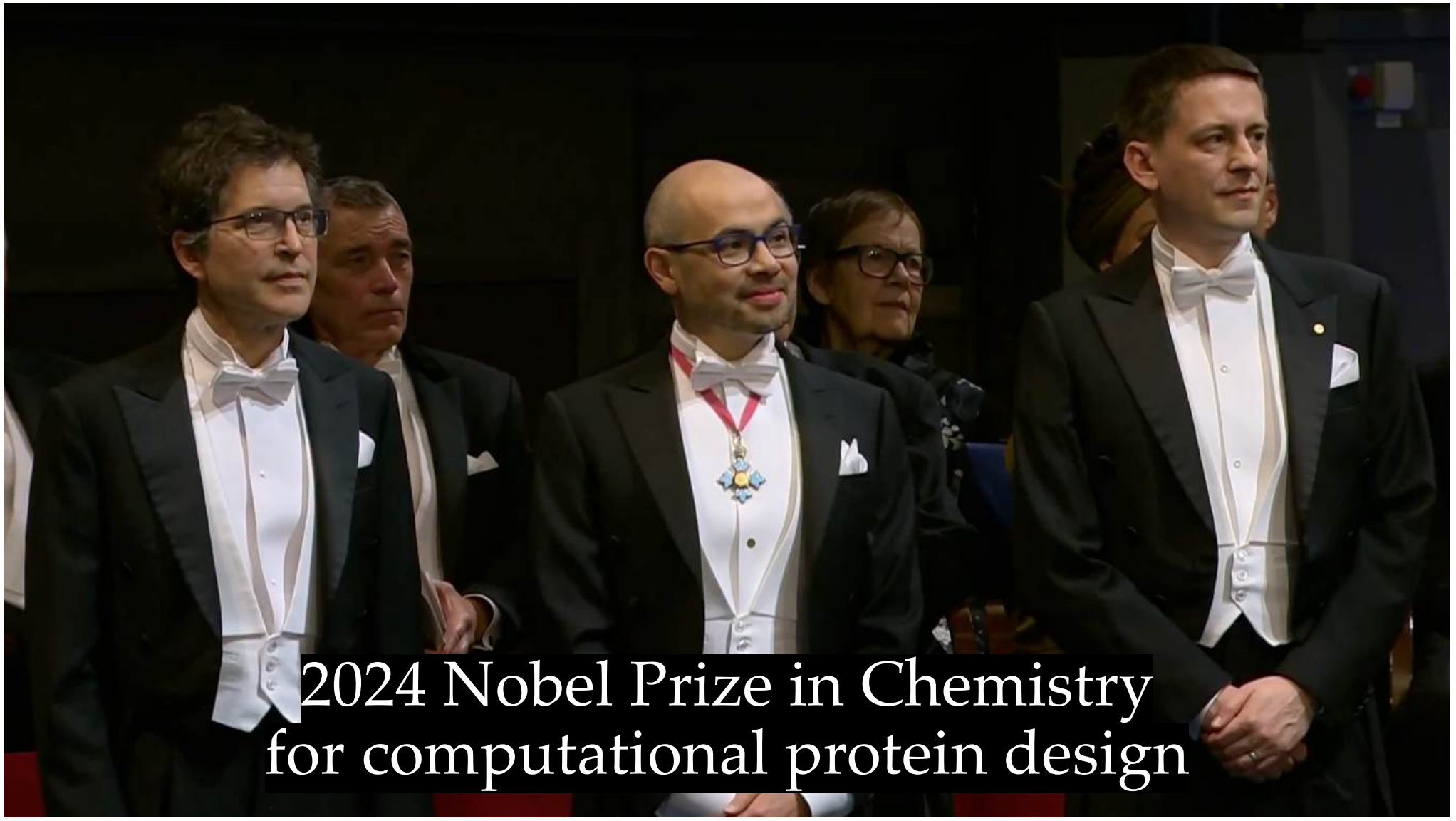
Jumper et al. 2021

AlphaFold 2
"Invariant point attention"



Baek et al. 2021

RosettaFold
SE(3)-equivariant Transformer

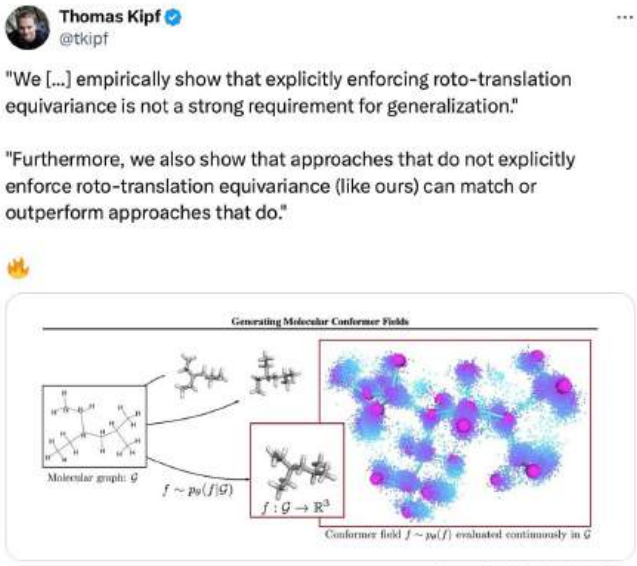

2024 Nobel Prize in Chemistry
for computational protein design


AND THEY LIVED
HAPPILY EVER AFTER

The Bitter Lesson: Equivariance is dead...long live equivariance!

Equivariance is the idea of giving a model the inductive biases to natively handle rotations, translations and (sometimes) reflections. It has been at the core of Geometric Deep Learning and biomolecular modelling research since AlphaFold 2. However, recent works by top labs have questioned the existing mantra.

- The first shots were fired by Apple, with a paper that obtained SOTA results on predicting the 3D structures of small molecules using a non-equivariant diffusion model with a transformer encoder.
- Remarkably, the authors showed that using the domain-agnostic model did not deleteriously impact generalization and was consistently able to outperform specialist models (assuming sufficient scale was used).
- Next was AlphaFold 3, which infamously dropped all the equivariance and frames constraints from the previous model in favour of another diffusion process coupled with augmentations and, of course, scale.
- Regardless, the greatly improved training efficiency of equivariant models means the practice is likely to stay for a while (at least for academic groups working on large systems such as proteins).



stateof.ai 2024

Benaich et al., State of AI report 2024

Swallowing the Bitter Pill: Simplified Scalable Conformer Generation

Yuyang Wang [1] **Ahmed A. Elhag** [1 2] **Navdeep Jaitly** [1] **Joshua M. Susskind** [1] **Miguel Ángel Bautista** [1]

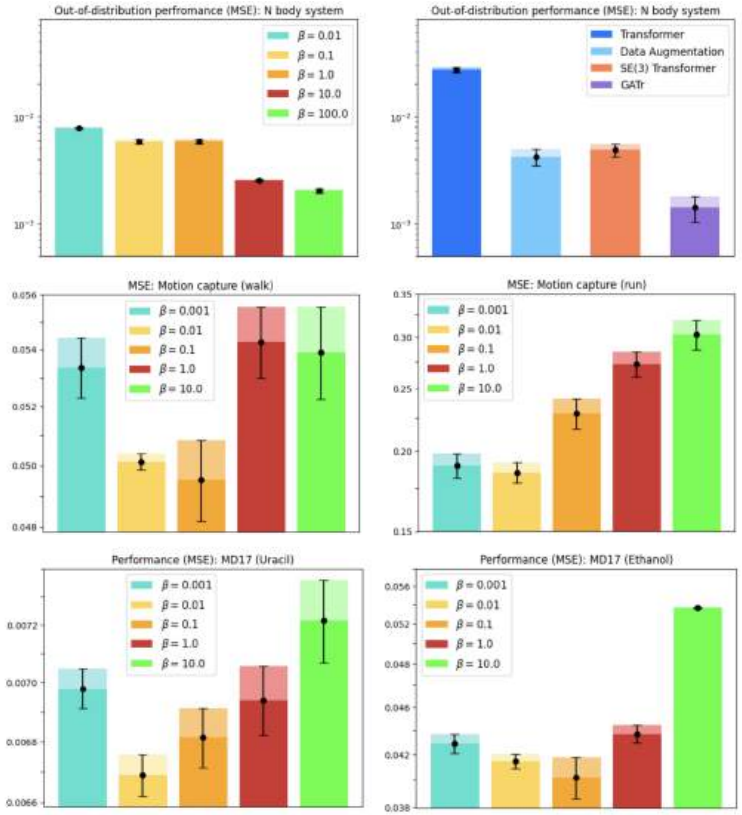
Abstract

We present a novel way to predict molecular conformers through a simple formulation that sidesteps many of the heuristics of prior works and achieves state of the art results by using the advantages of scale. By training a diffusion generative model directly on 3D atomic positions without making assumptions about the explicit structure of molecules (*e.g.* modeling torsional angles) we are able to radically simplify structure learning, and make it trivial to scale up the model sizes. This model, called Molecular Conformer Fields (MCF), works by parameterizing conformer structures as functions that map elements from a molecular graph directly to their 3D location in space. This formulation allows us to boil down the essence of structure prediction to learning a distribution over functions. Experimental results show that scaling up the model capacity leads to large gains in generalization performance *without enforcing inductive biases* like rotational equivariance. MCF represents an advance in extending diffusion models to handle complex scientific problems in a conceptually simple, scalable and effective manner.

is the vast complexity of the 3D structure space, encompassing factors such as bond lengths and torsional angles. Despite the molecular graph dictating potential 3D conformers through specific constraints, such as bond types and spatial arrangements determined by chiral centers, the conformational space experiences exponential growth with the expansion of the graph size and the number of rotatable bonds (Axelrod & Gomez-Bombarelli, 2022). This complicates brute force and exhaustive approaches, making them virtually unfeasible for even moderately small molecules.

Systematic methods, like OMEGA (Hawkins et al., 2010), offer rapid processing through rule-based generators and curated torsion templates. Despite their efficiency, these models typically fail on complex molecules, as they often overlook global interactions and are tricky to extend to inputs like transition states or open-shell molecules. Classic stochastic methods, like molecular dynamics (MD) and Markov chain Monte Carlo (MCMC), rely on extensively exploring the energy landscape to find low-energy conformers. Such techniques suffer from sampling inefficiency for large molecules and struggle to generate diverse representative conformers (Hawkins, 2017; Wilson et al., 1991; Grebner et al., 2011). In the domain of learning-based approaches, several works have looked at conformer generation problems through the lens of probabilistic modeling, using either

Wang, Elhag et al. 2023

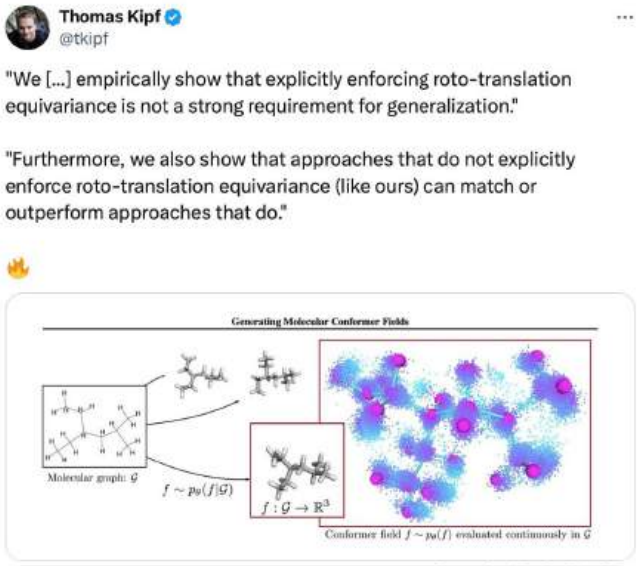


Elhag et B, 2024

The Bitter Lesson: Equivariance is dead...long live equivariance!

Equivariance is the idea of giving a model the inductive biases to natively handle rotations, translations and (sometimes) reflections. It has been at the core of Geometric Deep Learning and biomolecular modelling research since AlphaFold 2. However, recent works by top labs have questioned the existing mantra.

- The first shots were fired by Apple, with a paper that obtained SOTA results on predicting the 3D structures of small molecules using a non-equivariant diffusion model with a transformer encoder.
- Remarkably, the authors showed that using the domain-agnostic model did not deleteriously impact generalization and was consistently able to outperform specialist models (assuming sufficient scale was used).
- Next was AlphaFold 3, which infamously dropped all the equivariance and frames constraints from the previous model in favour of another diffusion process coupled with augmentations and, of course, scale.
- Regardless, the greatly improved training efficiency of equivariant models means the practice is likely to stay for a while (at least for academic groups working on large systems such as proteins).



stateof.ai 2024

Benaich et al., State of AI report 2024


EQUIVARIANCE
BRUTE-FORCE
DATA AUG-MENTATION
Quo vadimus?

The Hardware Lottery

Sara Hooker

Google Research, Brain Team

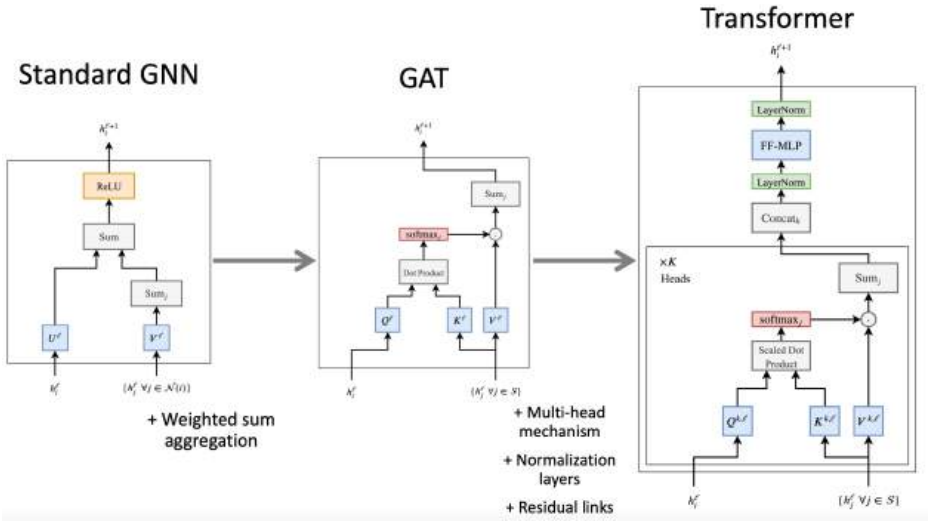
Hardware, systems and algorithms research communities have historically had different incentive structures and fluctuating motivation to engage with each other explicitly. This historical treatment is odd given that hardware and software have frequently determined which research ideas succeed (and fail). This essay introduces the term hardware lottery to describe when a research idea wins because it is suited to the available software and hardware and *not* because the idea is superior to alternative research directions. Examples from early computer science history illustrate how hardware lotteries can delay research progress by casting successful ideas as failures. These lessons are particularly salient given the advent of domain specialized hardware which make it increasingly costly to stray off of the beaten path of research ideas. This essay posits that the gains from progress in computing are likely to become even more uneven, with certain research directions moving into the fast-lane while progress on others is further obstructed.

Transformers are Graph Neural Networks

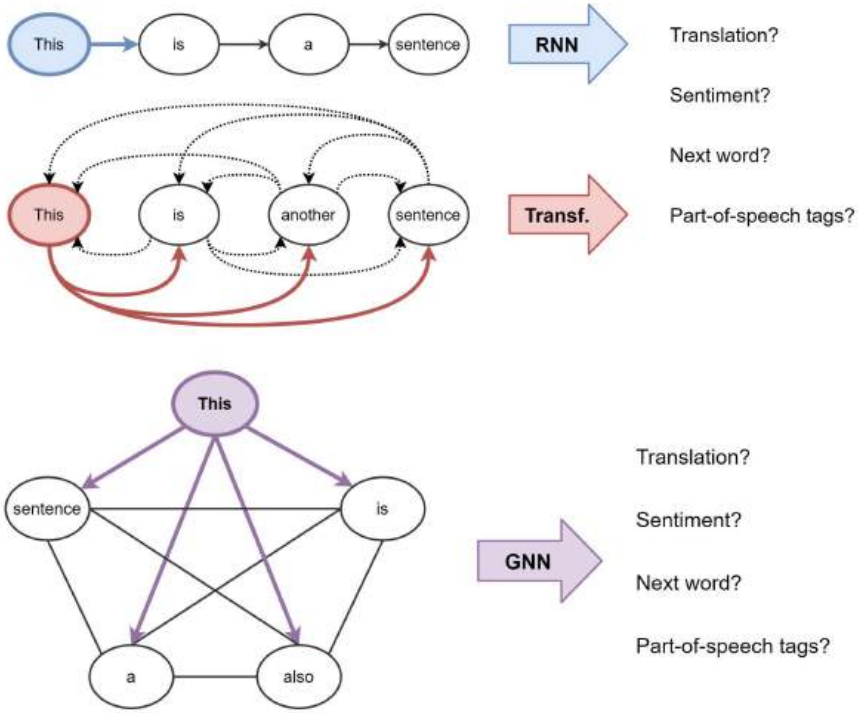
Exploring the connection between Transformer models such as GPT and BERT for Natural Language Processing, and Graph Neural Networks.

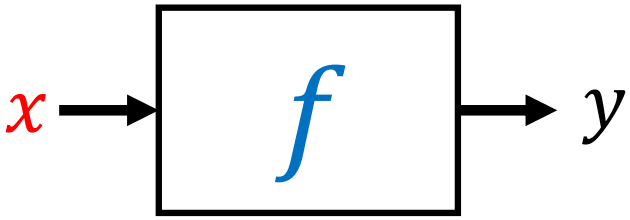
Chaitanya K. Joshi

Last updated on Jun 21, 2021 · 12 min read



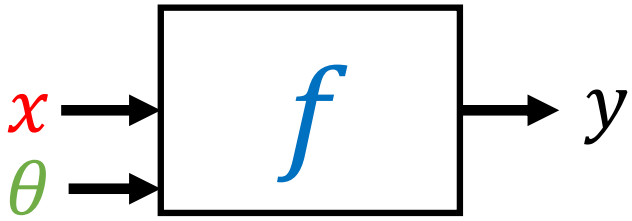
Joshi 2021





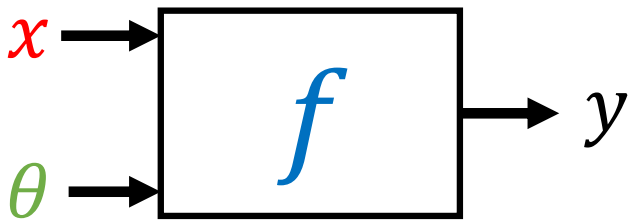
"How f interacts with the group G acting on x?"

$$f(g.x) = f(x)$$



"How f interacts with the group G acting on x?"

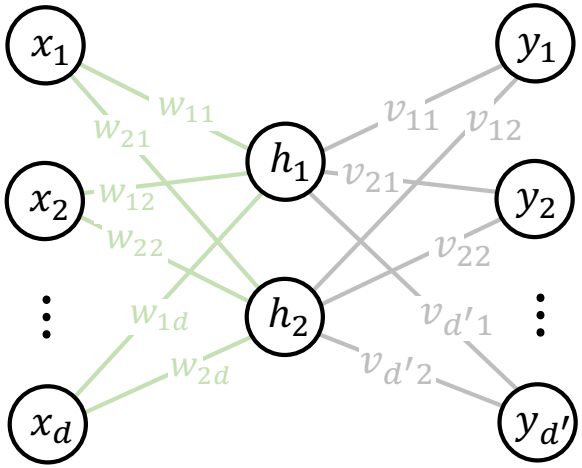
$$f(g.x) = f(x)$$



"How f interacts with the group G acting on x and H acting on θ?"

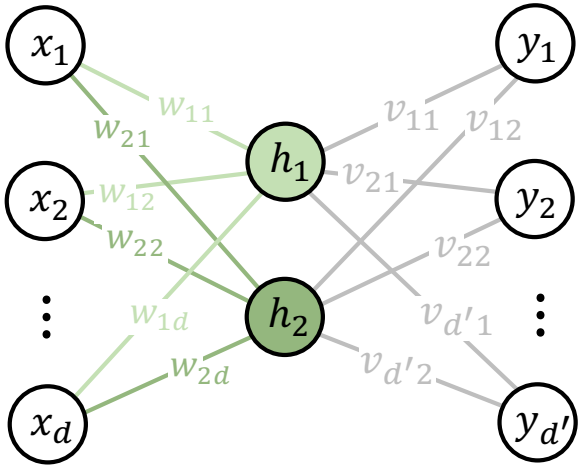
$$f(g.x, h.\theta) = f(x, \theta)$$

Symmetries of the Weights



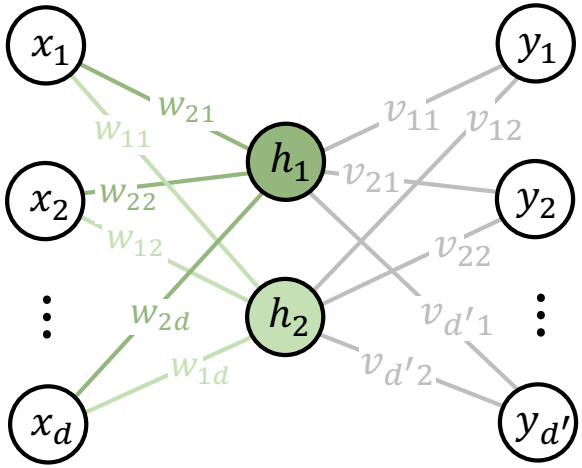
$$\mathbf{y} = \mathbf{V}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



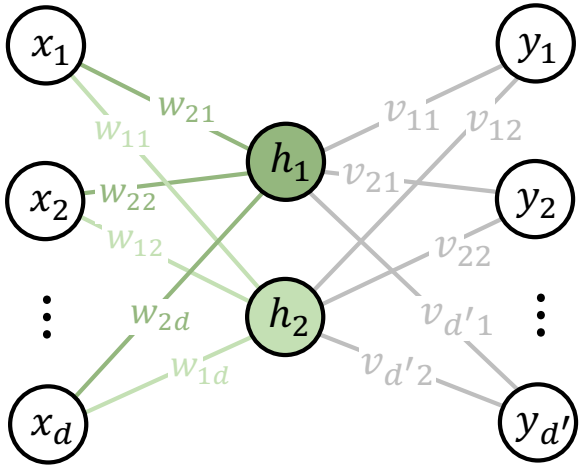
$$\mathbf{y} = \mathbf{V}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



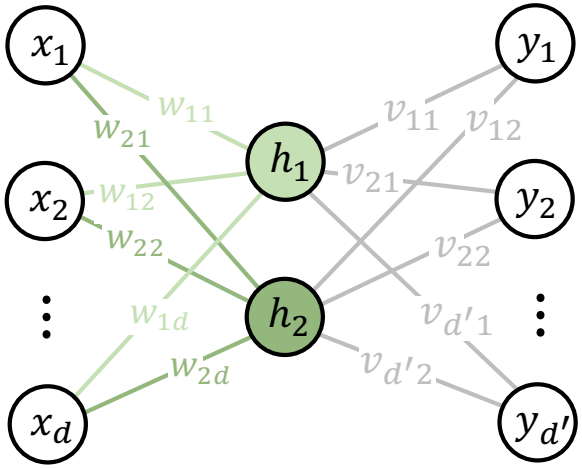
$$\mathbf{y} = \mathbf{V}\,\sigma\left(\mathbf{\Pi}\,\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



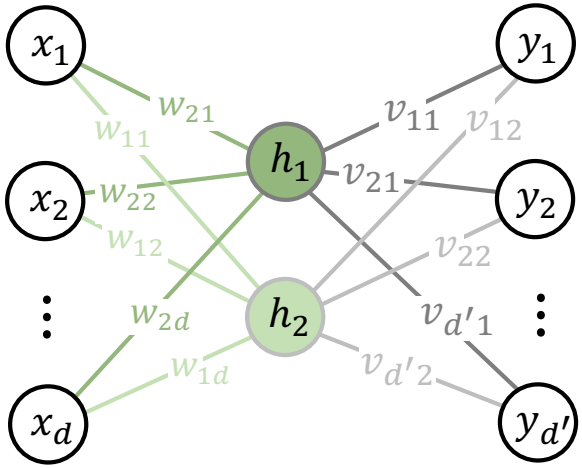
$$\mathbf{y} = \mathbf{V}\,\Pi\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



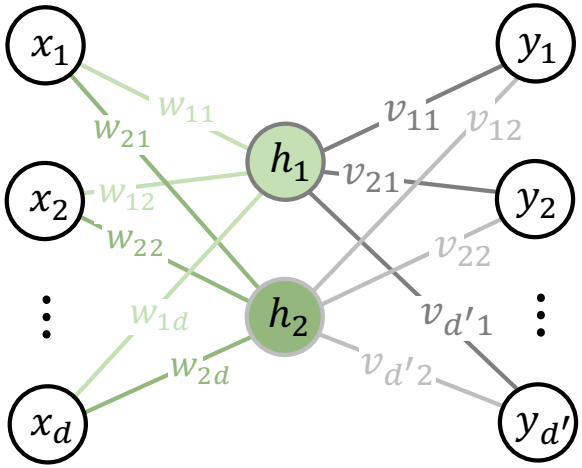
$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



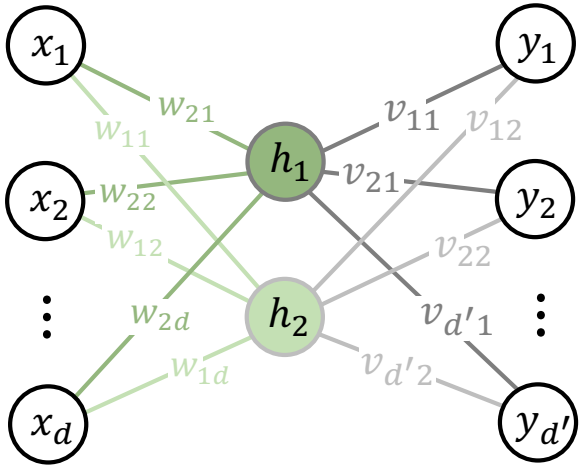
$$\mathbf{y} = \mathbf{V}\,\Pi\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



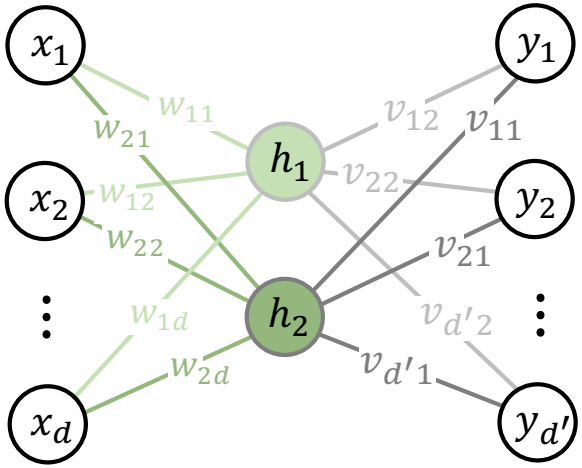
$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



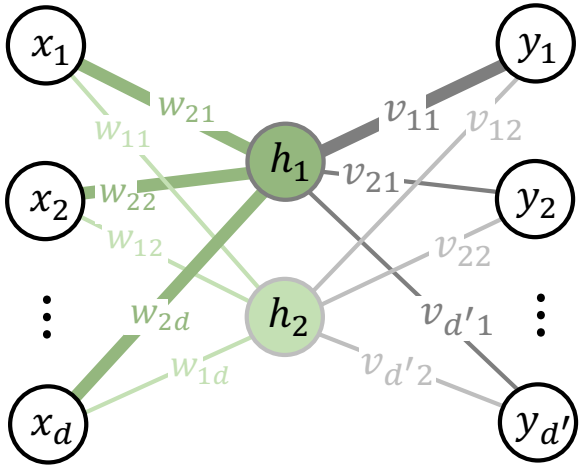
$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}^{\mathrm{T}}\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



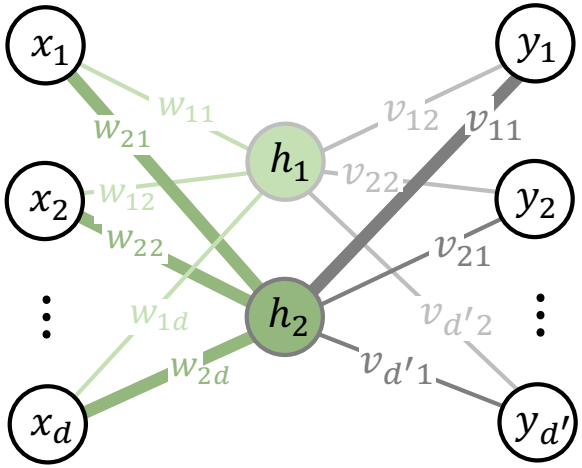
$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}^{\mathrm{T}}\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}^{\mathrm{T}}\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights



$$\mathbf{y} = \mathbf{V}\,\mathbf{\Pi}^{\mathrm{T}}\mathbf{\Pi}\,\sigma\left(\mathbf{W}\,\mathbf{x}\right)$$

Symmetries of the Weights

- L-layer neural network with weights $\boldsymbol{\theta} = (\mathbf{W}_1, \mathbf{b}_1, \ldots, \mathbf{W}_L, \mathbf{b}_L)$
- Parameter space symmetry $G = \mathrm{S}_{d_1} \times \cdots \times \mathrm{S}_{d_L}$

$$\mathbf{W}_1' = \boldsymbol{\Pi}_1^{\mathrm{T}} \mathbf{W}_1 \qquad \mathbf{b}_1' = \boldsymbol{\Pi}_1^{\mathrm{T}} \mathbf{b}_1$$

$$\mathbf{W}_l' = \boldsymbol{\Pi}_l^{\mathrm{T}} \mathbf{W}_l \boldsymbol{\Pi}_{l-1} \qquad \mathbf{b}_l' = \boldsymbol{\Pi}_l^{\mathrm{T}} \mathbf{b}_l$$

$$\vdots \qquad \vdots$$

$$\mathbf{W}_L' = \mathbf{W}_L \boldsymbol{\Pi}_{L-1} \qquad \mathbf{b}_L' = \mathbf{b}_L$$

such that $f(\cdot, g\boldsymbol{\theta}) = f(\cdot, \boldsymbol{\theta})$

Symmetries in the Gradient Space

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{B}} \underbrace{\ell(f(\mathbf{x},\boldsymbol{\theta}),\mathbf{y})}_{\mathcal{L}_{(\mathbf{x},\mathbf{y})}}$$

$$\nabla_{\mathbf{b}_l}\mathcal{L}_{(\mathbf{x},\mathbf{y})} = \mathbf{g}_l \qquad \text{where } \mathbf{g}_l = \frac{\partial \ell(f(\mathbf{x},\boldsymbol{\theta}),\mathbf{y})}{\partial \mathbf{u}_l}$$

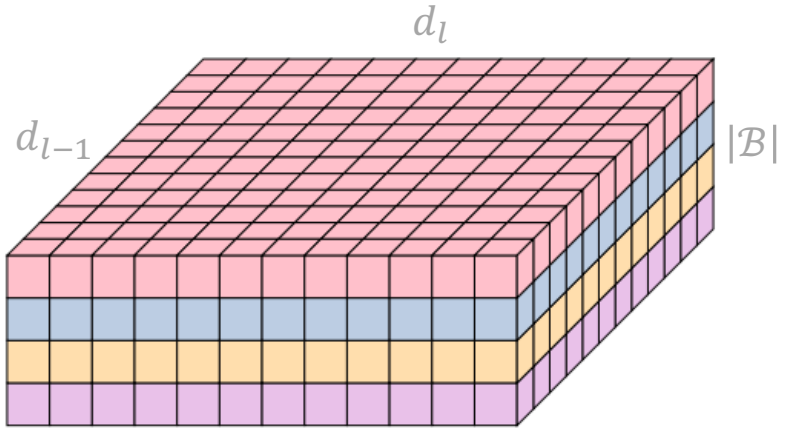
$$\nabla_{\mathbf{W}_l}\mathcal{L}_{(\mathbf{x},\mathbf{y})} = \mathbf{g}_l \mathbf{a}_{l-1}^{\mathrm{T}}$$

Gelberg et B, Maron 2025

Symmetries in the Gradient Space

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{B}} \ell(f(\mathbf{x},\boldsymbol{\theta}),\mathbf{y})$$

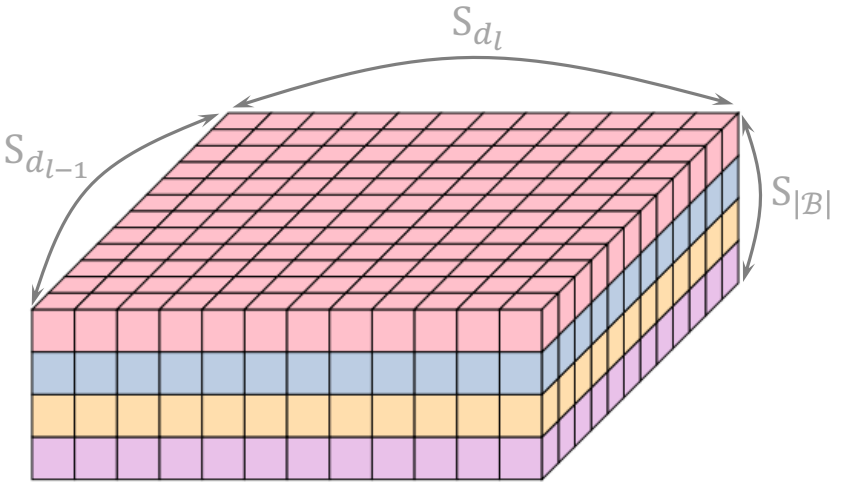
$$\nabla_{\mathbf{W}_l} =$$



Symmetries in the Gradient Space

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{B}} \ell(f(\mathbf{x},\boldsymbol{\theta}),\mathbf{y})$$

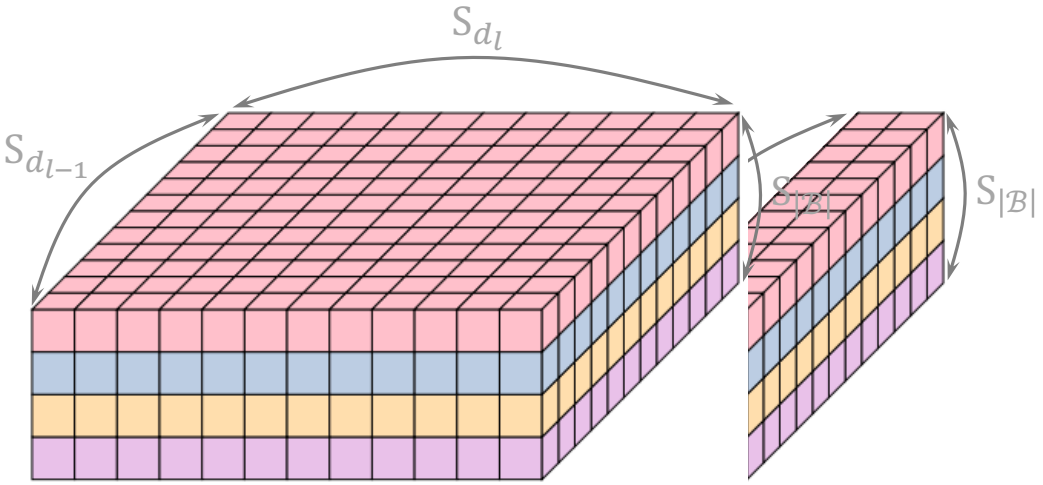
$$\nabla_{\mathbf{W}_l} =$$



Symmetries in the Gradient Space

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{B}} \ell(f(\mathbf{x},\boldsymbol{\theta}),\mathbf{y})$$

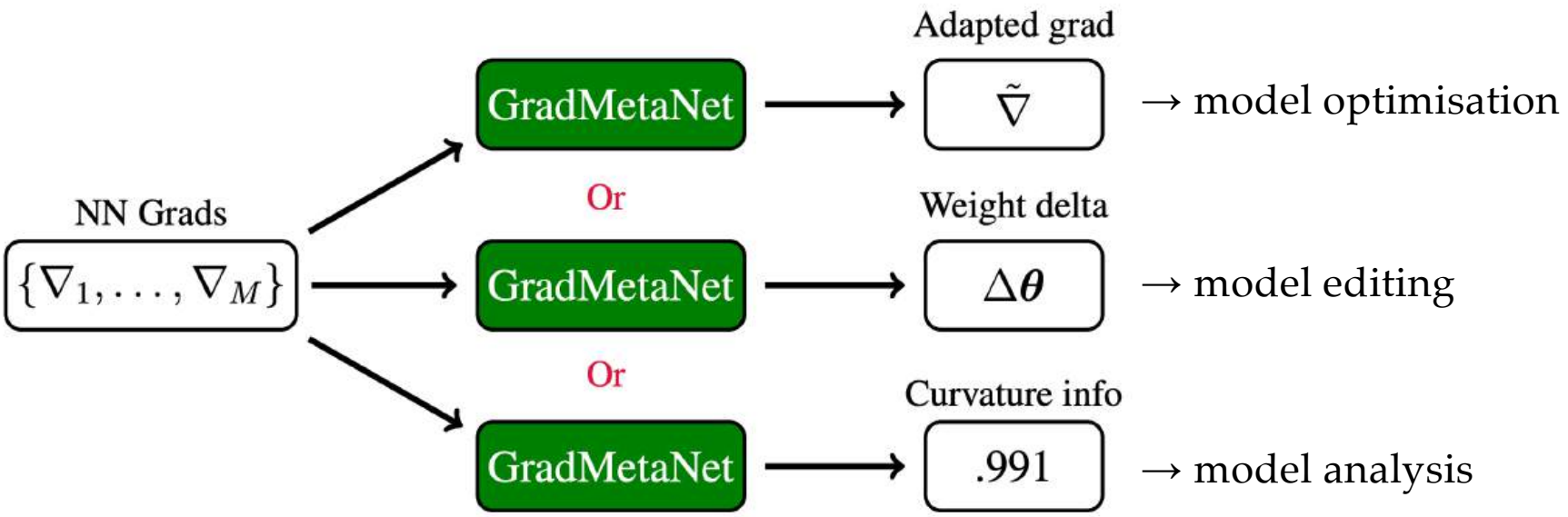
$$\nabla_{\mathbf{W}_l} =$$



$$G_b = S_{|\mathcal{B}|} \times S_{d_1} \times \cdots \times S_{d_L}$$

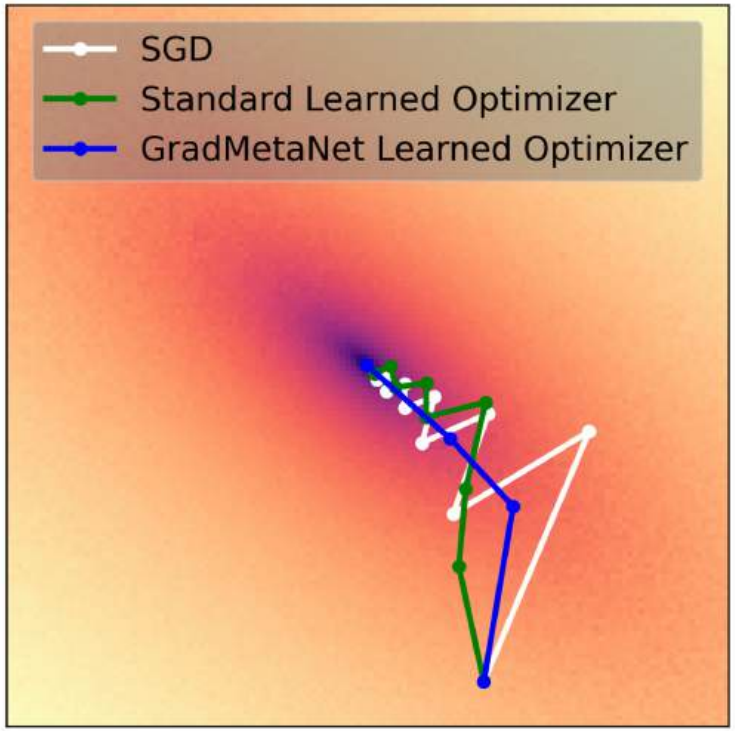
Gelberg et B, Maron 2025

GradMetaNet



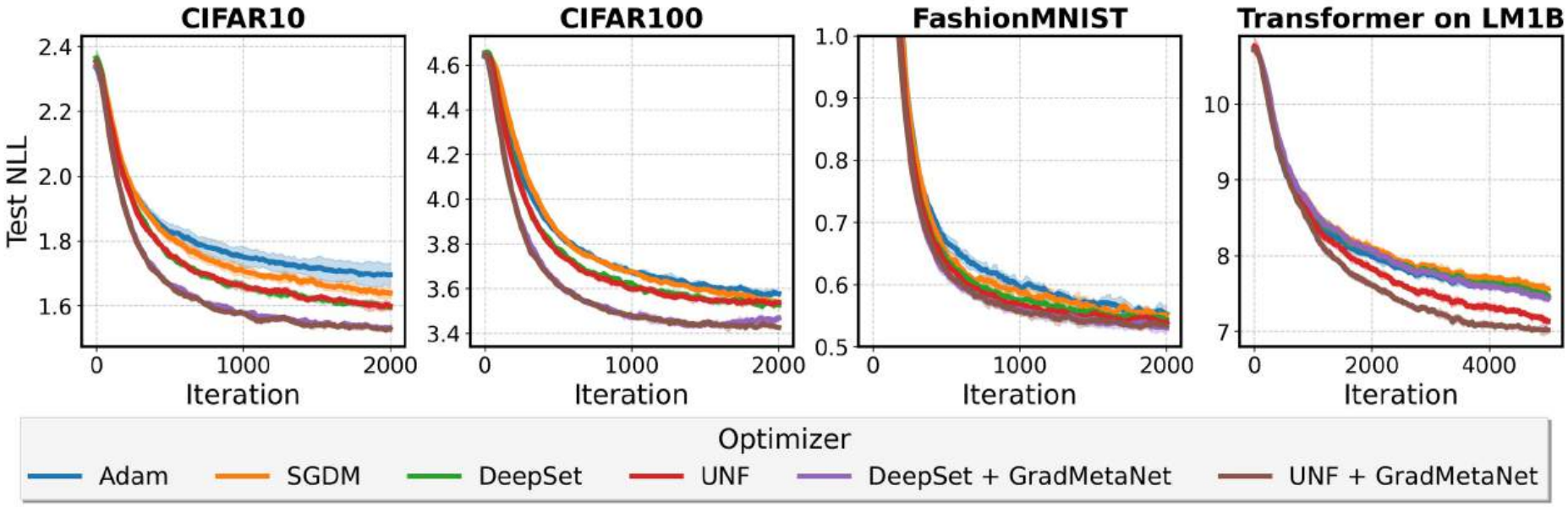
Gelberg et B, Maron 2025

GradMetaNet: Learning Optimisation

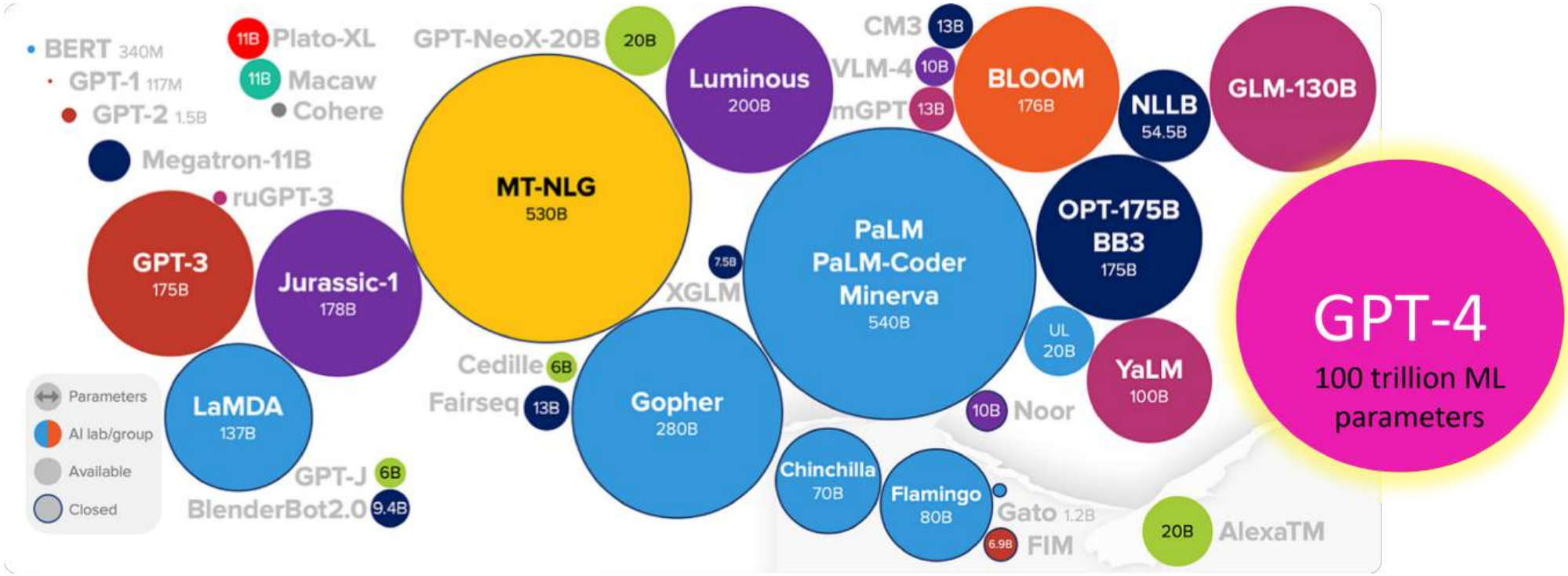


Gelberg et B, Maron 2025

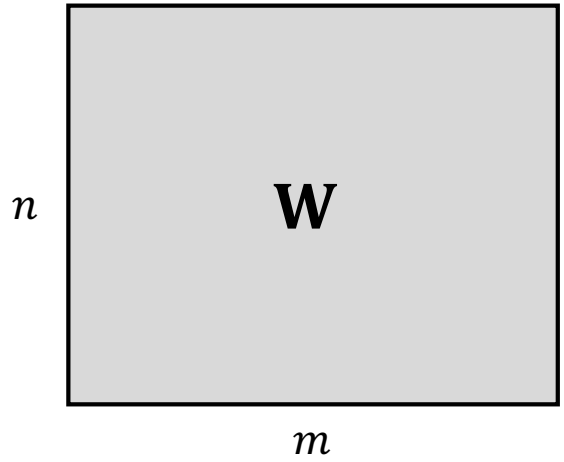
GradMetaNet: Learning Optimisation



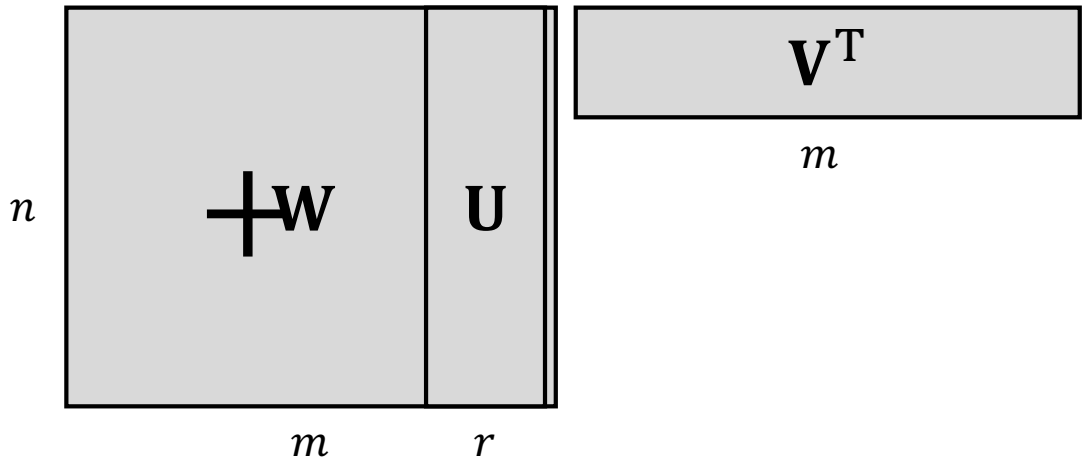
Gelberg et B, Maron 2025


BERT 340M
GPT-1 117M
GPT-2 1.5B
Megatron-11B
11B Plato-XL
11B Macaw
Cohere
ruGPT-3
GPT-3
175B
Jurassic-1
178B
LaMDA
137B
GPT-J 6B
BlenderBot2.0 9.4B
MT-NLG
530B
GPT-NeoX-20B 20B
Luminous
200B
7.5B XGLM
Cedille 6B
Fairseq 13B
Gopher
280B
PaLM
PaLM-Coder
Minerva
540B
CM3 13B
VLM-4 10B
mGPT 13B
BLOOM
176B
NLLB
54.5B
GLM-130B
OPT-175B
BB3
175B
UL
20B
YaLM
100B
10B Noor
Chinchilla
70B
Flamingo
80B
Gato 1.2B
6.9B FIM
20B AlexaTM
GPT-4
100 trillion ML
parameters
Parameters
AI lab/group
Available
Closed

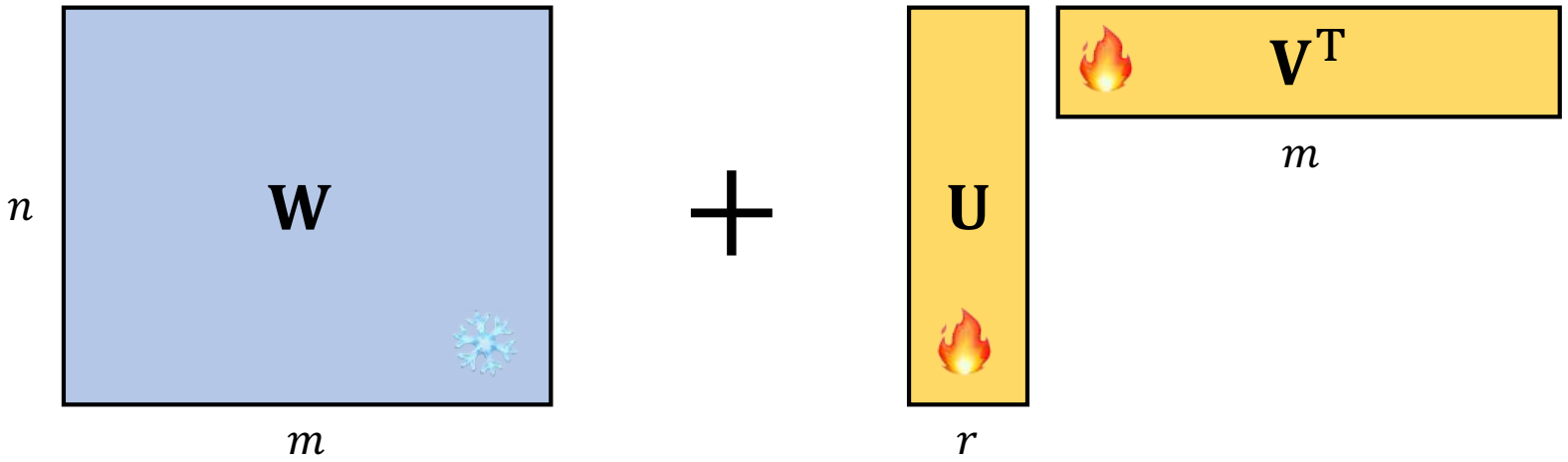
Low-Rank Adaptors (LoRA)



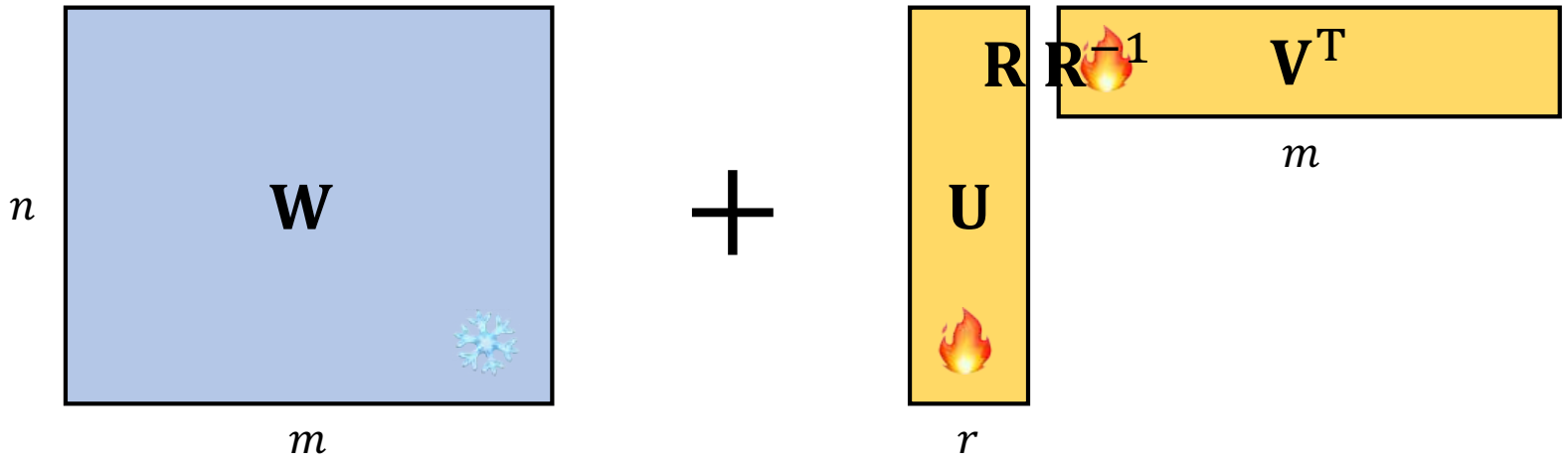
Low-Rank Adaptors (LoRA)



Low-Rank Adaptors (LoRA)



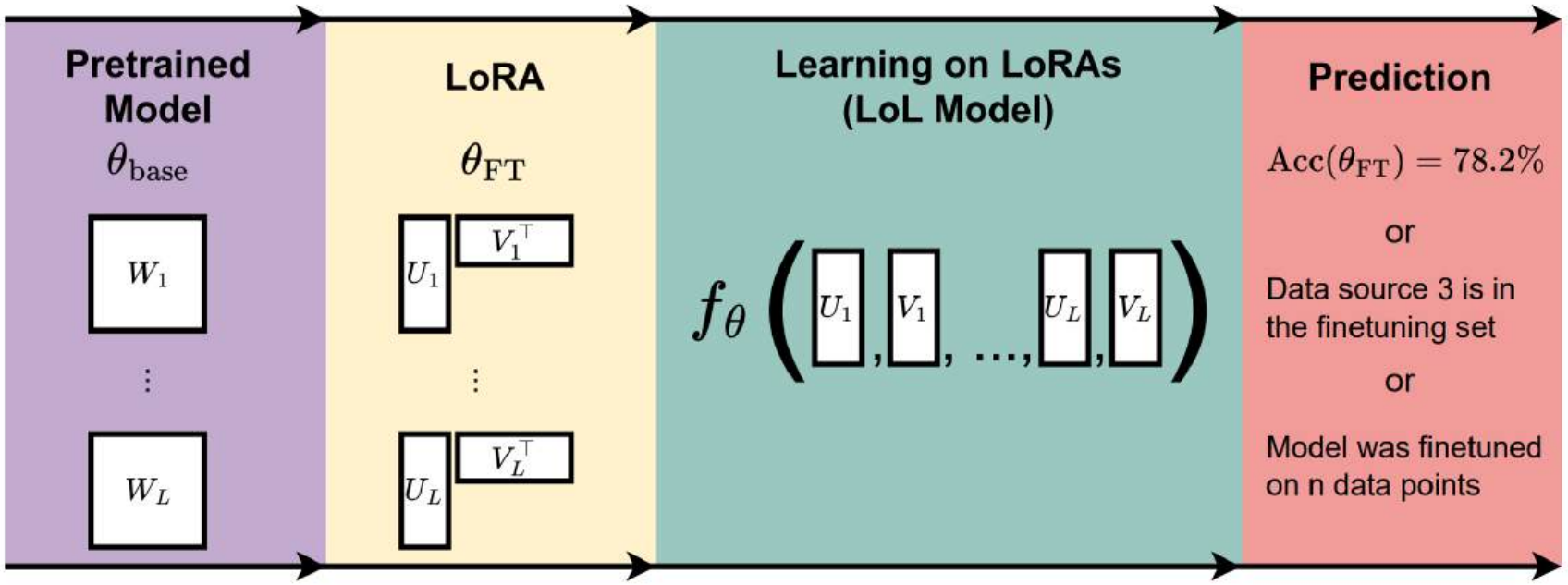
Low-Rank Adaptors (LoRA)



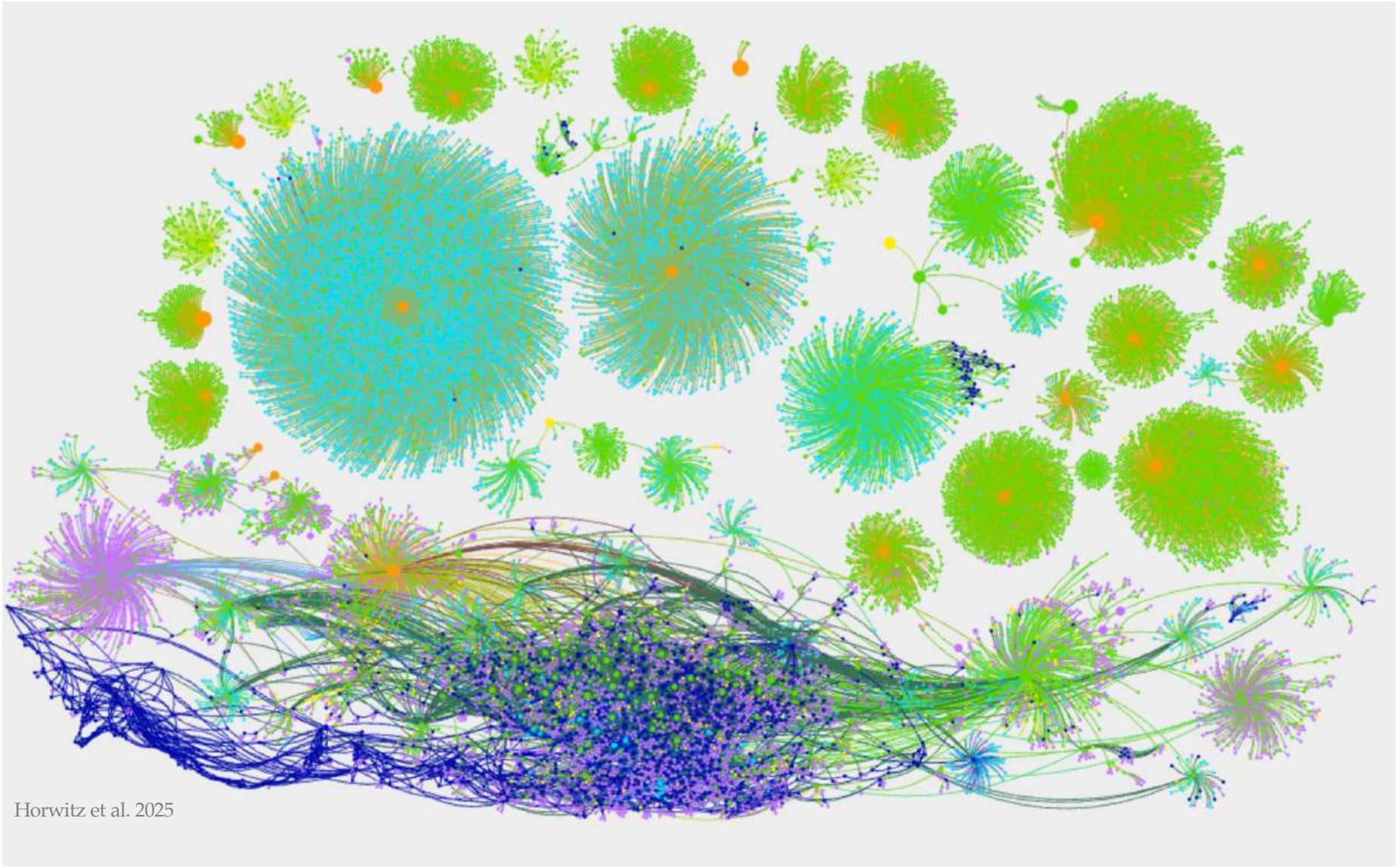
LoRA $(\mathbf{U}, \mathbf{V})$ *defined up to* $\mathrm{GL}(r)$

Putterman, Lim, Gelberg et B 2025

Learning on LoRAs (LOL)

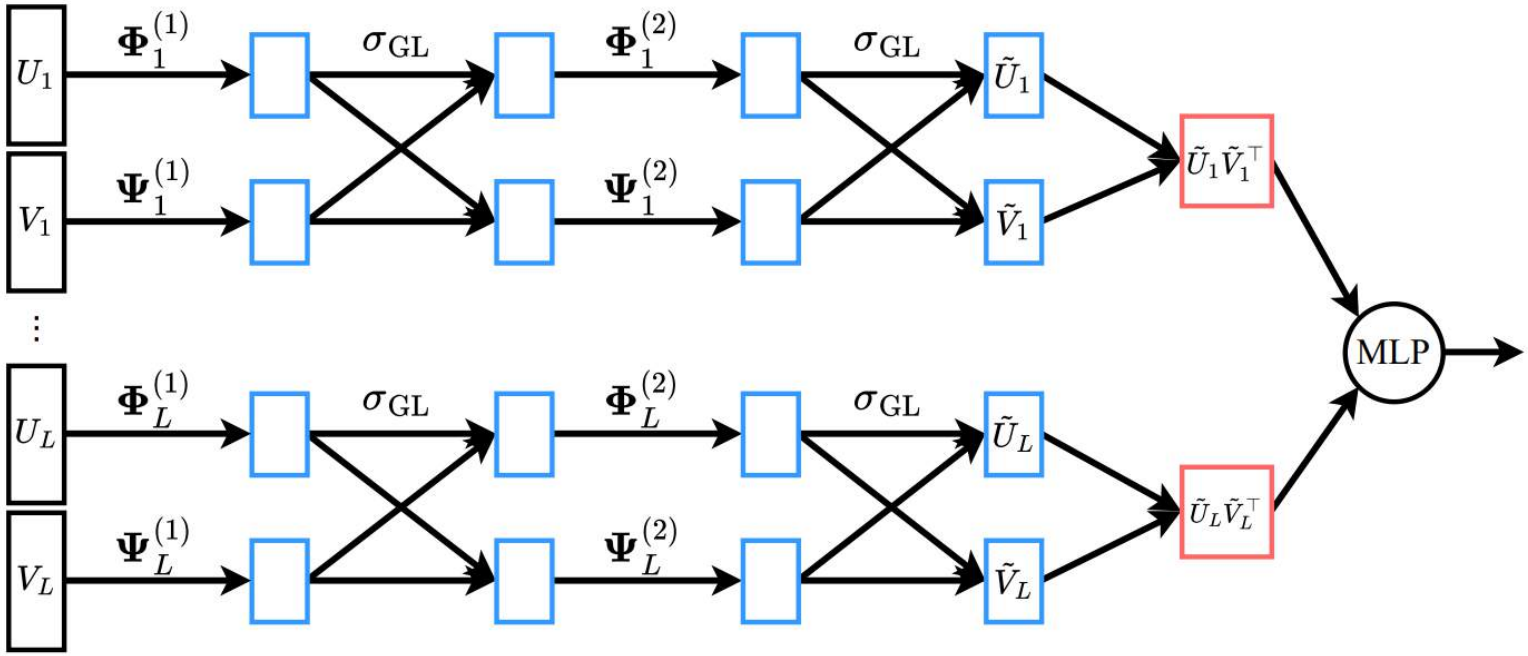


Putterman, Lim, Gelberg et B 2025



Horwitz et al. 2025

Learning on LoRAs (LOL)



Putterman, Lim, Gelberg et B 2025

Learning on LoRAs (LOL)

LoL Model	GL-Inv.	O-Inv.	Expressive	Preprocess Time	Forward Time
Naive architectures					
MLP($[U, V]$)	✗	✗	✓	$O((m+n)r)$	$O((m+n)r)$
Transformer($[U, V]$)	✗	✗	✓	$O((m+n)r)$	$O((m+n)r)$
MLP($UV^\top$)	✓	✓	✓	$\boldsymbol{O(mnr)}$	$\boldsymbol{O(mn)}^2$
Efficient symmetry-aware architectures					
MLP(O-Align($[U, V]$))	✗	✓	✓	$O((m+n)r^2)$	$O((m+n)r)$
MLP($\sigma(UV^\top)$)	✓	✓	✗	$O((m+n)r^2)$	$O((m+n)r)$
GL-net	✓	✓	✓	$O((m+n)r)$	$O((m+n)r)$

Putterman, Lim, Gelberg et B 2025

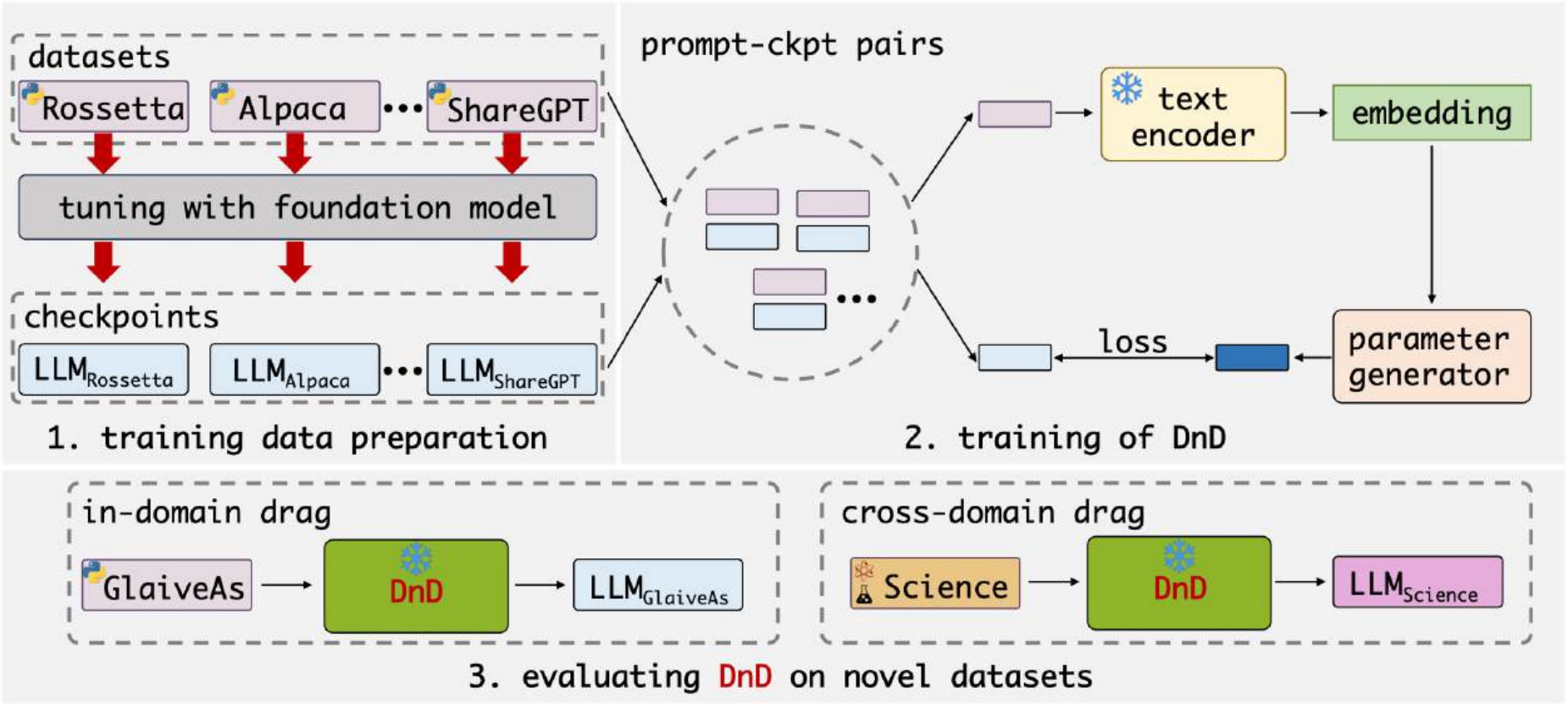
Learning on LoRAs (LOL)

	LoL Model	CelebA Attributes		Imagenette Classes	
		Test Loss (↓)	Test Acc (↑)	Test Loss (↓)	Test Acc (↑)
Naive Models	$\text{MLP}([U, V])$	$.554 \pm .000$	72.4 ± 0.0	$.709 \pm .004$	49.6 ± 1.3
	$\text{Transformer}([U, V])$	$.586 \pm .014$	73.2 ± 0.9	$.695 \pm .001$	50.0 ± 1.3
	$\text{MLP}(UV^\top)$	$.267 \pm .007$	89.1 ± 0.4	$.264 \pm .011$	88.9 ± 0.6
Efficient Invariant	$\text{MLP}(\text{O-Align}([U, V]))$	$.333 \pm .008$	87.2 ± 0.5	$.278 \pm .008$	87.8 ± 0.3
	$\text{MLP}(\sigma(UV^\top))$	$.509 \pm .013$	77.3 ± 1.3	$.638 \pm .013$	65.6 ± 0.6
	GL-net	$\mathbf{.232 \pm .007}$	$\mathbf{91.3 \pm 0.1}$	$\mathbf{.244 \pm .005}$	$\mathbf{90.4 \pm 0.3}$

Using LoL models to predict *CelebA* attributes (left) and *Imagenette* classes (right) of the finetuning data of diffusion models, given only the LoRA weights.

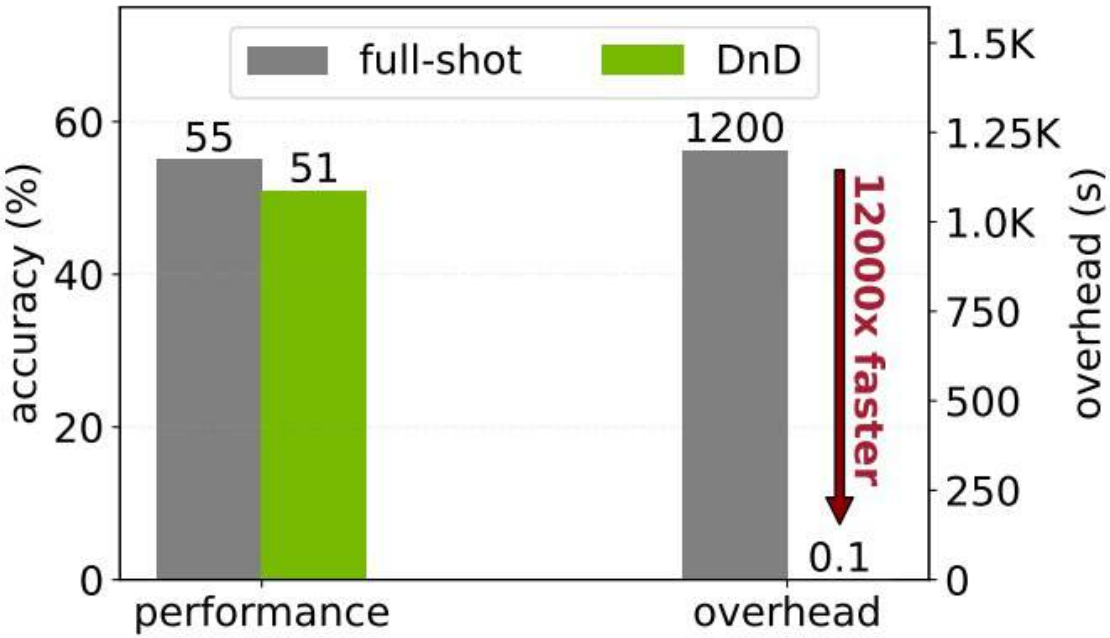
Putterman, Lim, Gelberg et B 2025

Drag-and-Drop LLMs



Liang, Tang, Zhou et B 2025

Drag-and-Drop LLMs



Liang, Tang, Zhou et B 2025

THE STORY CONTINUES IN THE WEIGHT SPACE

ICLR
International Conference On Learning Representations

First Workshop on Weight Space Learning

https://weight-space-learning.github.io/

Invited Speaker

Stella X. Yu
University of Michigan

Michael Mahoney
UC Berkeley, ICSI

Boris Knyazev
Samsung AI Lab (SAIT)

Naomi Saphra
Harvard University

Ludwig Schmit
Stanford University / Anthropic

Organization

Konstantin Schürholt

Giorgos Bouritsas

Eliahu Horwitz

Derek Lim

Yoav Gelberg

Bo Zhao

Allan Zhou

Damian Borth

Stefanie Jegelka

Steering

Michael Bronstein

Gal Chechik

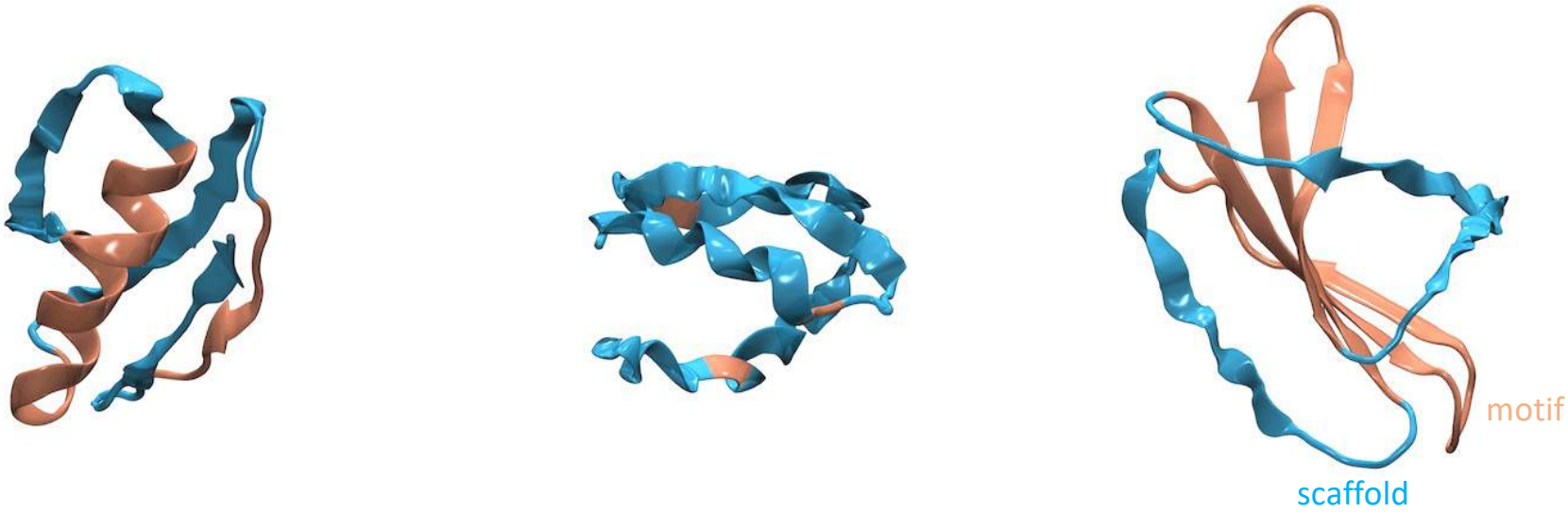
Stella X. Yu

Haggai Maron

Yedid Hoshen


Thank you!

Motif-Scaffolding

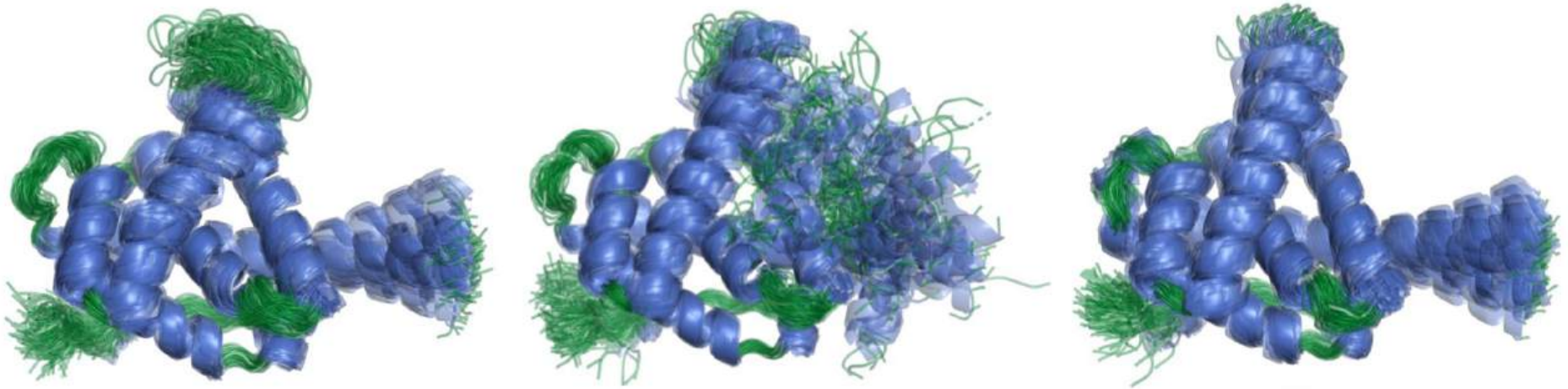


Designing novel proteins where functional part (motif) is known and the remaining part (scaffold) is generated

Huguet et Bose, Tong, B 2024 (FoldFlow++)

dreamfold

Zero-shot Molecular Dynamics

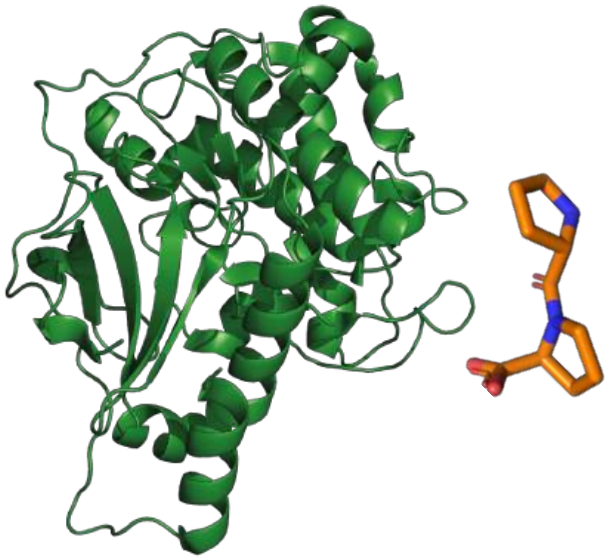


Groundtruth **ESMFlow-MD** **FoldFlow++**

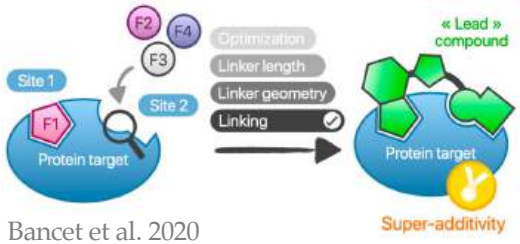
Protein conformations generated by different models

Jing et al. 2024 (ESMFlow); Huguet et Bose, Tong, B 2024 (FoldFlow++)

dreamfold



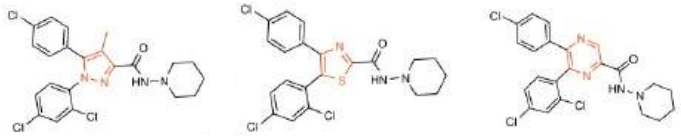
Fragment-based



Bancet et al. 2020

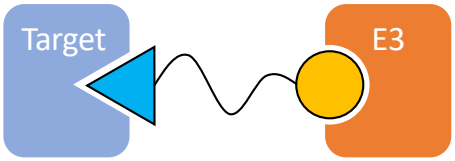
- Molecule growing
- Fragment merging
- Fragment linking

Scaffold hopping



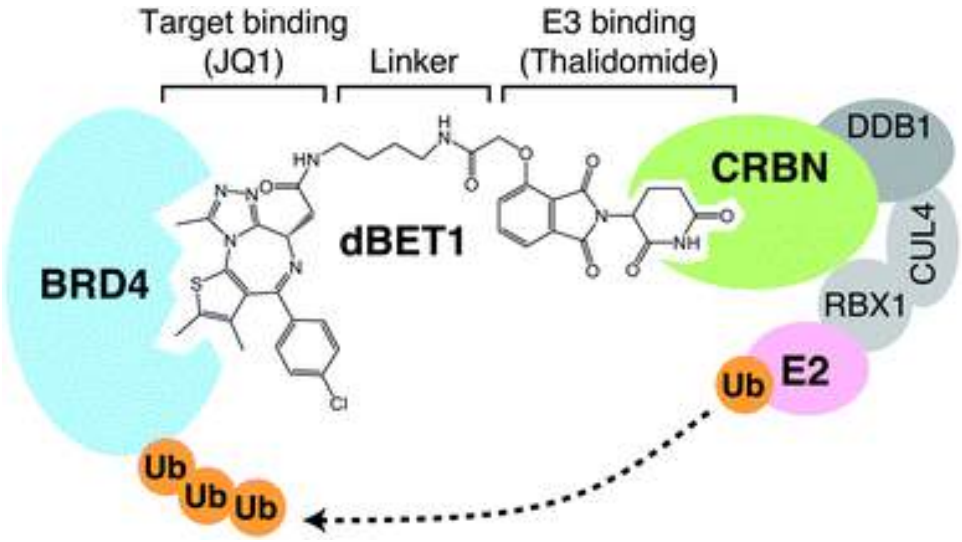
Sun et al. 2012

Targeted protein degradation



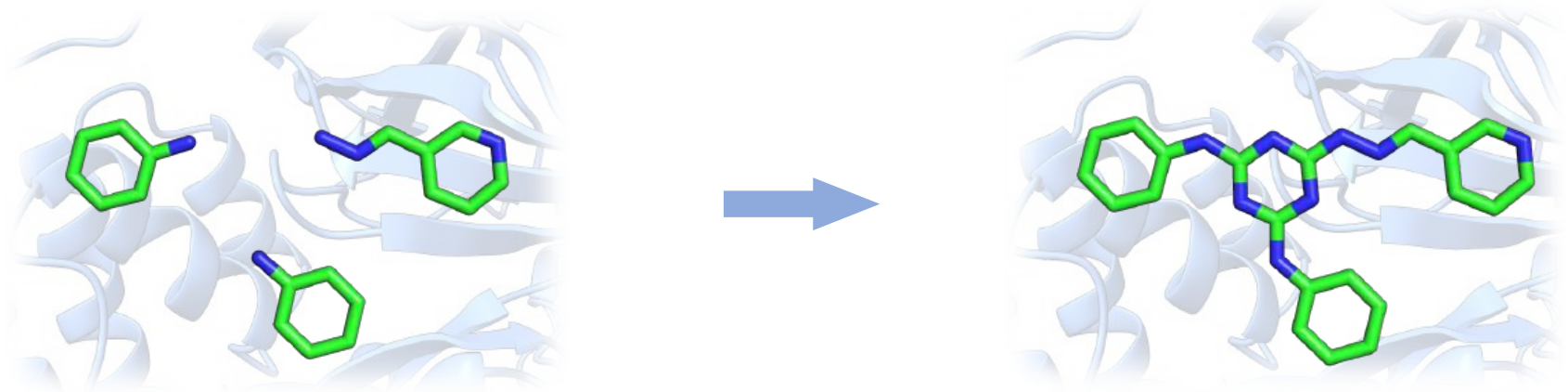
Békés et al. 2022

- PROTAC
- Molecular glue



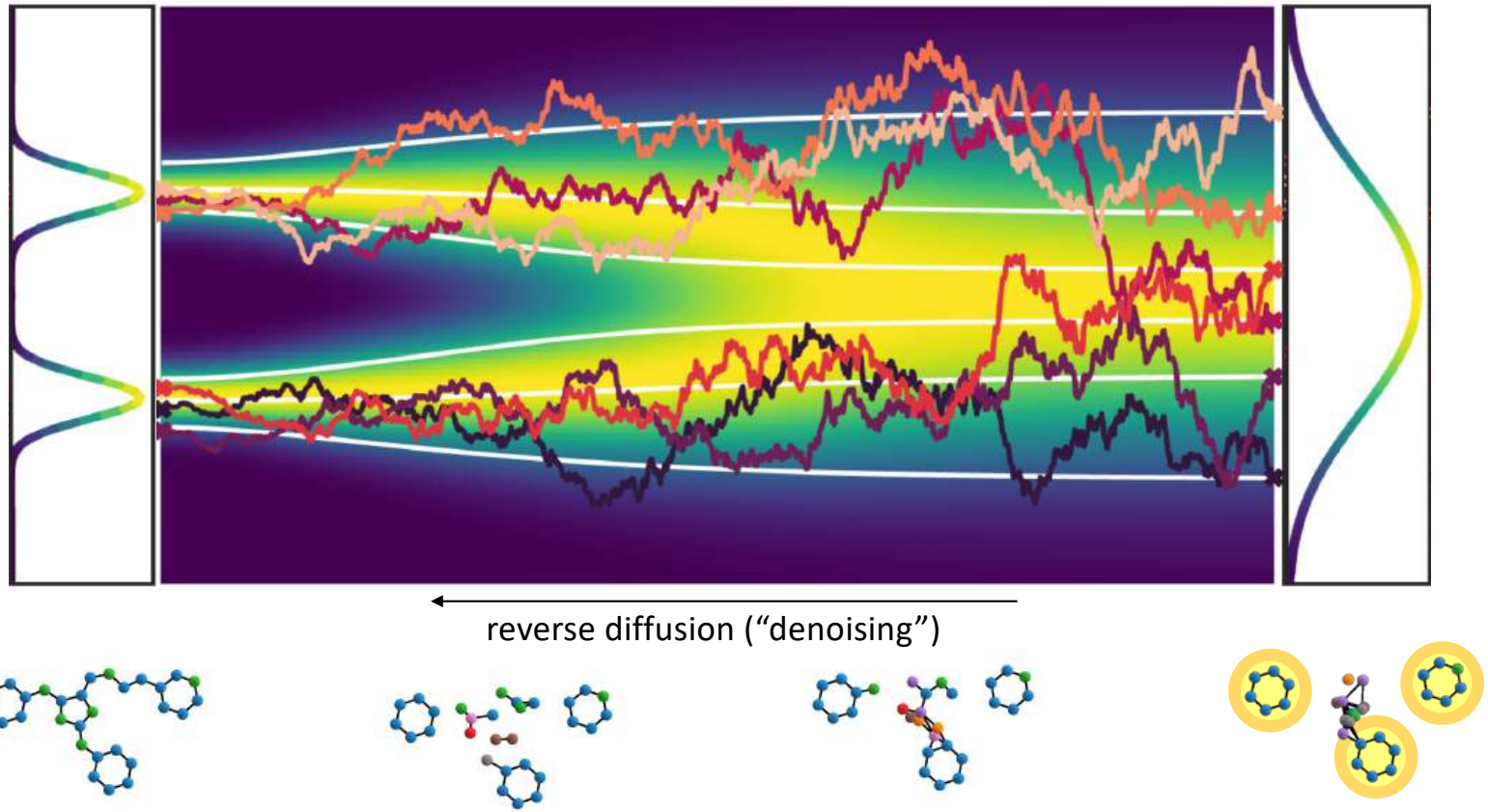
Yamamoto et al. 2022

Problem setup: "molecular impainting"



- Fragments placed in 3D space (known fixed orientation)
- Variable linker size
- Unknown attachment atoms (anchors)
- Variable number of fragments
- No clashes with protein pocket

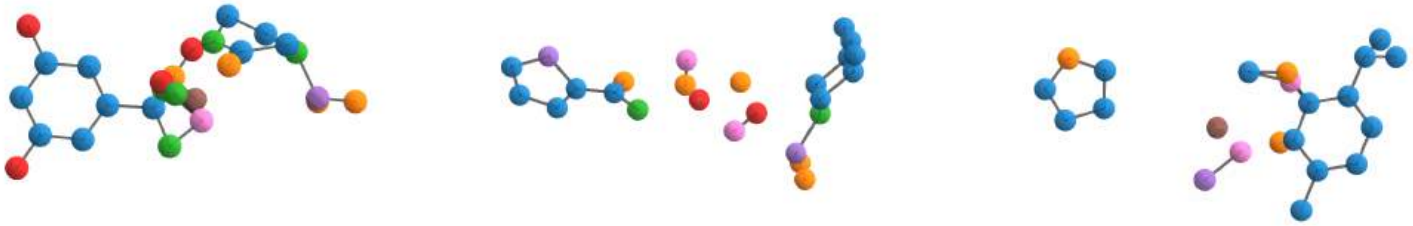
DiffLinker



Igashov, Stärk, Vignac et Welling, B, Correia 2022

Linking pairs of fragments

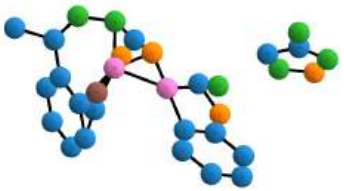
	Method	QED ↑	SA ↓	# Rings ↑	Valid, %	Unique, %	Novel, %
ZINC	DeLinker + ConfVAE + MMFF	0.64	3.11	0.21	**98.3**	44.2	**47.1**
	3DLinker (given anchors)	0.65	3.11	0.23	**99.3**	29.0	41.2
	3DLinker	0.65	3.14	0.24	71.5	29.2	41.9
	DiffLinker	**0.68**	**3.01**	0.25	93.8	24.0	30.3
	DiffLinker (given anchors)	**0.68**	**3.03**	0.26	97.6	22.7	32.4
	DiffLinker (sampled size)	0.65	3.19	**0.32**	90.6	**51.4**	42.9
	DiffLinker (given anchors, sampled size)	0.65	3.24	**0.36**	94.8	**50.9**	**47.7**
CASF	DeLinker + ConfVAE + MMFF	0.35	4.05	0.35	**95.7**	51.6	**55.6**
	DiffLinker	**0.41**	**4.00**	0.34	85.3	40.5	41.8
	DiffLinker (given anchors)	0.40	**4.03**	**0.38**	**90.2**	37.3	48.4
	DiffLinker (sampled size)	**0.40**	4.06	0.30	63.7	**60.0**	49.3
	DiffLinker (given anchors, sampled size)	0.40	4.10	**0.38**	68.4	**57.1**	**56.9**

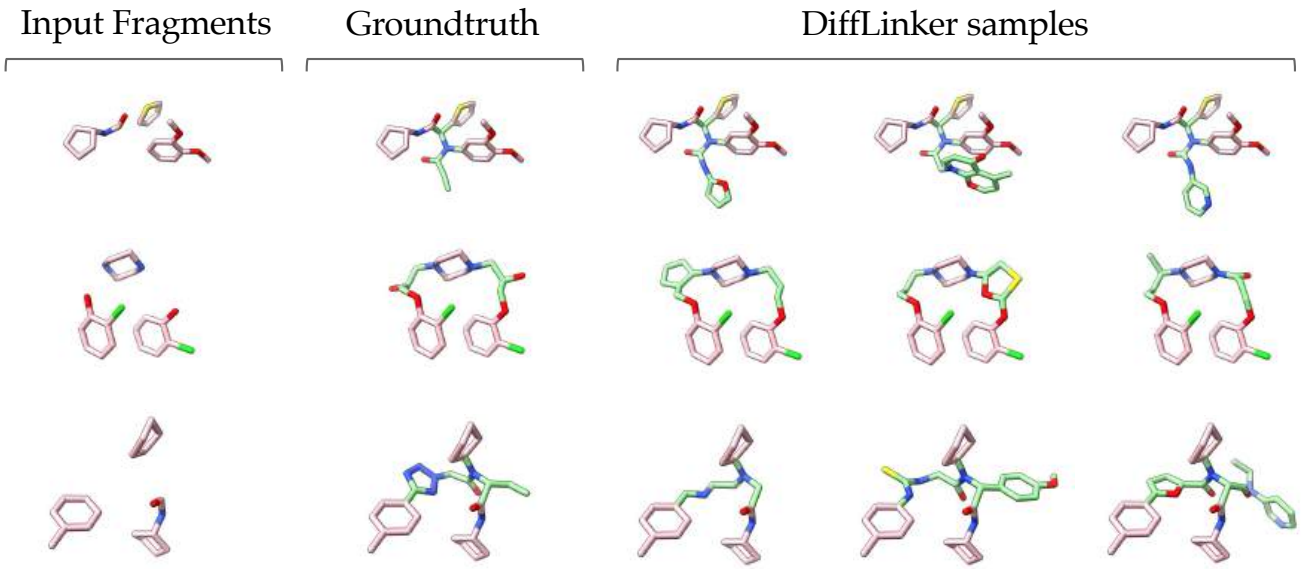


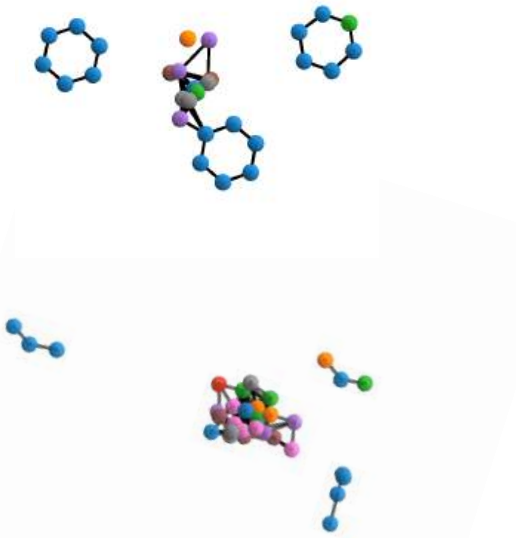
Igashov, Stärk, Vignac et Welling, B, Correia 2022

Linking multiple fragments

	Method	QED ↑	SA ↓	# Rings ↑	Valid, %	Unique, %	Novel, %
GEOM	3DLinker	0.36	3.56	0.00	16.3	**73.7**	—
	DiffLinker	**0.48**	**2.99**	0.75	**94.0**	34.7	68.6
	DiffLinker (given anchors)	**0.49**	**3.01**	**0.79**	**94.0**	35.0	68.4
	DiffLinker (sampled size)	0.45	3.27	0.75	87.2	62.4	**76.0**
	DiffLinker (given anchors, sampled size)	0.46	3.33	**0.83**	88.8	**63.6**	**76.0**



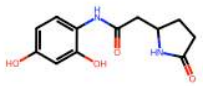




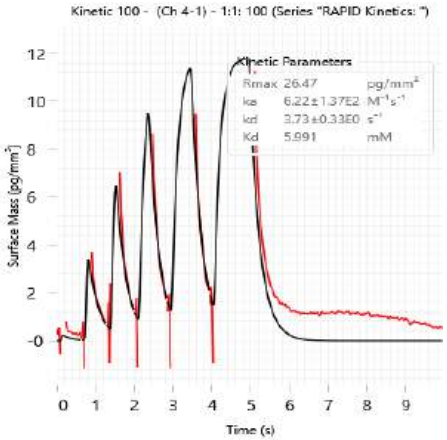
Igashov, Stärk, Vignac et Welling, B, Correia 2022

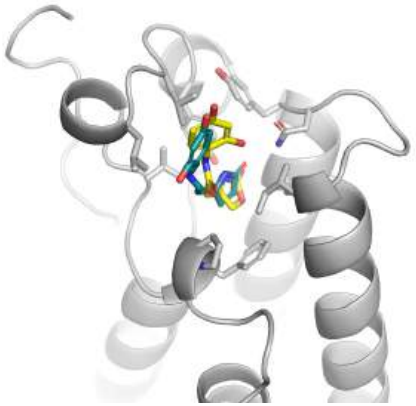
Experimental results: Binders

LogP 0.02

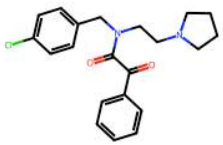


High solubility,
up to 5 mM

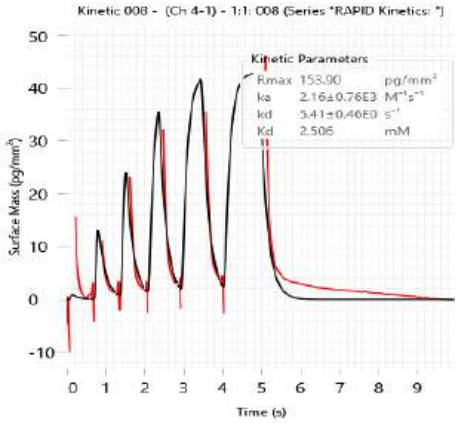


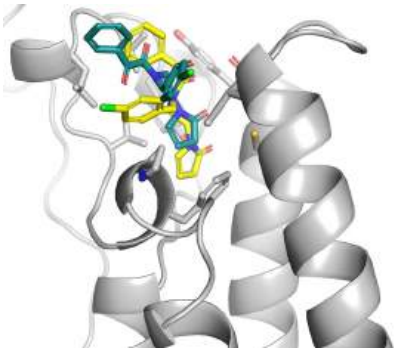


LogP 3.88



Soluble at 1 mM

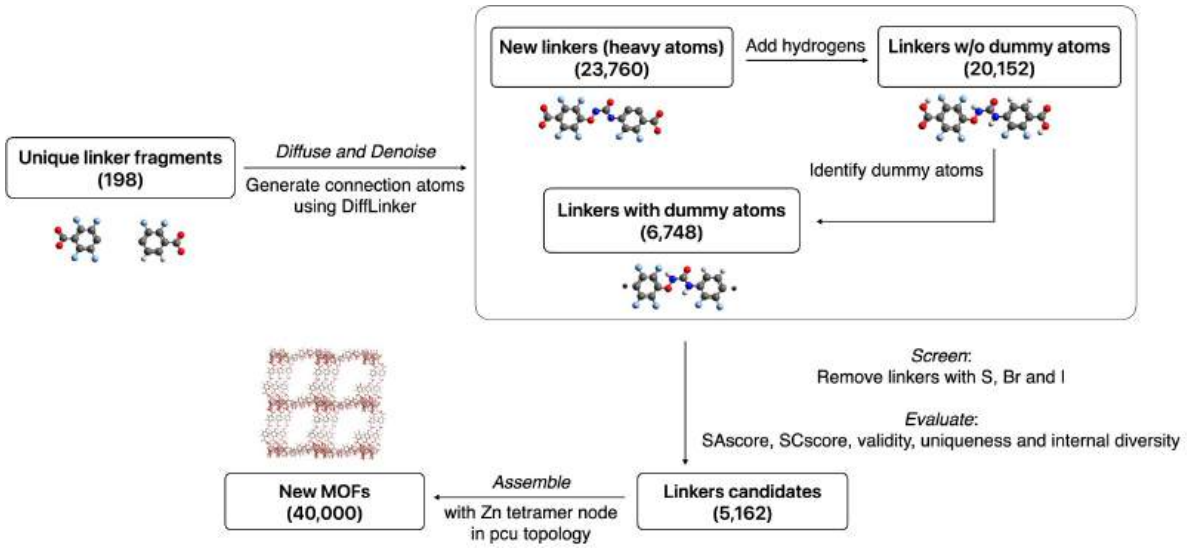




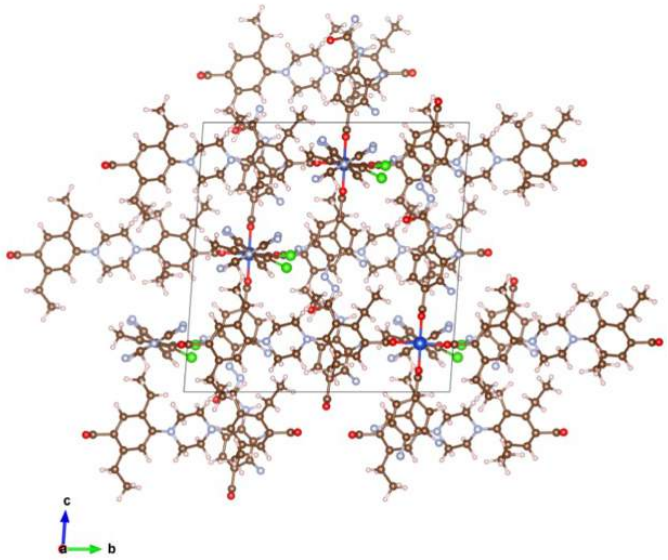
Bromodomain-containing protein 4 (BrD4)

Elizarova et al. 2024 (work in progress)

Unexpected Application: Material Design for Carbon capture



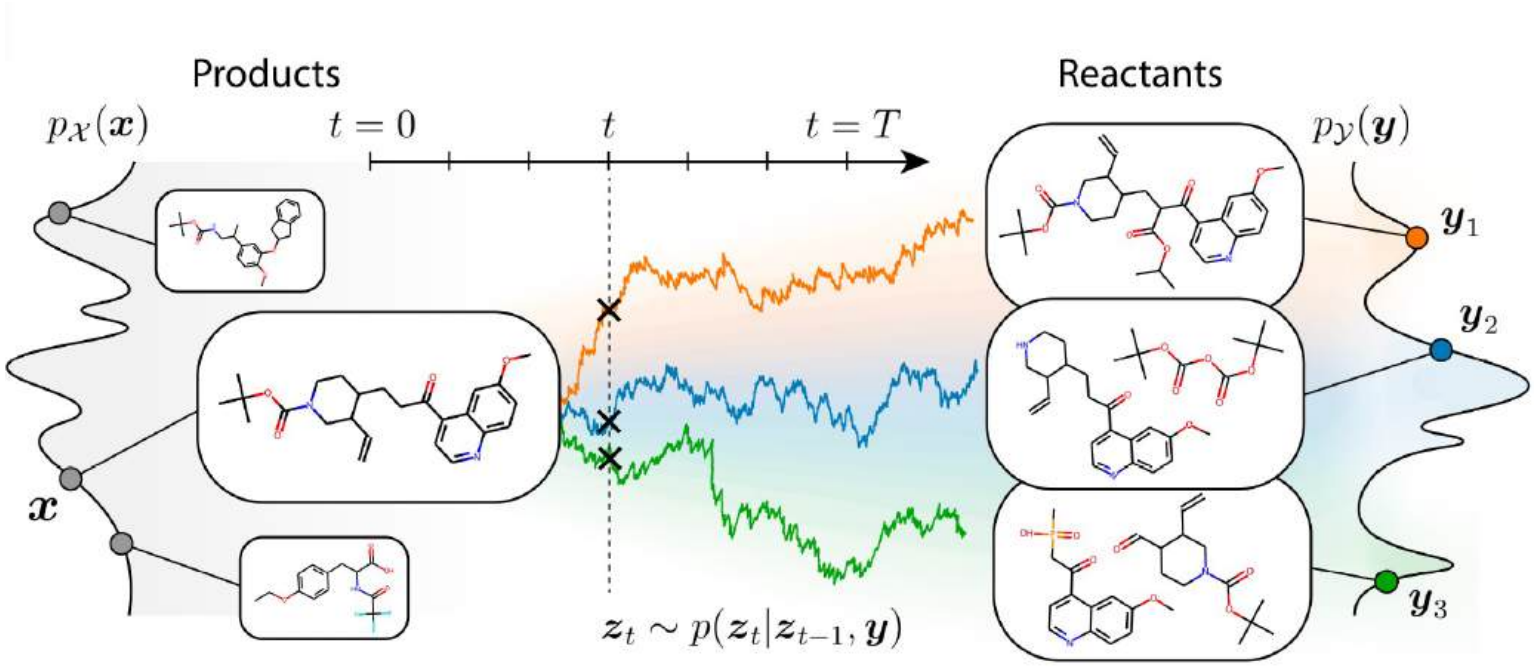
Generation of Metal-Organic Frameworks (MOF) for C capture



Example of generated MOF

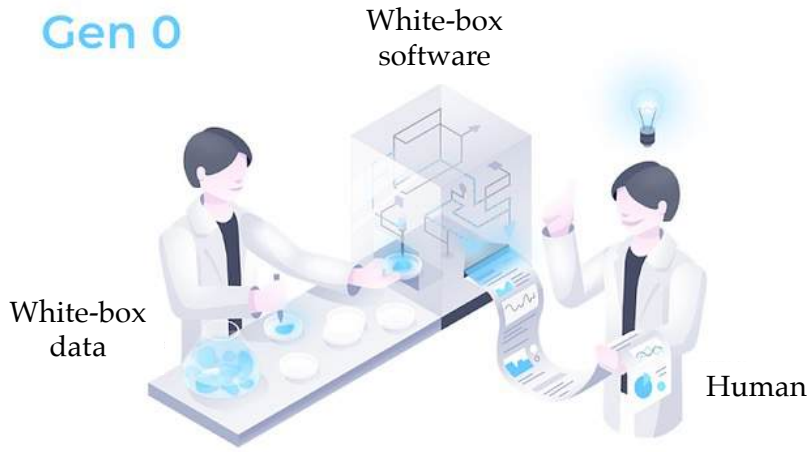
Park et al. 2024

Future directions: Synthesis prediction



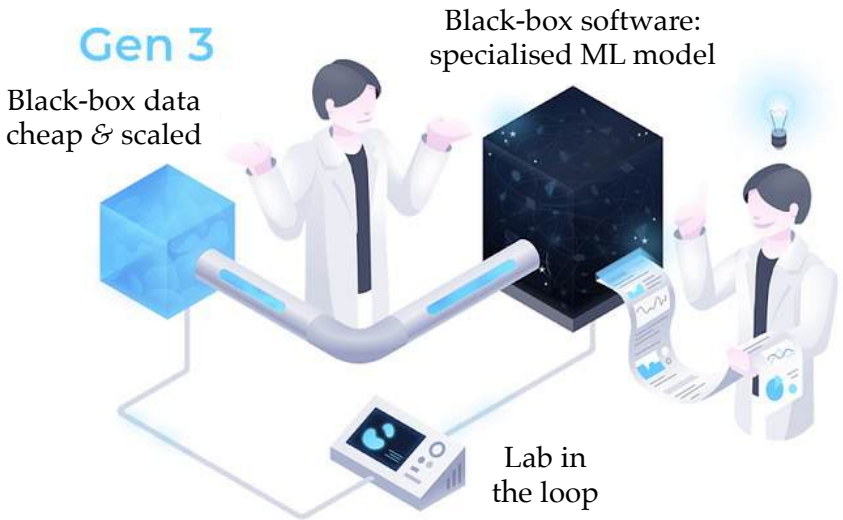
RetroBridge: generative model for retrosynthesis design

Igashov, Schneuing, Segler, B, Correia 2023


Gen 0
White-box software
White-box data
Human


Gen 1
Black-box software
White-box data


Gen 2
Black-box software
White-box data massively scaled


Gen 3
Black-box software: specialised ML model
Black-box data cheap & scaled
Lab in the loop

Naef, B 2024

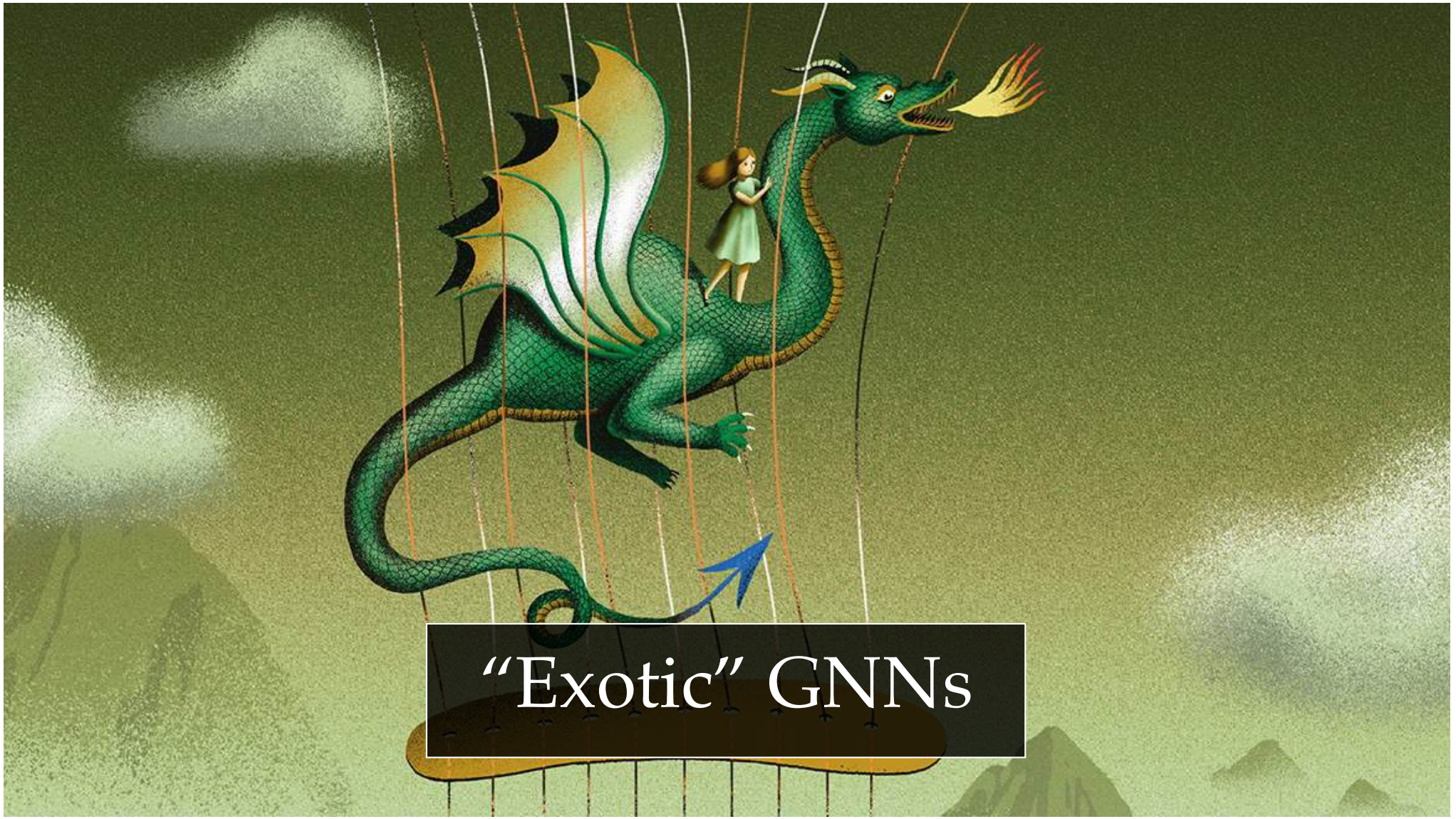
Conclusions

- Geometric deep learning is a powerful tool for molecular modelling
- Rootes in fundamental mathematical principles of invariance *&* symmetry
- Geometric generative models models
- Experimental validation!

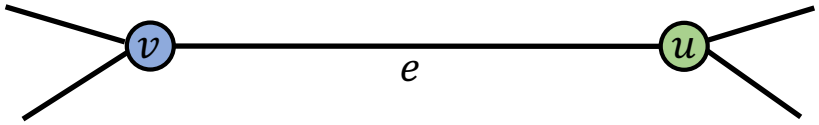
“The knowledge of certain principles easily compensates the lack of knowledge of certain facts”

—Claude Adrien Helvétius

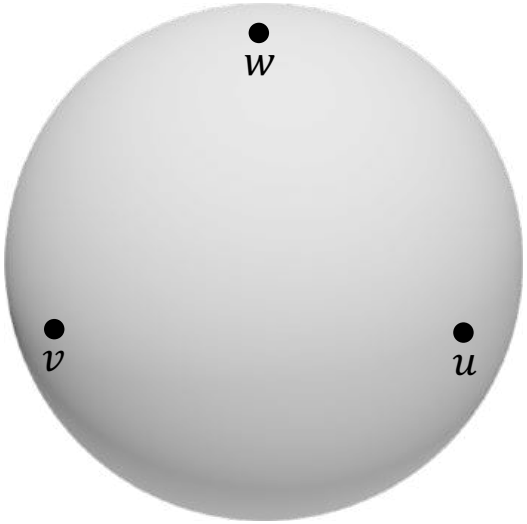

Thank you!


“Exotic” GNNs

Cellular Sheaves

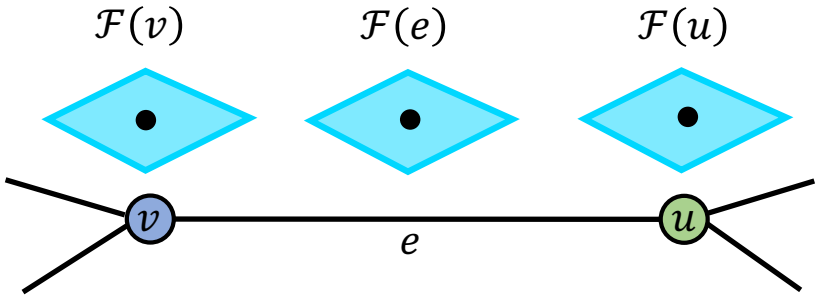


Graph

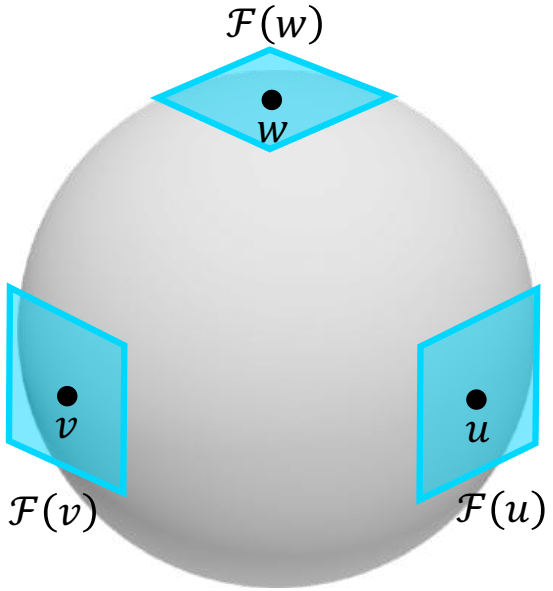


Manifold

Cellular Sheaves

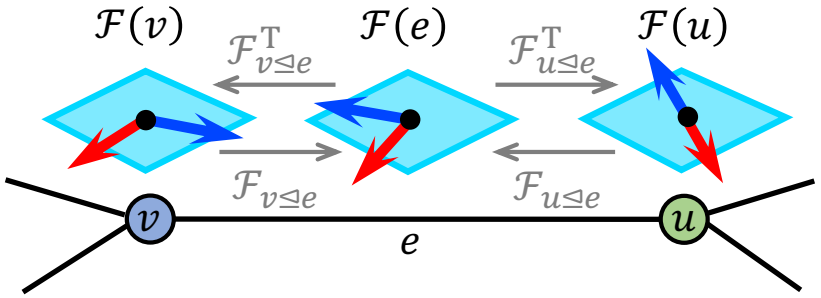


Cellular sheaf $\mathcal{F}$

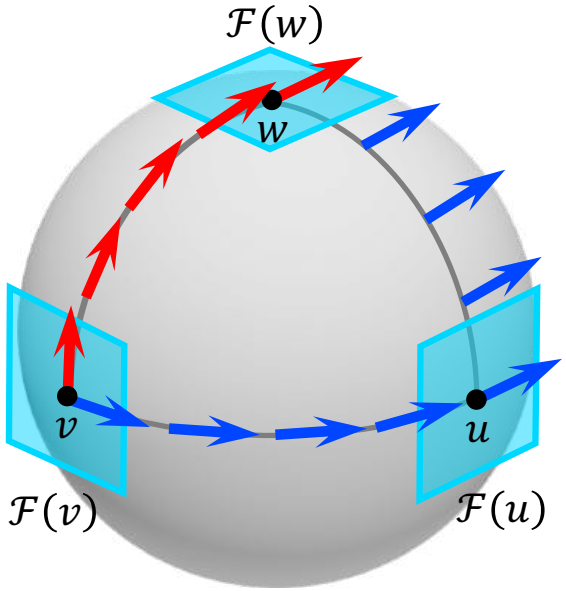


Manifold + Connection

Cellular Sheaves



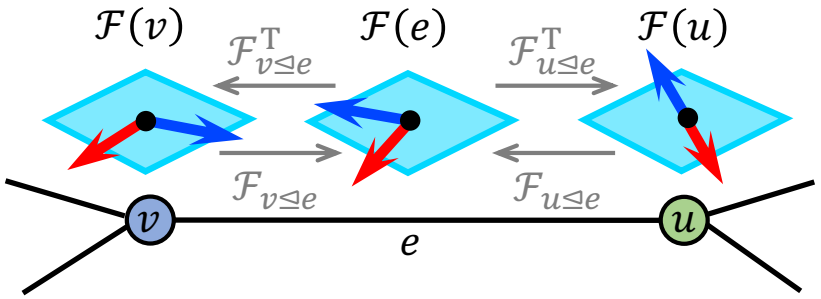
Cellular sheaf $\mathcal{F}$



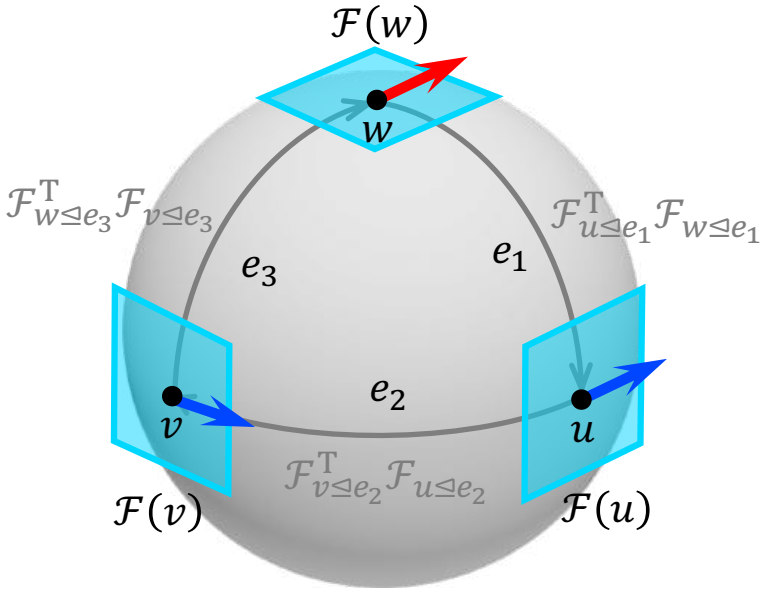
Manifold + Connection

Bodnar, di Giovanni, et B. 2022

Cellular Sheaves

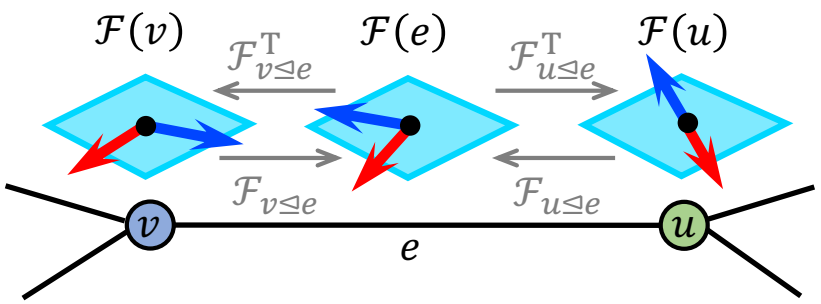


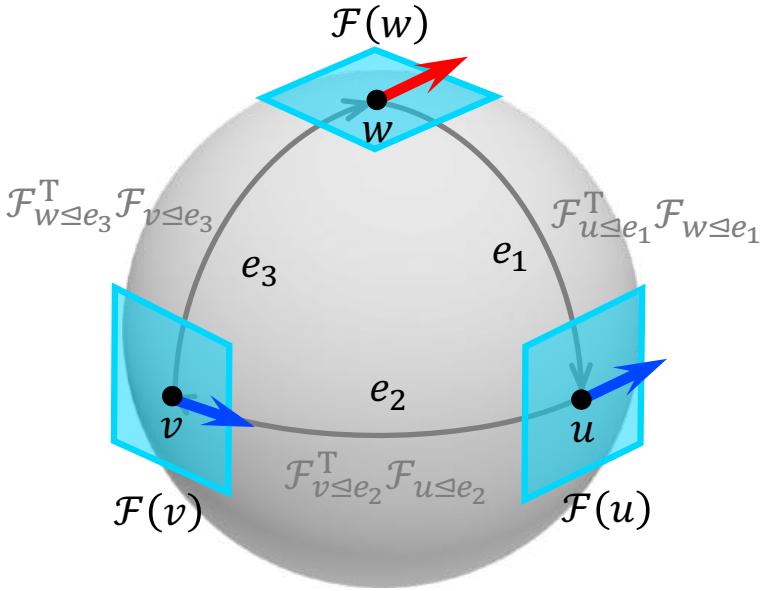
Cellular sheaf $\mathcal{F}$



Analogy to parallel transport on manifolds

Cellular Sheaves

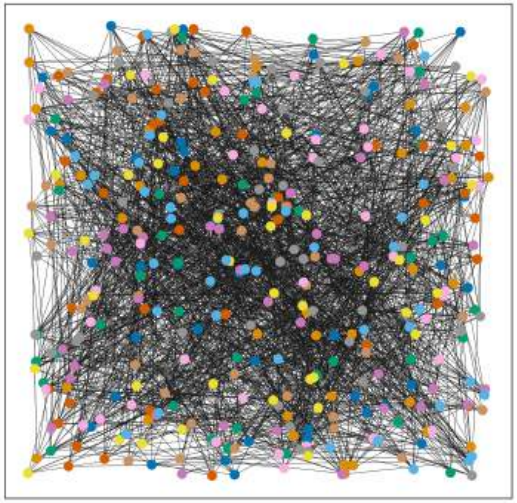




Endow graph with "geometry" leading to richer diffusion with better separation, ability to cope with heterophily, and no oversmoothing

Bodnar, di Giovanni, et B. 2022

Diffusion on Cellular Sheaves



$$\dot{\mathbf{X}}(t) = \Delta_{\mathcal{F}}\mathbf{X}(t) \text{ with i.c. } \mathbf{X}(0) = \mathbf{X}$$

Node classification = limit of sheaf diffusion equation with an appropriate sheaf

Bodnar, di Giovanni, et B. 2022

Alternative to Weisfeiler-Lehman for expressive power?

Graph type	**# Node classes**	**Sheaf class $\mathcal{F}$, dim=d**	
Homophilic	2	Symmetric d=1	✓
Heterophilic	2	Symmetric d=1	✗
	2	Non-symmetric d=1	✓
	≥3	Non-symmetric d=1	✗
	≤2d	Orthogonal, d={2,4}	✓

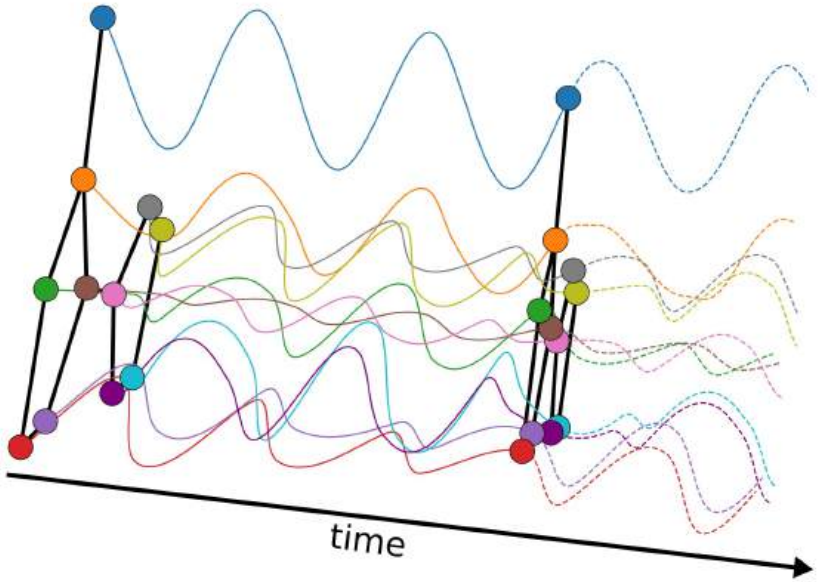
The capability of sheaf diffusion to solve node classification problem in the limit

Bodnar, di Giovanni, et B. 2022

What do we gain from physics-inspired GNNs?

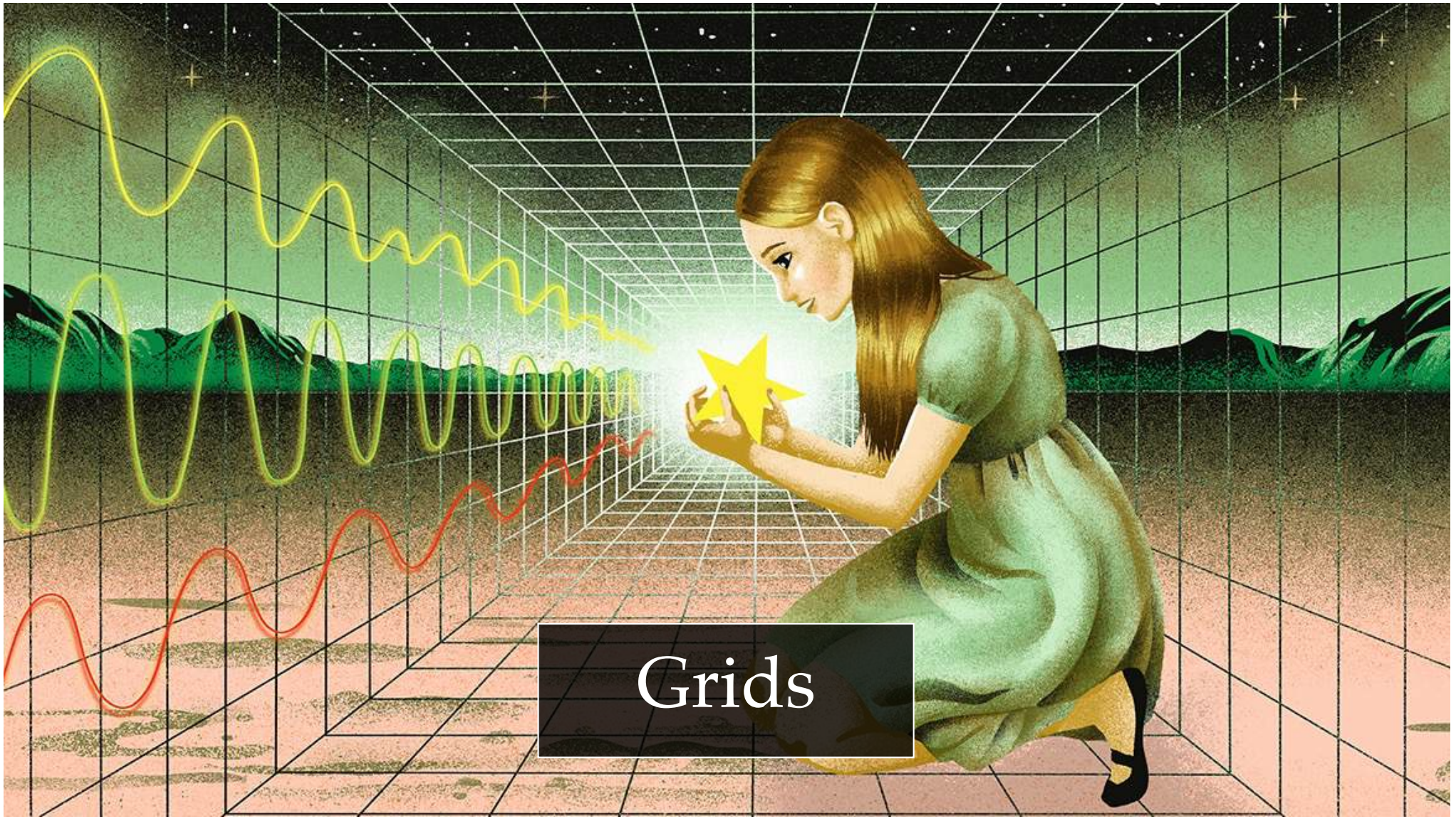
- New perspectives on old problems (e.g. oversmoothing, bottlenecks, etc.)
- New architectures
  - Many GNNs can be formalised as a discretised Graph Diffusion equation
  - More efficient solvers (multistep, adaptive, implicit, multigrid, etc.)
  - Implicit schemes = multi-hop filters
- Principled architectural 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 and algebraic topology)
- Other physical models

Graph-Coupled Oscillators

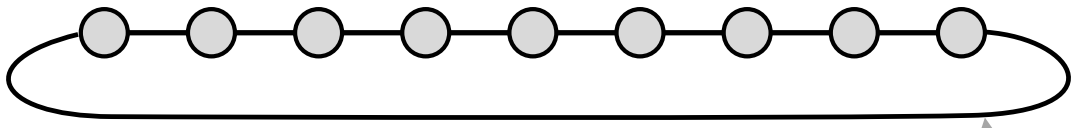


Dynamics of a system of coupled oscillators on a molecular graph

Rusch et B. 2022


Grids

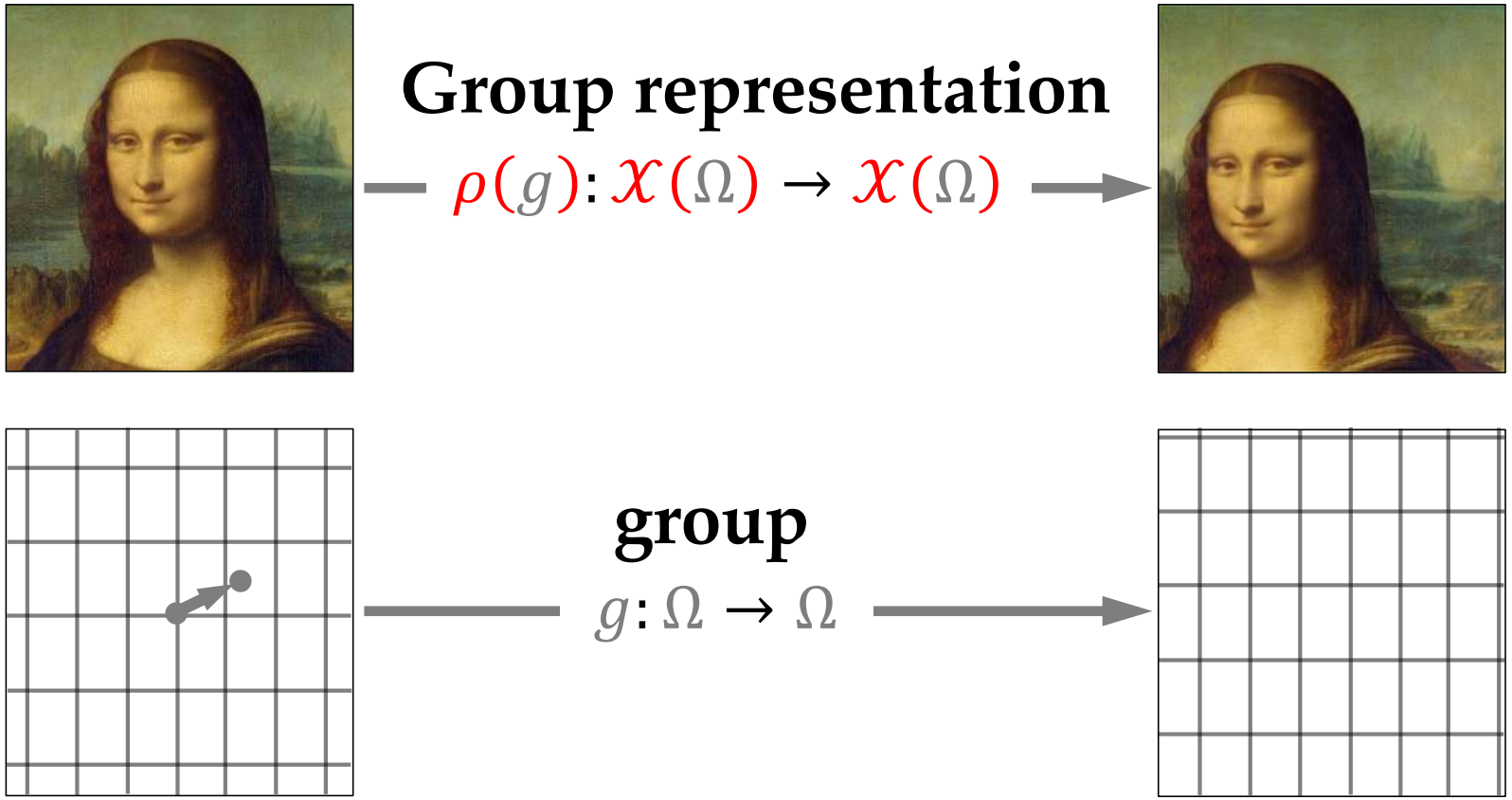
Grids vs Graphs



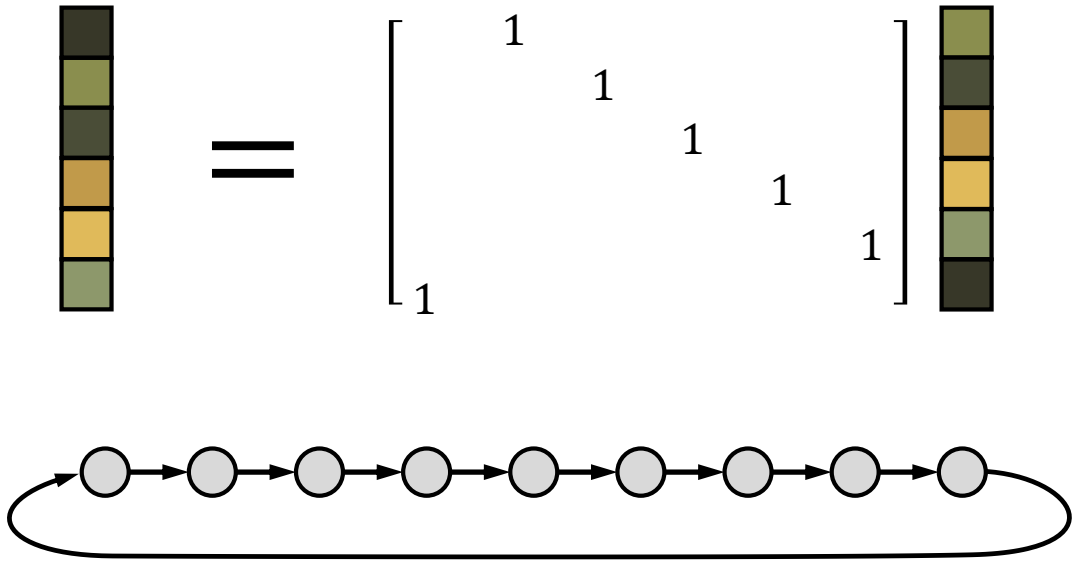
grid = ring graph

periodic b.c.

"Lifting"

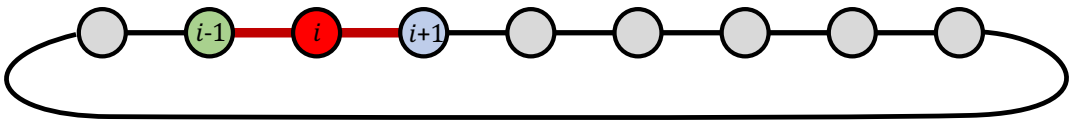

Group representation
$\rho(g): \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)$
group
$g: \Omega \to \Omega$

Shift operator



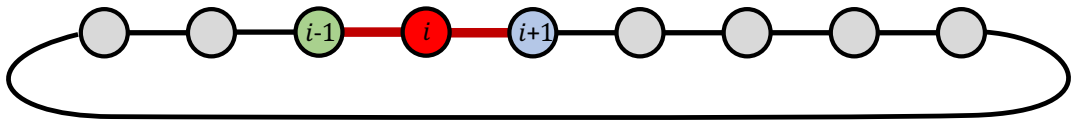
Note: Observe that the adjacency matrix of the directed ring graph is exactly the shift operator. We will use it when defining graph convolutions.

Grids vs Graphs



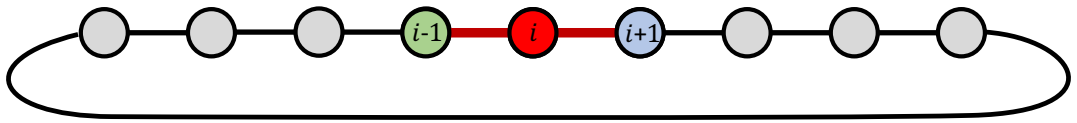
fixed neighbourhood structure

Grids vs Graphs



fixed neighbourhood structure

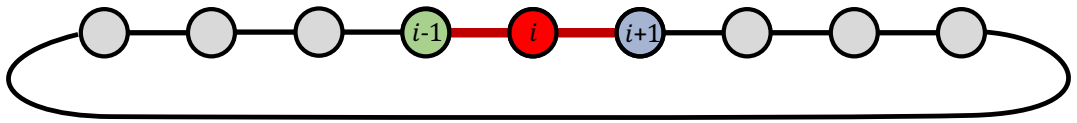
Grids vs Graphs



local aggregation function

$$f(\mathbf{x}_i) = \phi(\mathbf{x}_i, \{\mathbf{x}_{i-1}, \mathbf{x}_{i+1}\})$$

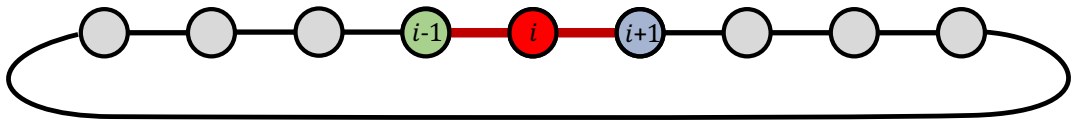
Grids vs Graphs



local aggregation function

$$f(\mathbf{x}_i) = \phi(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1})$$

Grids vs Graphs



linear local aggregation function

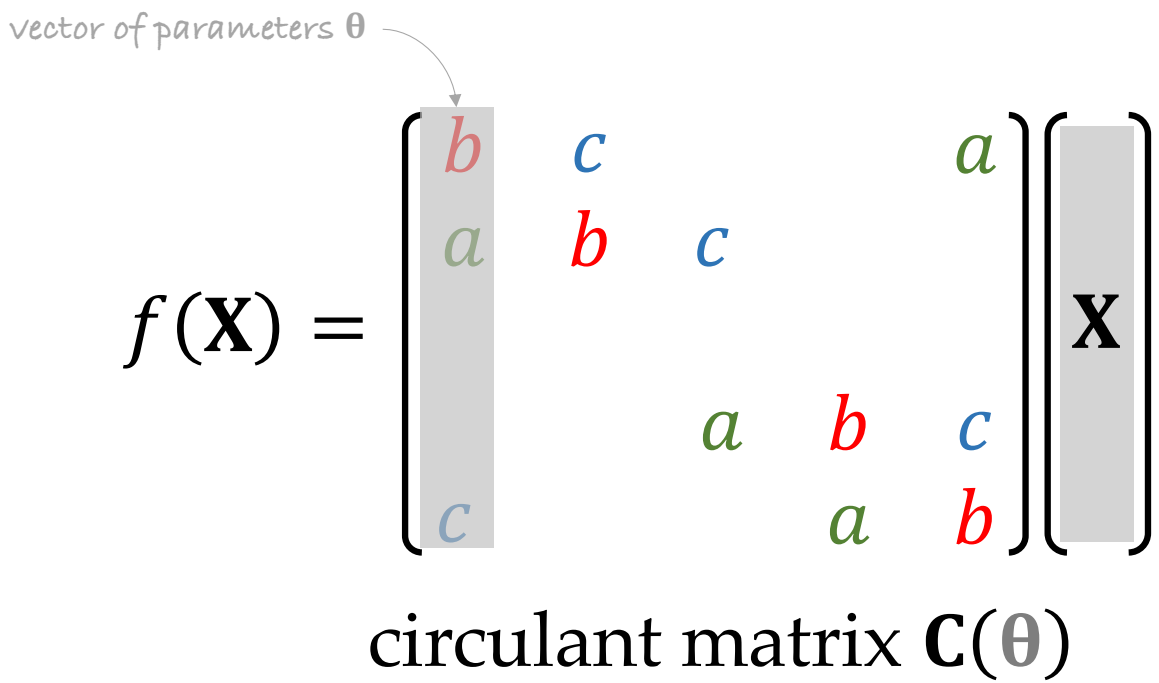
$$f(\mathbf{x}_i) = a\mathbf{x}_{i-1} + b\mathbf{x}_i + c\mathbf{x}_{i+1}$$

Convolution

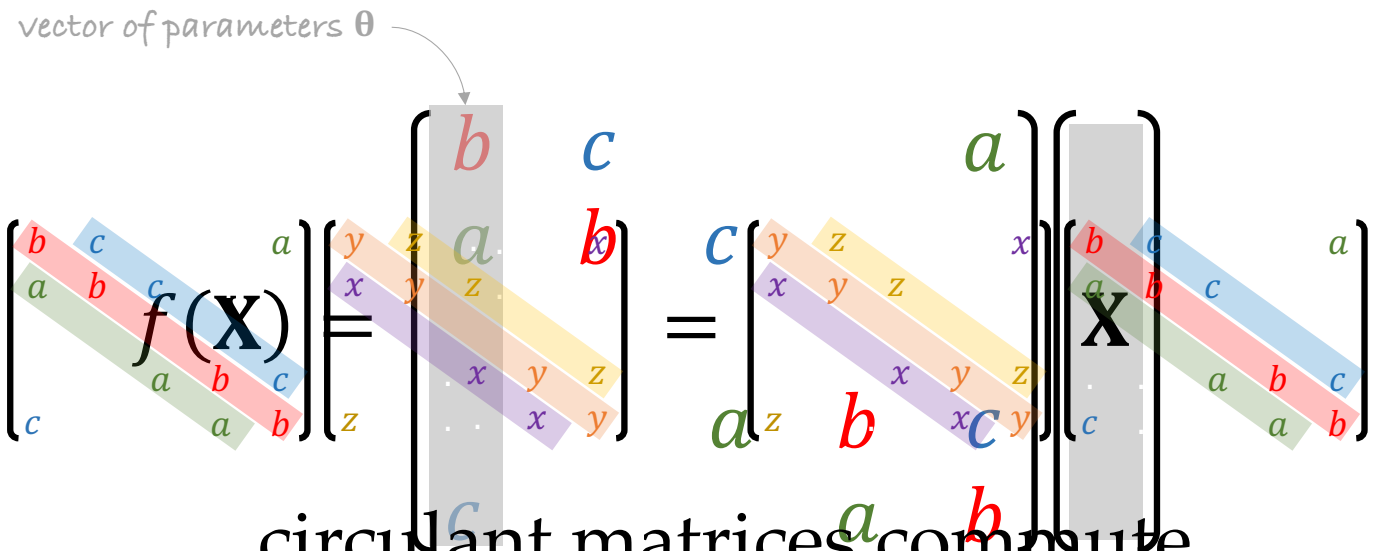
$$f(\mathbf{X}) = \begin{bmatrix} b & c & & & a \\ a & b & c & & \\ & a & b & c & \\ & & a & b & \\ c & & & a & b \end{bmatrix} \begin{bmatrix} \\ \mathbf{X} \\ \\ \\ \\ \end{bmatrix}$$

circulant matrix = convolution

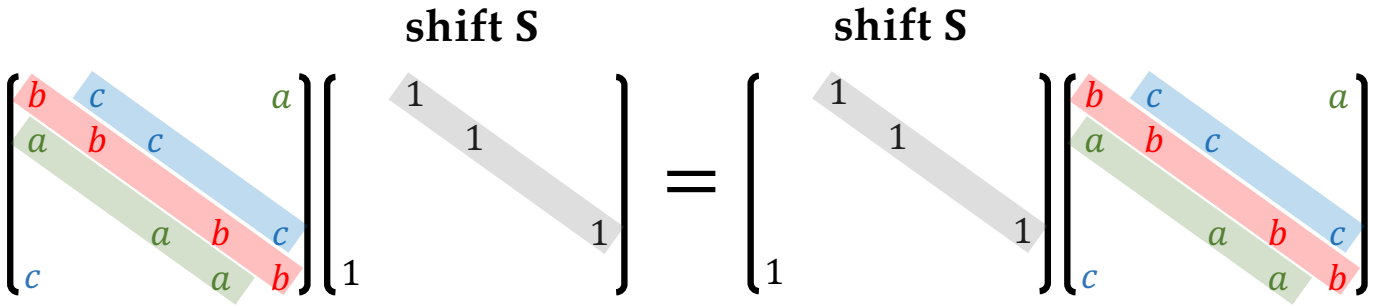
Convolution



Deriving Convolution from Symmetry

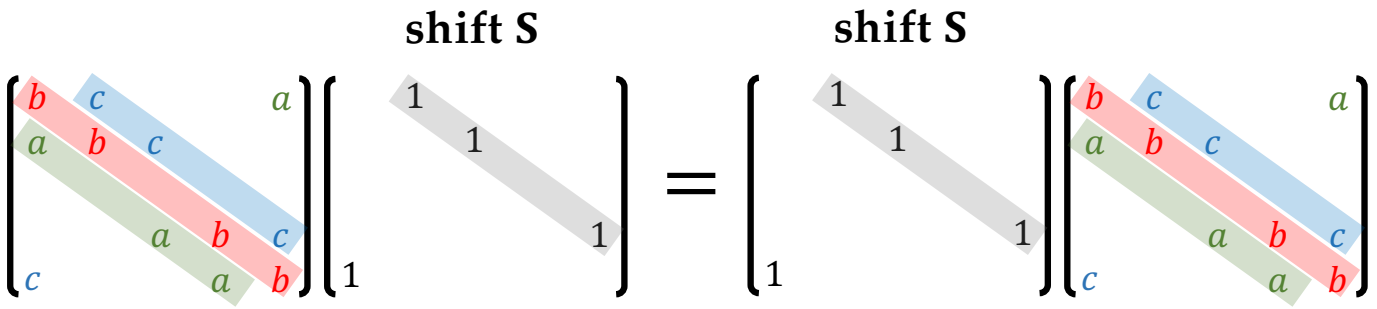


Deriving Convolution from Symmetry



circulant matrix $\Longrightarrow$ commutes with shift

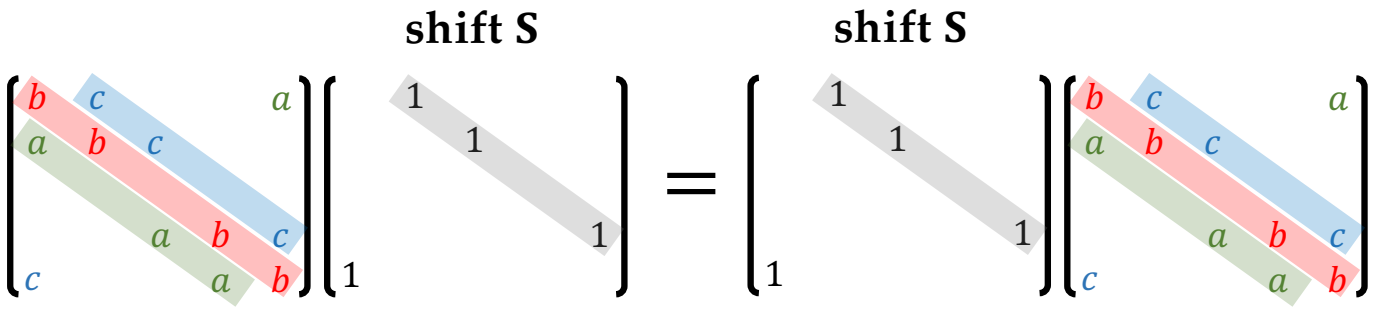
Deriving Convolution from Symmetry



convolution $\Longrightarrow$ shift-equivariant

CS = SC

Deriving Convolution from Symmetry

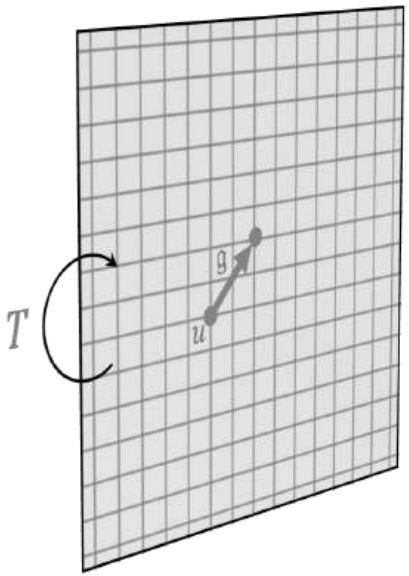


convolution $\Leftrightarrow$ shift-equivariant

convolution **emerges** from translation symmetry

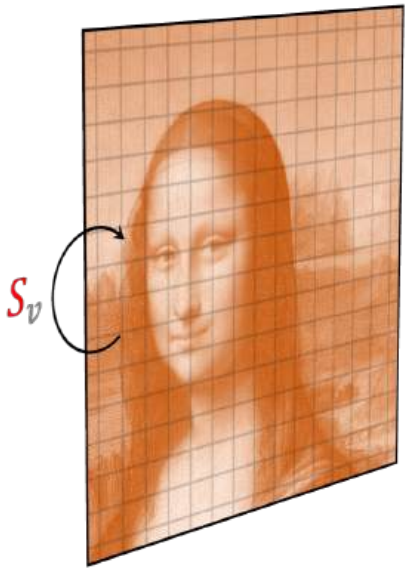
CNNs as an Instance of Geometric Deep Learning Blueprint

Plane $\mathbb{R}^2$



Translation group $T(2)$

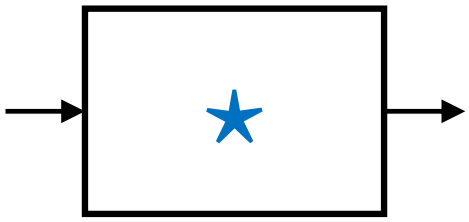
images $\mathcal{X}(\mathbb{R}^2)$



Shift operator S

$S_v x(u) = x(u - v)$

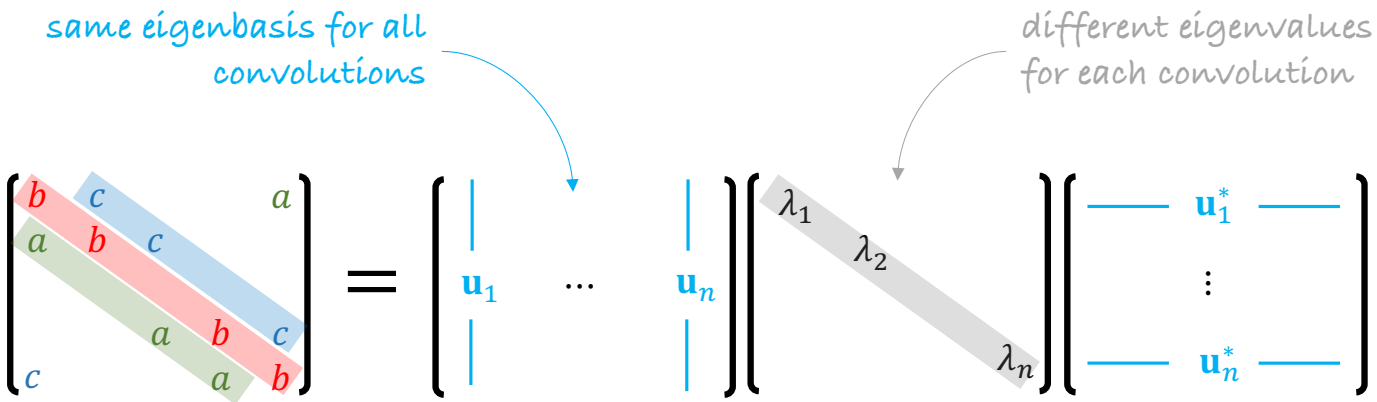
functions $\mathcal{F}(\mathcal{X}(\Omega))$



Convolutional layer

$(Sx \star y) = S(x \star y)$

Deriving Fourier Transform from Symmetry

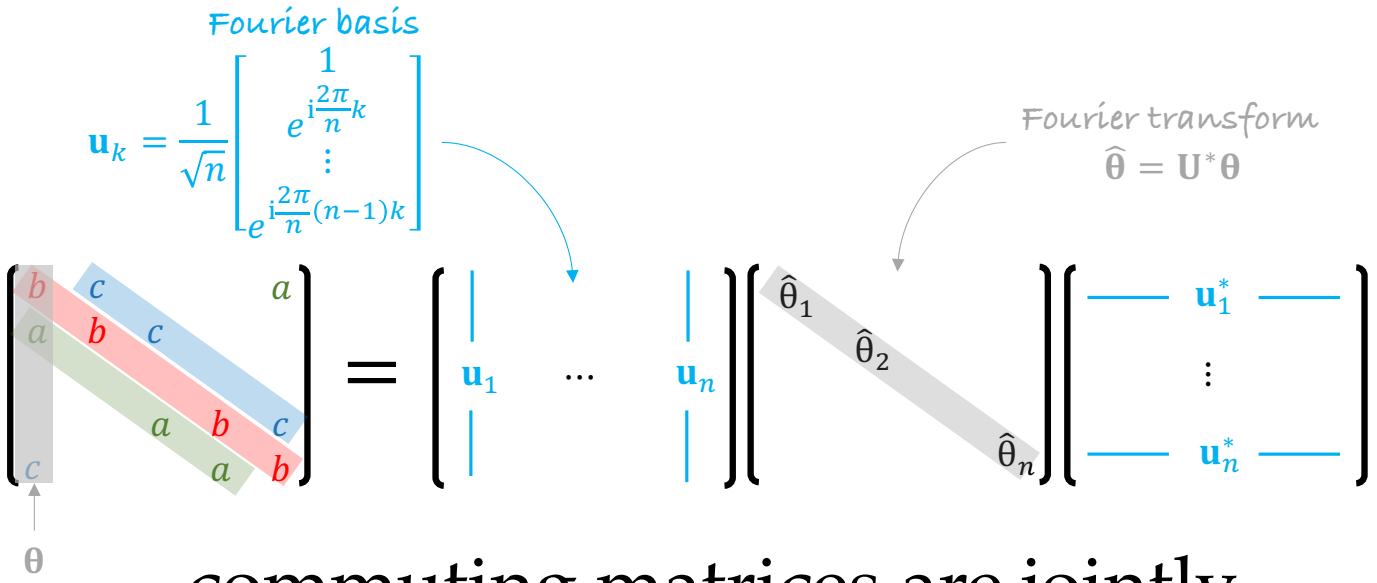


commuting matrices are jointly diagonalisable

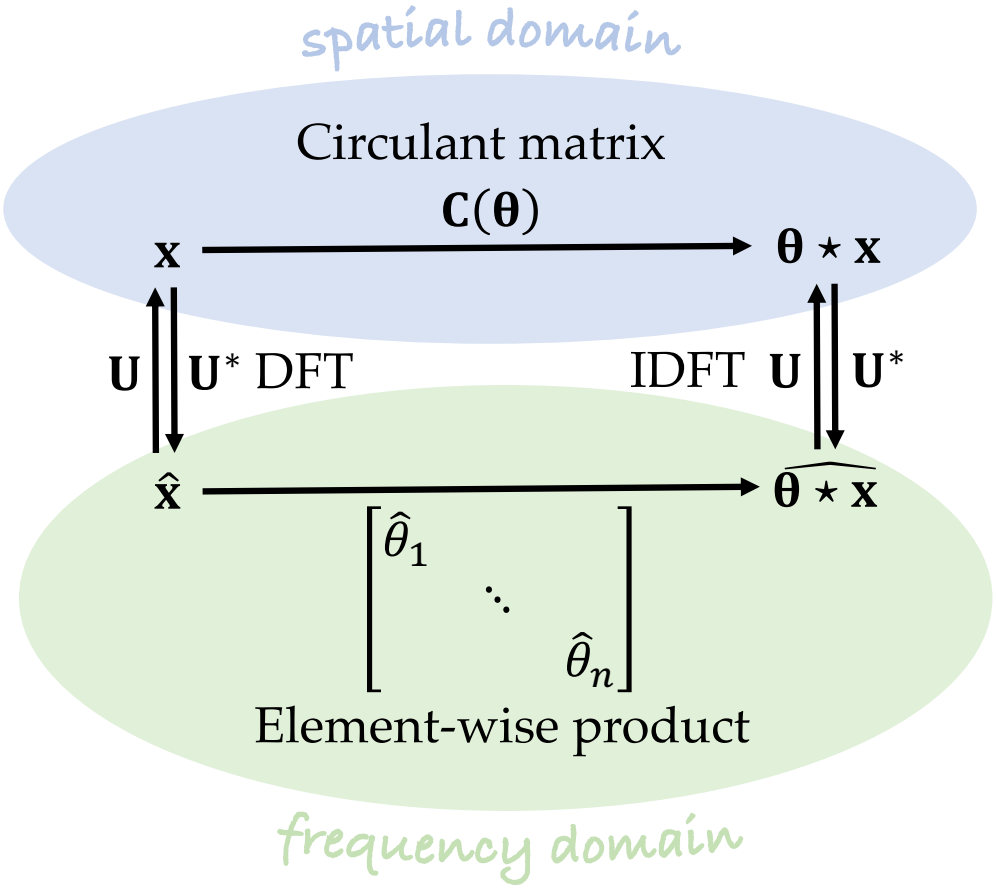
jointly diagonalisable

eigenvectors of **S**

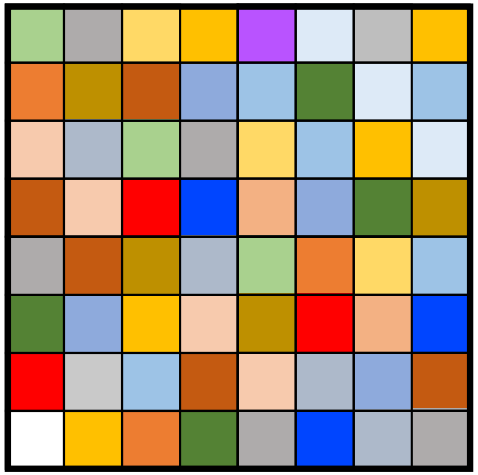
Deriving Fourier Transform from Symmetry



commuting matrices are jointly diagonalisable by Fourier Transform


spatial domain
Circulant matrix
$\mathbf{C}(\boldsymbol{\theta})$
$\mathbf{x}$
$\boldsymbol{\theta} \star \mathbf{x}$
$\mathbf{U}$
$\mathbf{U}^*$ DFT
IDFT $\mathbf{U}$
$\mathbf{U}^*$
$\hat{\mathbf{x}}$
$\widehat{\boldsymbol{\theta} \star \mathbf{x}}$
$\begin{bmatrix} \hat{\theta}_1 & & \\ & \ddots & \\ & & \hat{\theta}_n \end{bmatrix}$
Element-wise product
frequency domain

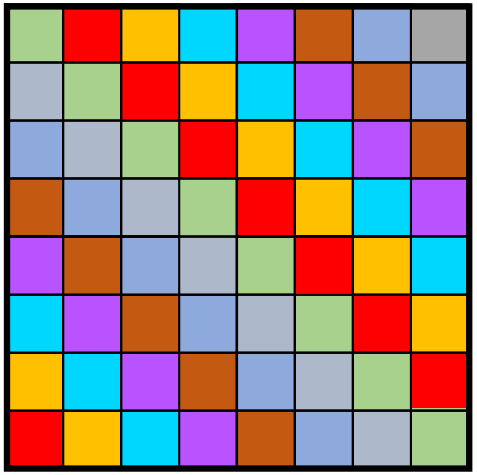
Fully connected



Trivial $G = \{e\}$

$\mathcal{O}(n^2)$ DOF

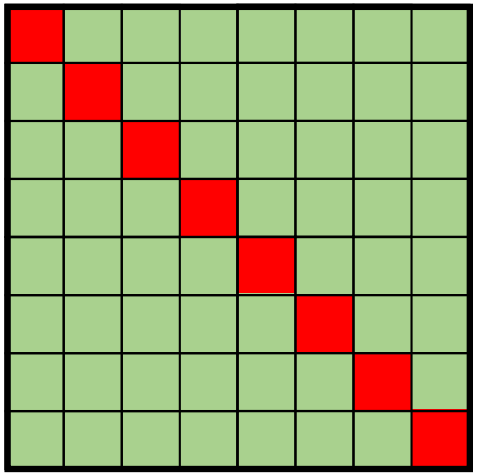
Grids



Translation

$\mathcal{O}(n)$ DOF

Graphs / Sets

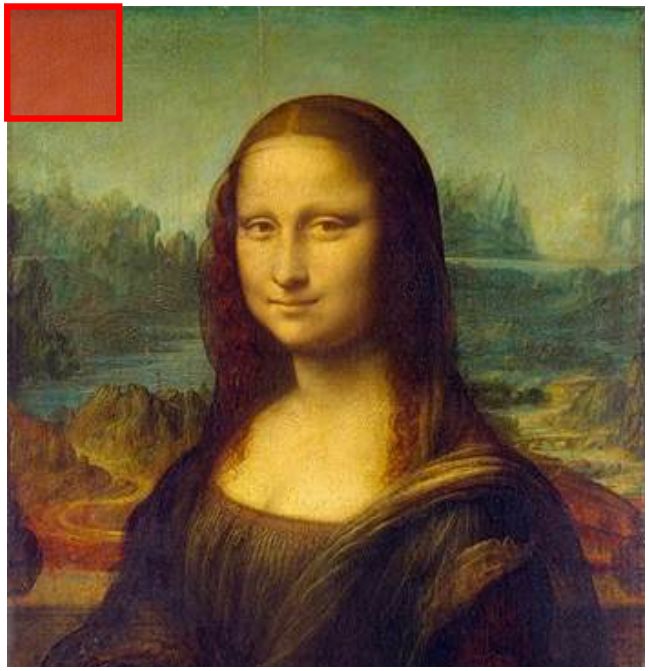


Permutation

$\mathcal{O}(1)$ DOF


Groups

Convolution, revisited



Convolution, revisited

convolution = matching shifted filter

$$(x \star \psi)(u) = \langle x, S_u \psi \rangle = \int_{-\infty}^{+\infty} x(v)\psi(u - v)\mathrm{d}v$$

shift vector

shift operator

domain $\Omega \cong$ symmetry group G

i.e., $\star\psi \colon \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)$

Group Convolution

convolution = matching transformed filter

$$(x \star \psi)(g) = \langle x, \rho(g)\psi \rangle = \int_{\Omega} x(v)\psi(g^{-1}v)\mathrm{d}v$$

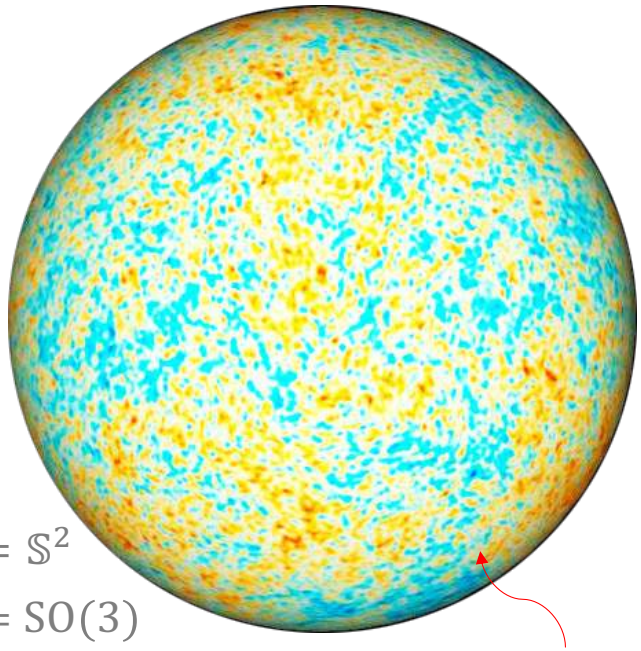
group element

group representation

The convolution outputs a signal on the group G

i.e., $\star\psi: \mathcal{X}(\Omega) \to \mathcal{X}(G)$

Convolution on the Sphere



sphere $\Omega = \mathbb{S}^2$

rotation group $\mathfrak{G} = \mathrm{SO}(3)$

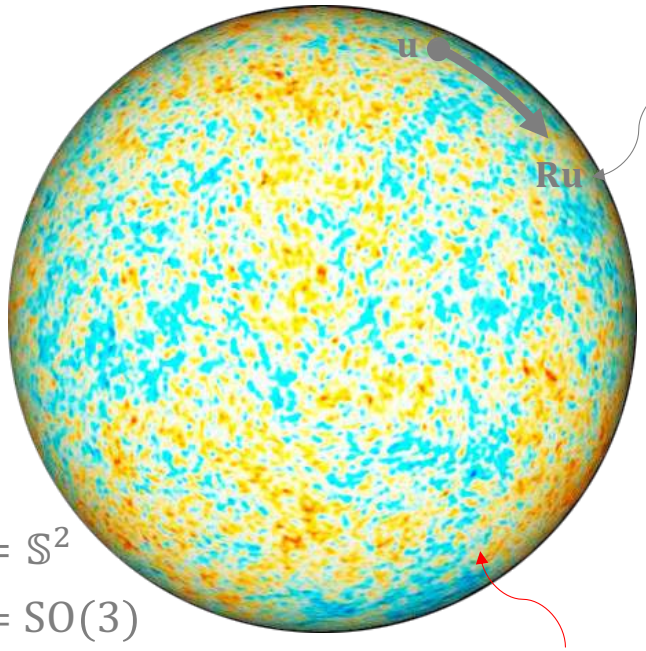
spherical signal x

Cohen, Welling 2016

Convolution on the Sphere

$$(x \star \psi)(\mathbf{R}) = \int_{\mathbb{S}^2} x(\mathbf{u})\psi(\mathbf{R}^{-1}\mathbf{u})\mathrm{d}\mathbf{u}$$

signal on $\mathrm{SO}(3)$

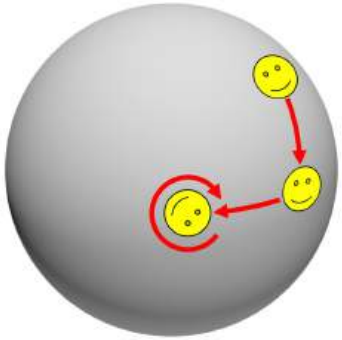


rotation transformation

sphere $\Omega = \mathbb{S}^2$

rotation group $\mathfrak{G} = \mathrm{SO}(3)$

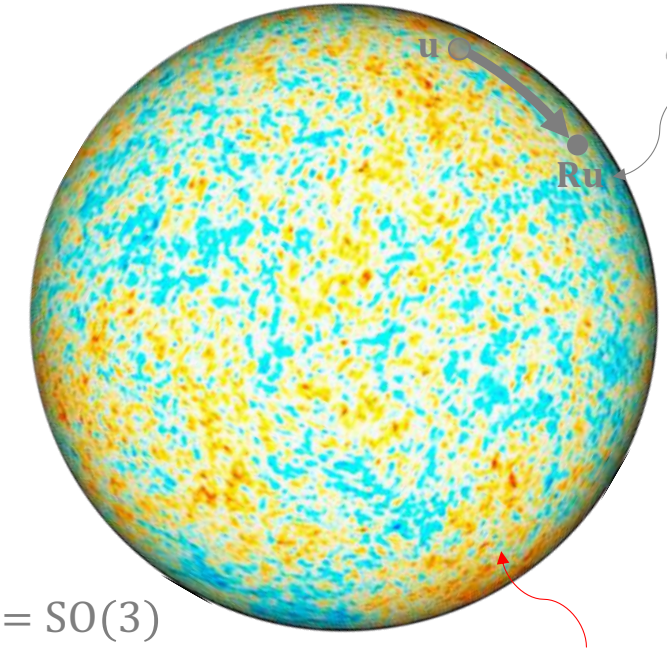
spherical signal x



Cohen, Welling 2016

Convolution on the Sphere

$$\big((x \star \psi) \star \phi\big)(\mathbf{R}) = \int_{\mathrm{SO}(3)} (x \star \psi)(\mathbf{Q})\,\phi(\mathbf{R}^{-1}\mathbf{Q})\,\mathrm{d}\mathbf{Q}$$



convolution on SO(3)

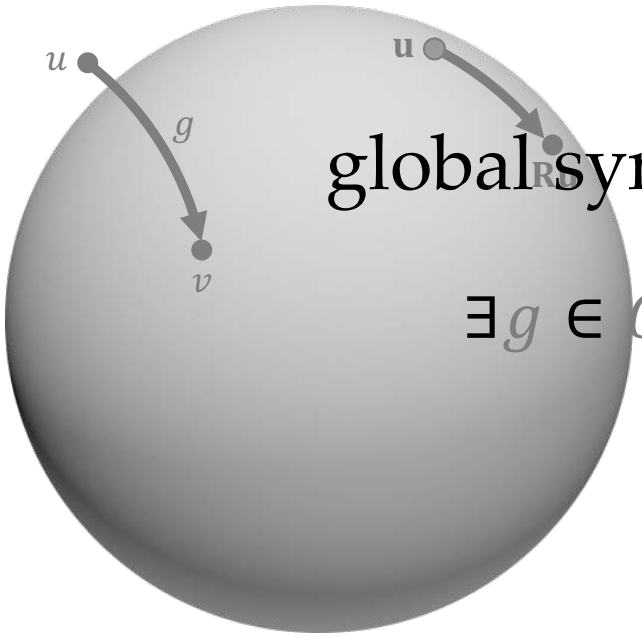
rotation transformation

$\Omega = \mathfrak{G} = \mathrm{SO}(3)$

spherical signal x

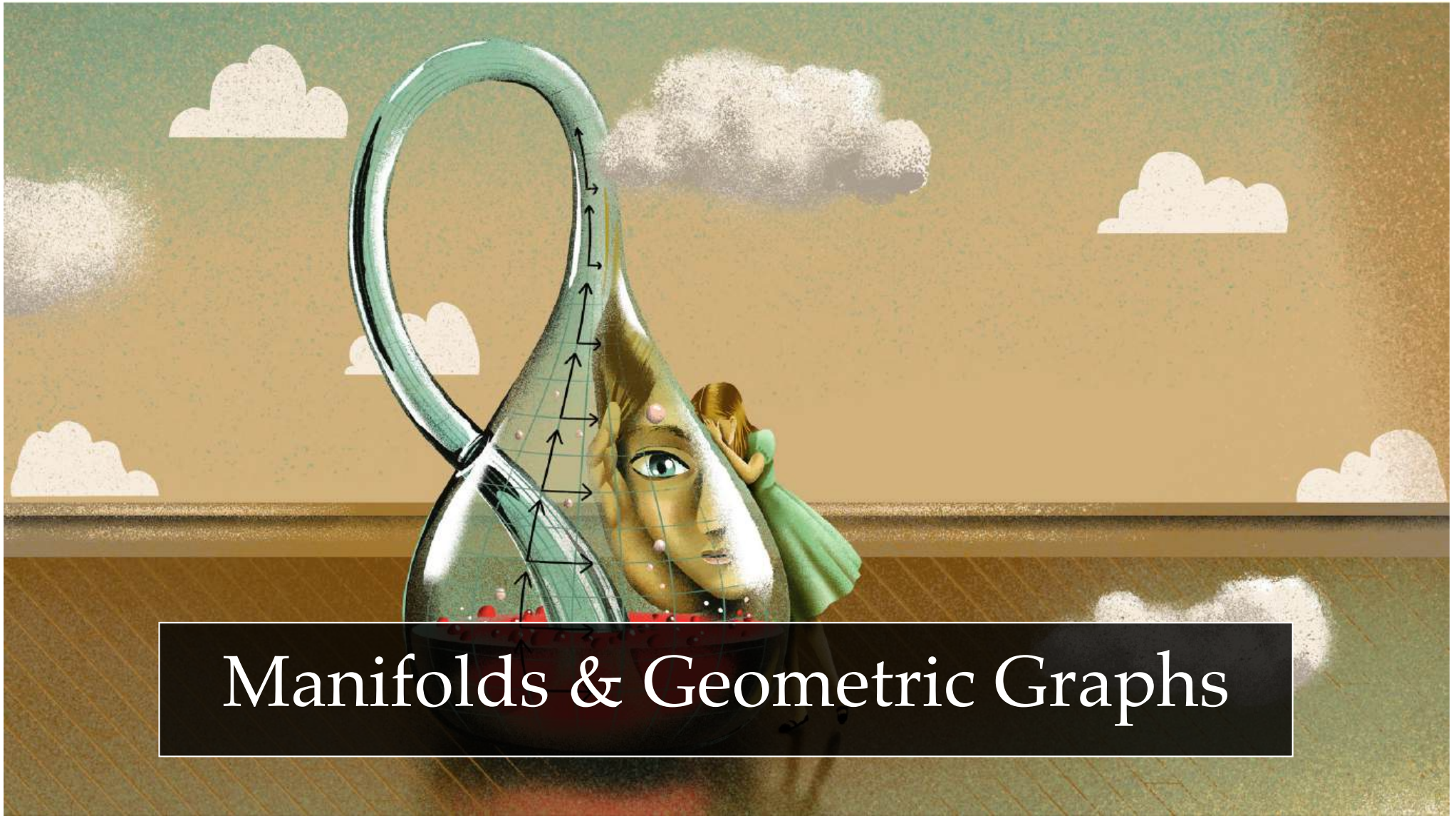
Cohen, Welling 2016

Homogeneous Spaces



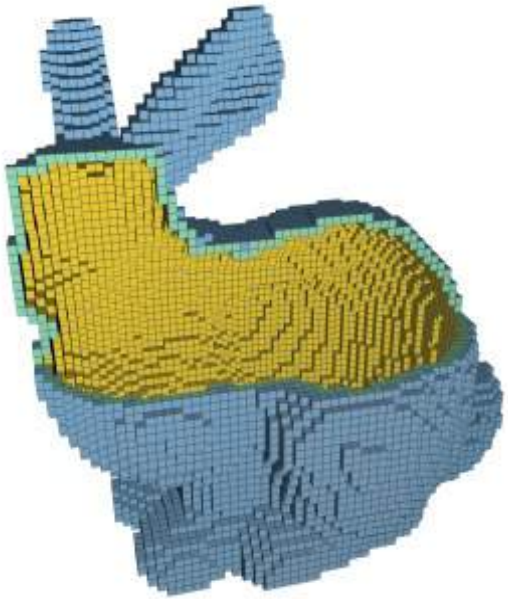
global symmetry group

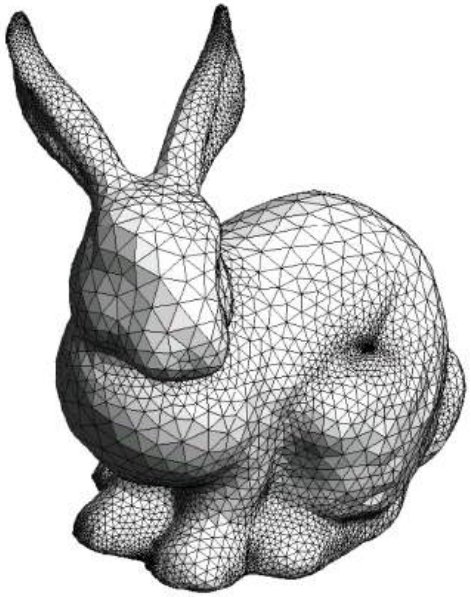
$\exists g \in G$ s.t. $gu = v$



Manifolds & Geometric Graphs

Why Manifolds?





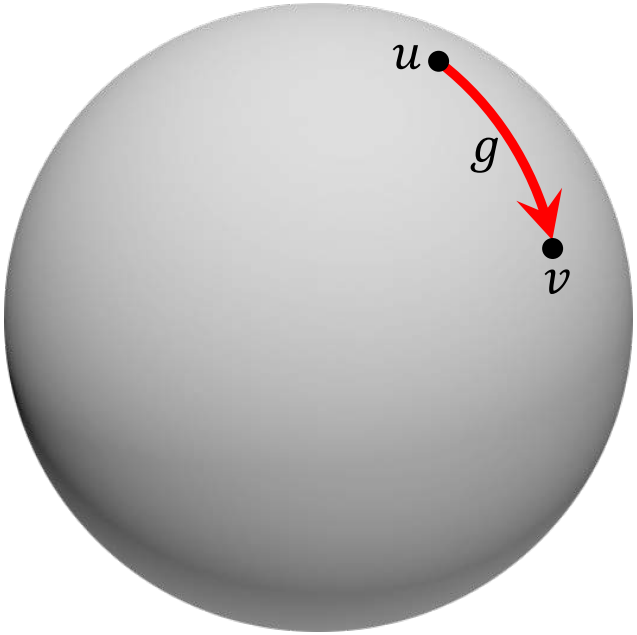
More efficient representation: no “waste” for internal structures

Natural model for deformable shapes

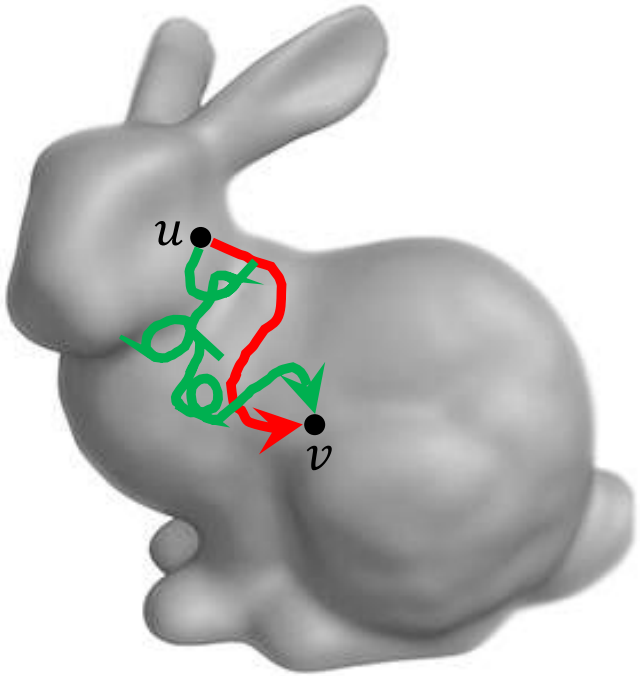
Why Manifolds?

In protein modeling, abstract out internal structure that is irrelevant for interactions + allow some conformation changes

Homogeneous Spaces

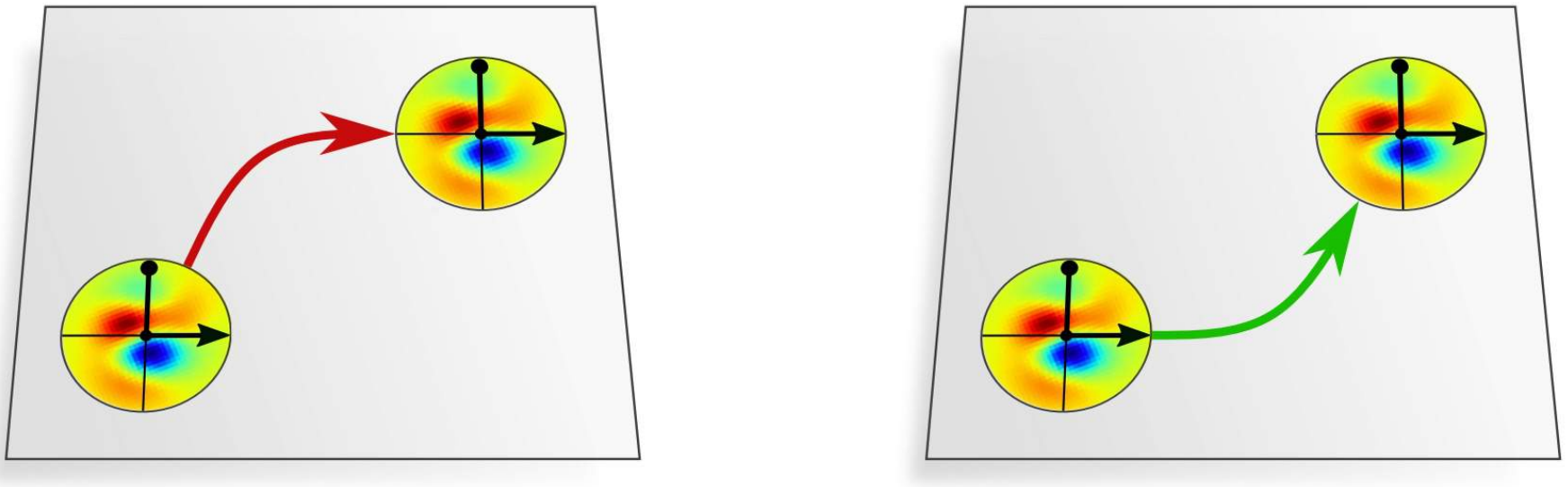


global symmetry group
$\exists g \in G$ s.t. $gu = v$



no global symmetry group

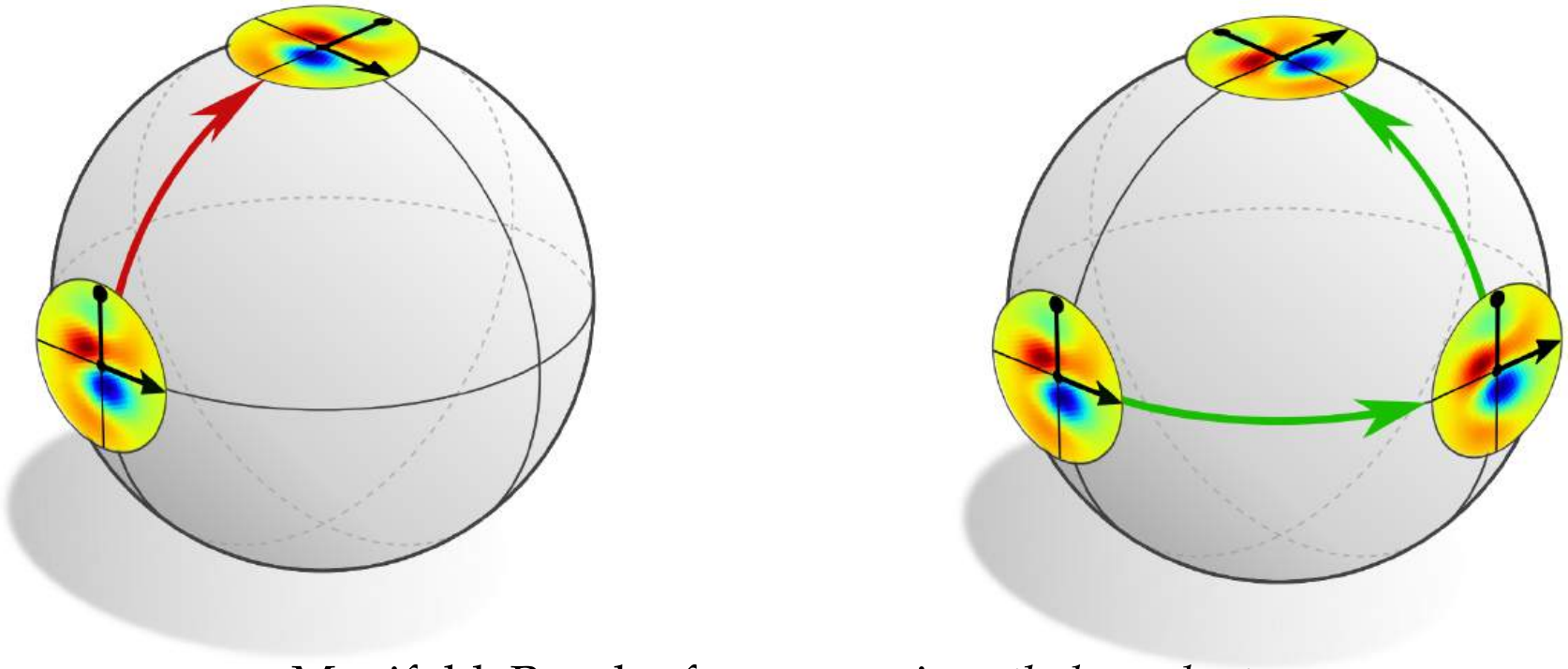
Euclidean Convolution



Euclidean space: Transport the filter around the domain

Figure: M. Weiler et al. 2021

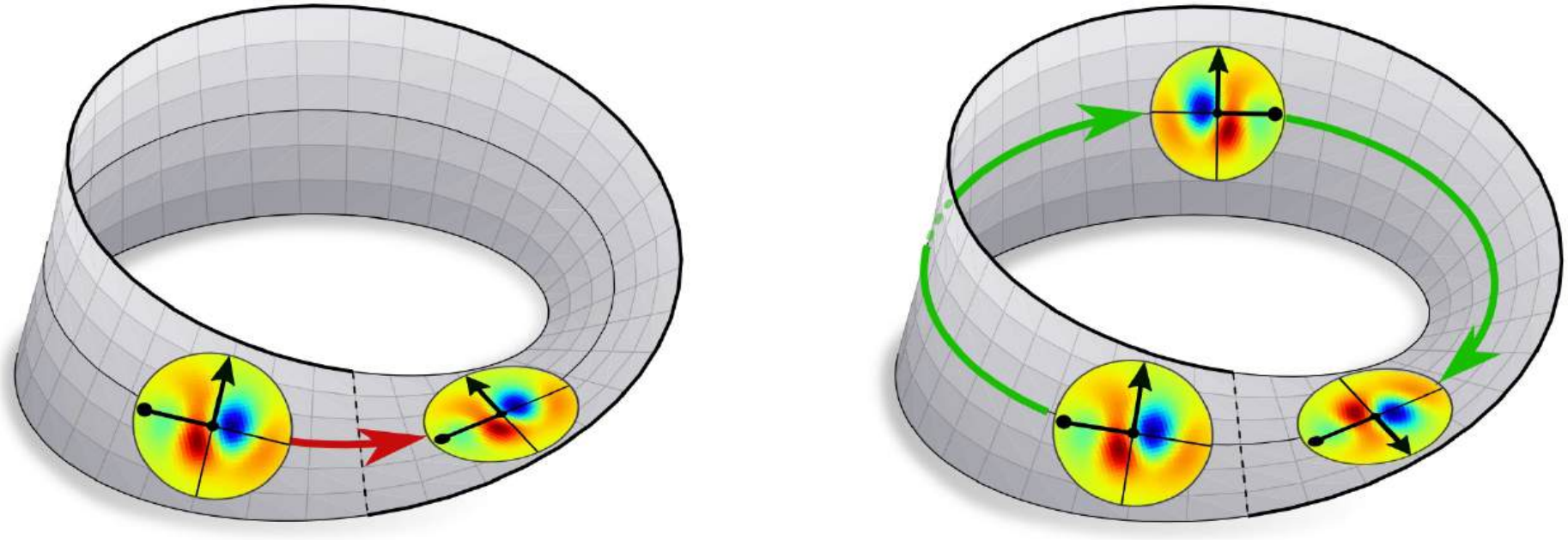
Non-Euclidean Convolution



Manifold: Result of transport is *path dependent*

Figure: M. Weiler et al. 2021

Non-Euclidean Convolution

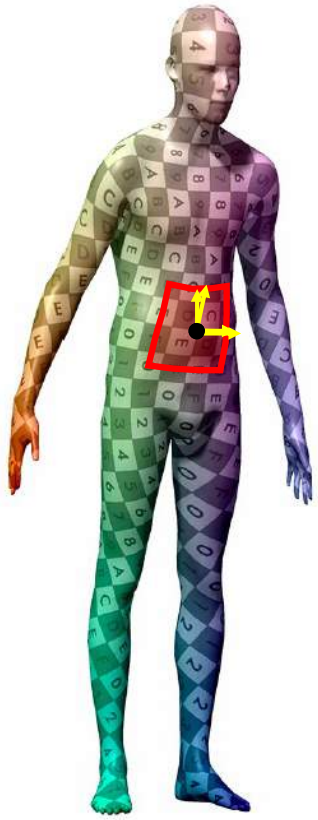


Manifold: Result of transport is *path dependent*

Figure: M. Weiler et al. 2021

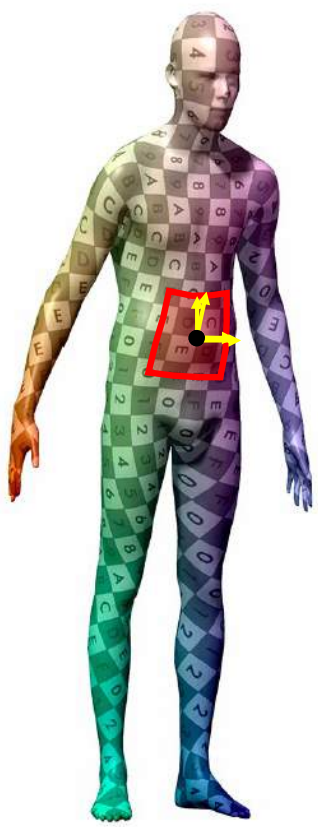
Two Types of Invariance

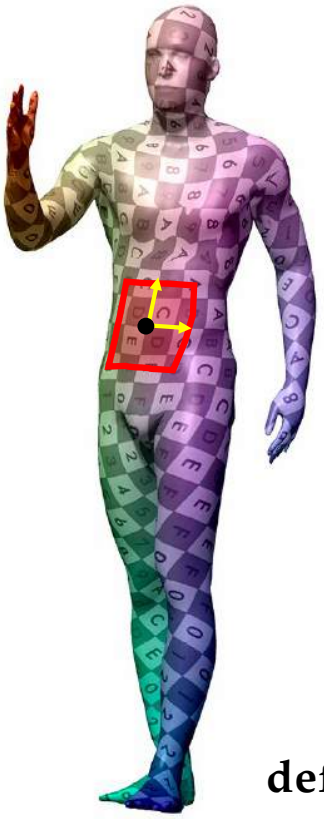
Local gauge transformation



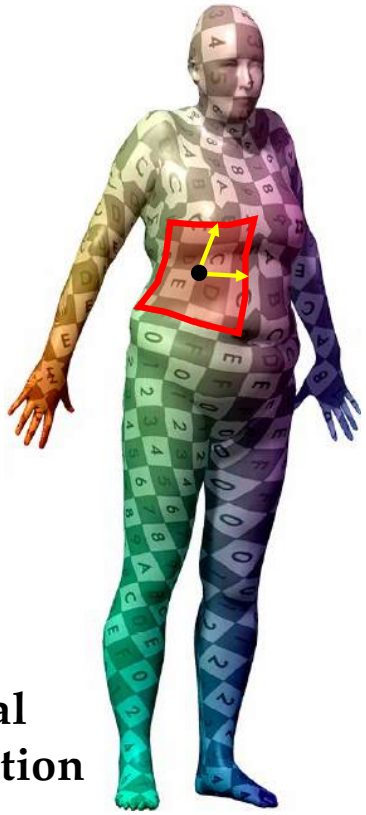
Two Types of Invariance

Local gauge transformation

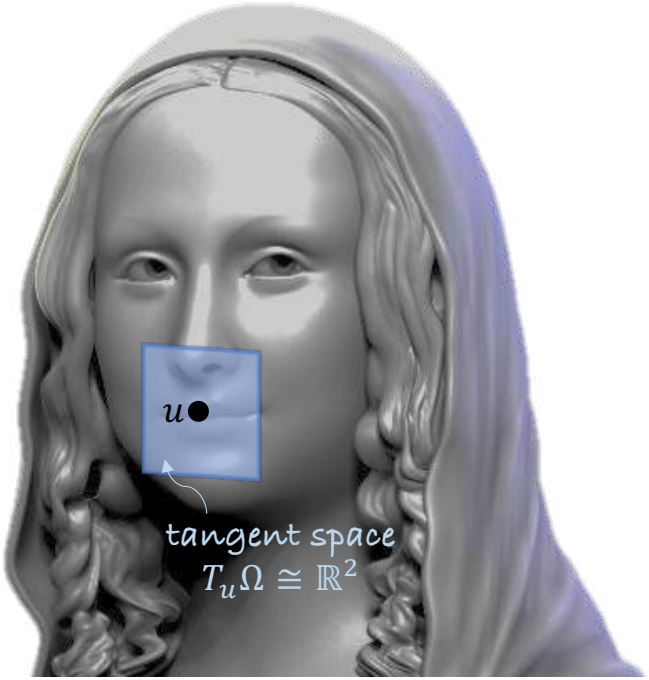




Global deformation



Manifolds



manifold Ω

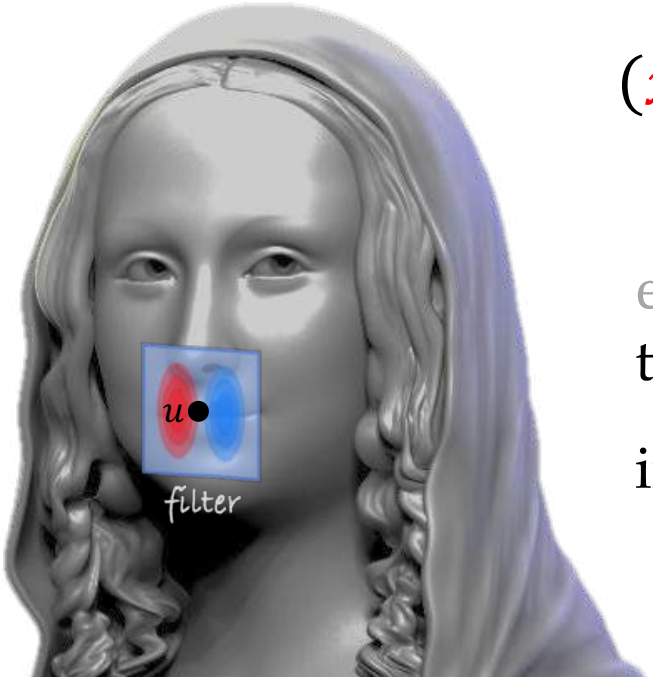
manifold = locally Euclidean space

Riemannian metric = local length / direction

Intrinsic quantity = expressed solely in terms of the Riemannian metric

Isometry = metric-preserving deformation

Geodesic CNNs



manifold Ω
isometry group $\mathrm{Iso}(\Omega)$

Masci et B 2015; Boscaini et B 2016; Monti et B 2017

Exponential map
$\exp_u : T_u\Omega \to \Omega$

$$(x \star \psi)(u) = \int_{T_u\Omega} \psi(v)\, x(\exp_u v)\, \mathrm{d}v$$

$\exp_u$ is an intrinsic map allowing to express the signal x locally in the tangent space $T_u\Omega$

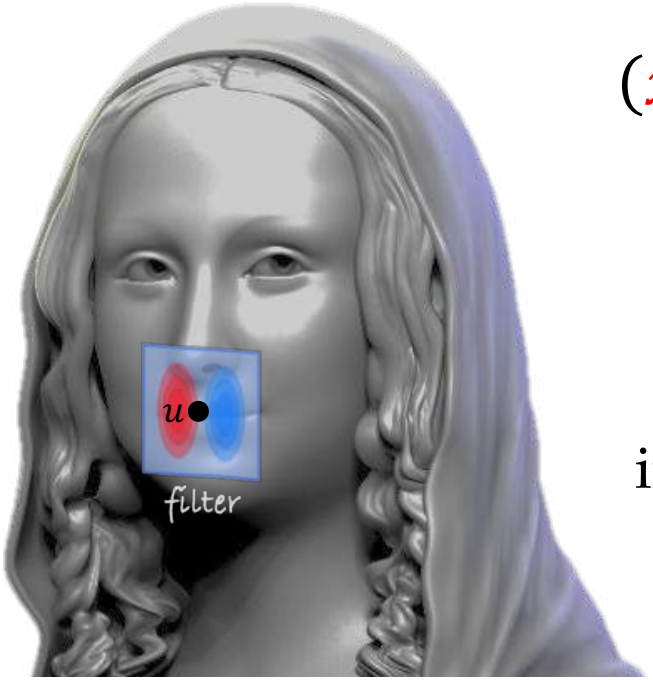
intrinsic filter = invariant to isometries

Geodesic CNNs

Anisotropic intrinsic filters on a manifold

Masci et B 2015; Boscaini et B 2016; Monti et B 2017

Geodesic CNNs



manifold Ω
isometry group $\mathrm{Iso}(\Omega)$

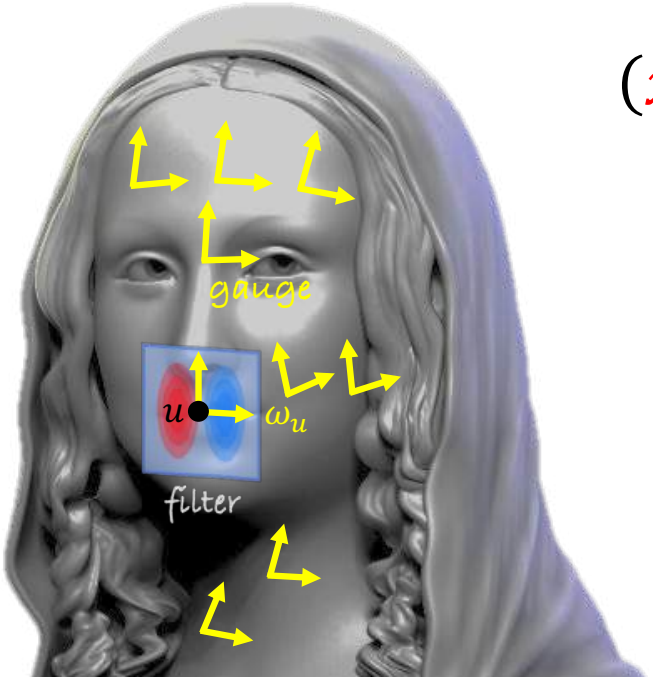
$$(x \star \psi)(u) = \int_{T_u\Omega} \psi(v)\, x(\exp_u v)\, \mathrm{d}v$$

Problem: these are abstract vectors!

intrinsic filter = invariant to isometries

Masci et B 2015; Boscaini et B 2016; Monti et B 2017

Geodesic CNNs

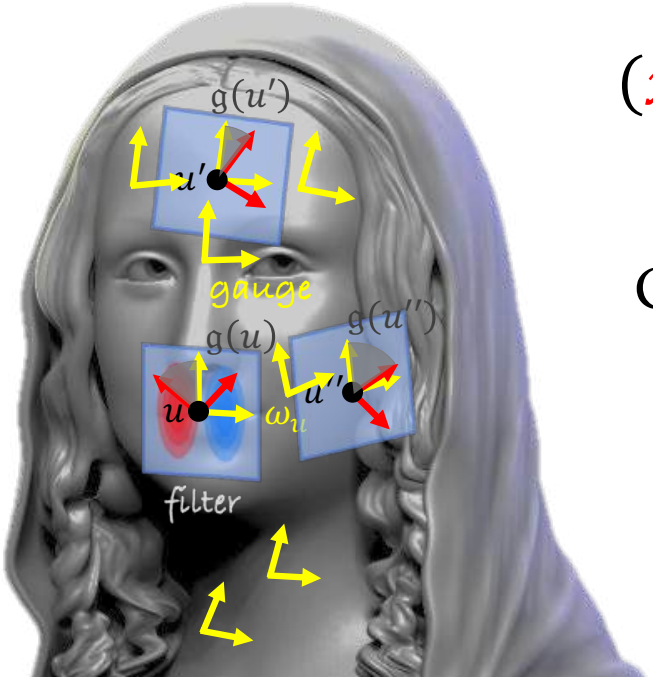


manifold Ω
isometry group $\mathrm{Iso}(\Omega)$

local reference frame
$\omega_u : \mathbb{R}^2 \to T_u\Omega$

$$(x \star \psi)(u) = \int_{\mathbb{R}^2} \psi(\mathbf{v}) x(\exp_u \omega_u \mathbf{v}) \mathrm{d}\mathbf{v}$$

Gauge Transformations



manifold Ω
structure group G

Cohen et al. 2019; Weiler et al. 2021

$$(x \star \psi)(u) = \int_{\mathbb{R}^2} \psi(\mathbf{v}) x(\exp_u \omega_u \mathbf{v}) \mathrm{d}\mathbf{v}$$

Gauge defined up to gauge transformation

$$g : \Omega \to G$$

Structure Group

A gauge is defined up to a *gauge transformation* $g: \Omega \to G$

• "Naked" manifold	$\mathrm{GL}(s)$	invertible matrices
• Manifold+orientation	$\mathrm{GL}^+(s)$	invertible matrices with det>0
• Manifold+volume	$\mathrm{SL}(s)$	matrices with det=1
• Manifold+metric	$\mathrm{O}(s)$	orthogonal matrices
• Manifold+metric+orientation	$\mathrm{SO}(s)$	orthogonal matrices with det=1
• Manifold+frame field	{id}	identity (no ambiguity)

Weiler et al. 2021

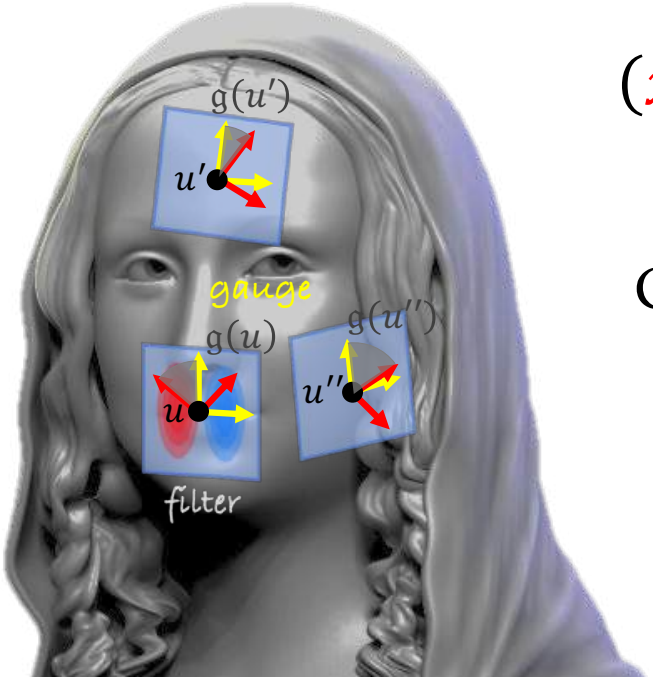
Structure Group

A gauge is defined up to a *gauge transformation* $g: \Omega \to G$

• "Naked" manifold	$\mathrm{GL}(s)$	invertible matrices
• Manifold+orientation	$\mathrm{GL}^+(s)$	invertible matrices with det>0
• Manifold+volume	$\mathrm{SL}(s)$	matrices with det=1
• Manifold+metric	$\mathrm{O}(s)$	orthogonal matrices
• Manifold+metric+orientation	$\mathrm{SO}(s)$	orthogonal matrices with det=1
• Manifold+frame field	{id}	identity (no ambiguity)

Weiler et al. 2021

Gauge-equivariant CNNs



manifold Ω
structure group $G = \mathrm{SO}(2)$

$$(x \star \psi)(u) = \int_{\mathbb{R}^2} \psi(\mathbf{v}) x(\exp_u \omega_u \mathbf{v}) \mathrm{d}\mathbf{v}$$

Gauge defined up to gauge transformation

$$g : \Omega \to \mathrm{SO}(2)$$

gauge-equivariant filter

$$\psi(g^{-1}\mathbf{v}) = \rho(g^{-1})\psi(\mathbf{v})\rho(g)$$

Cohen et al. 2019

“Hairy Ball” (a.k.a. Poincaré-Hopf) Theorem

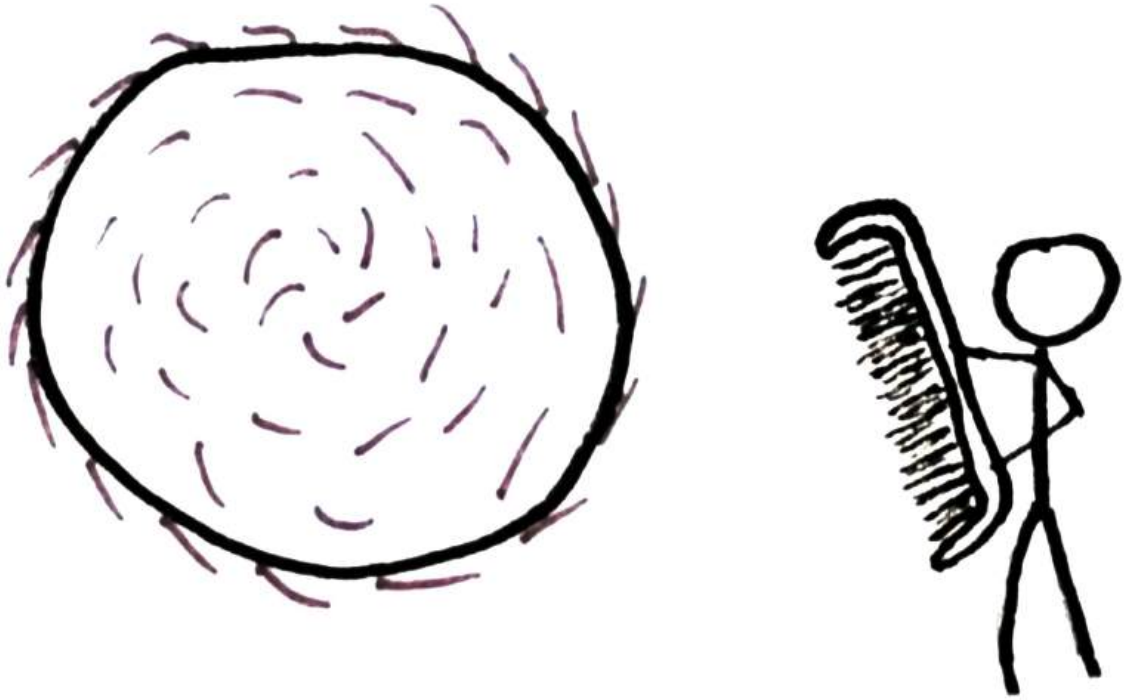
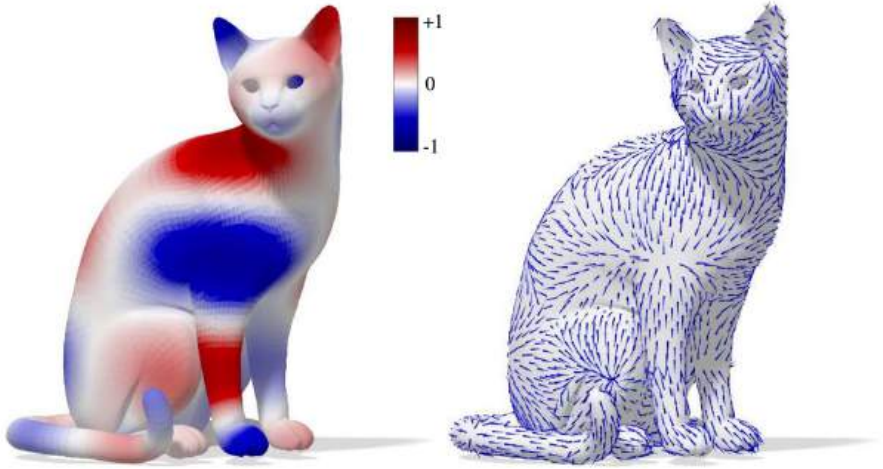
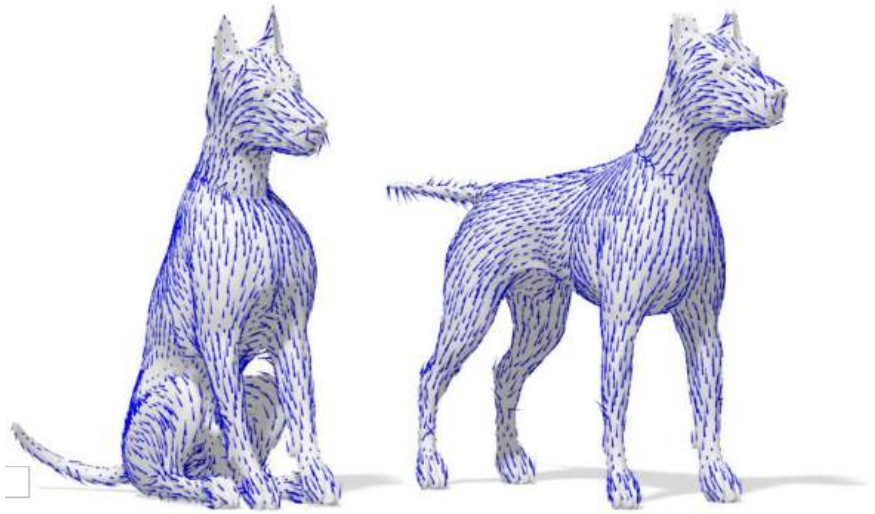


Image: Minutephysics

Theory vs Practice: Stable Gauges



Gradient of intrinsic function

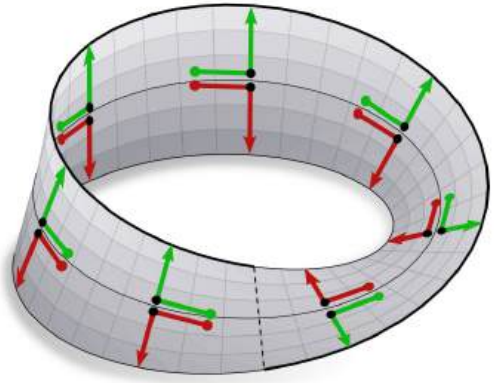


Deformation-invariant
stable gauge

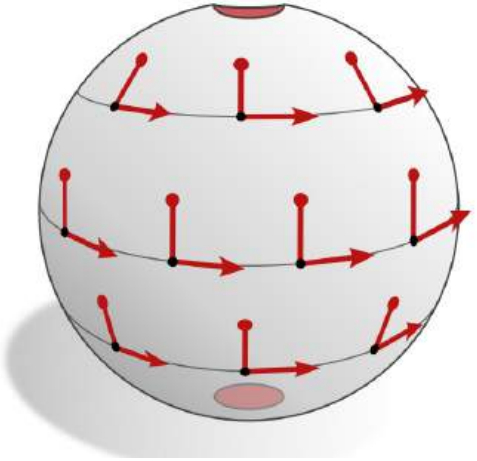
Melzi et B 2019; Monti et B 2017

Structure Group

rotation
SO(2)

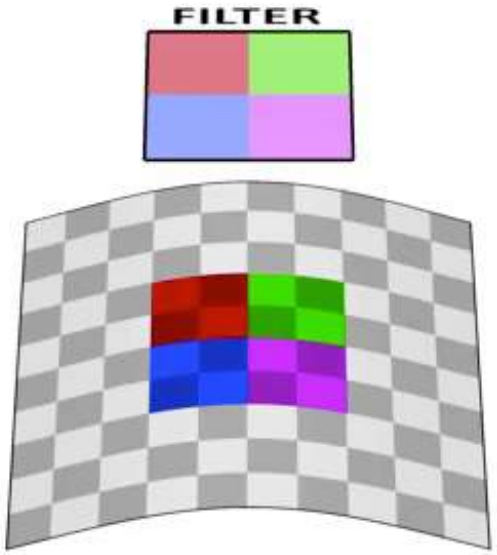


reflection
R

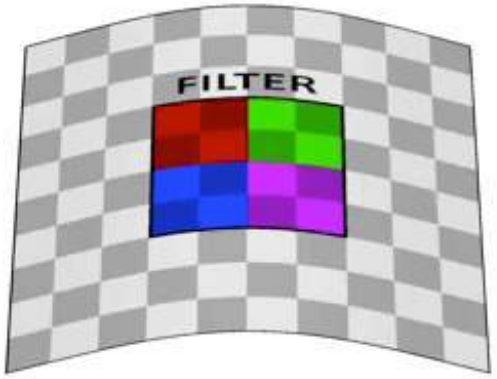


fixed gauge
{id}

Weiler et al. 2021



Euclidean (extrinsic) convolution

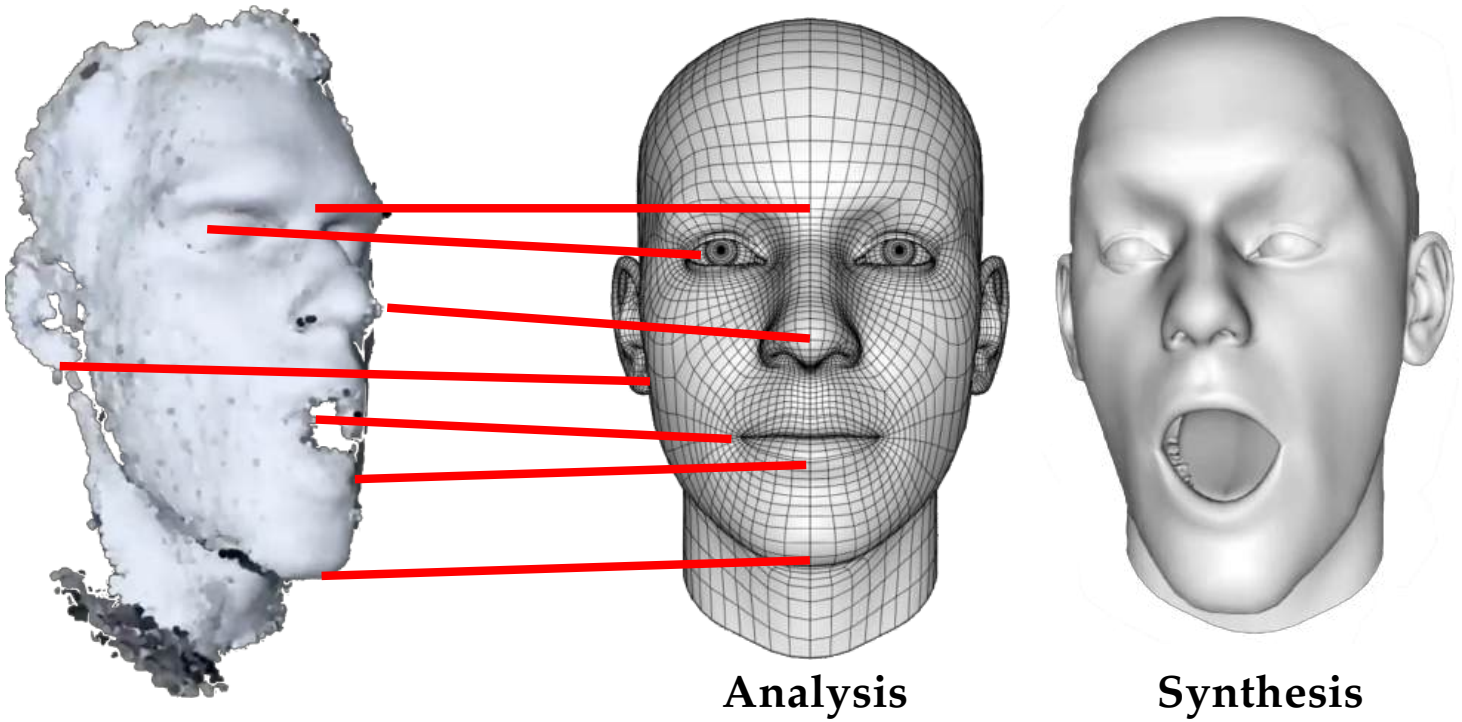


Geometric (intrinsic) convolution


Applications

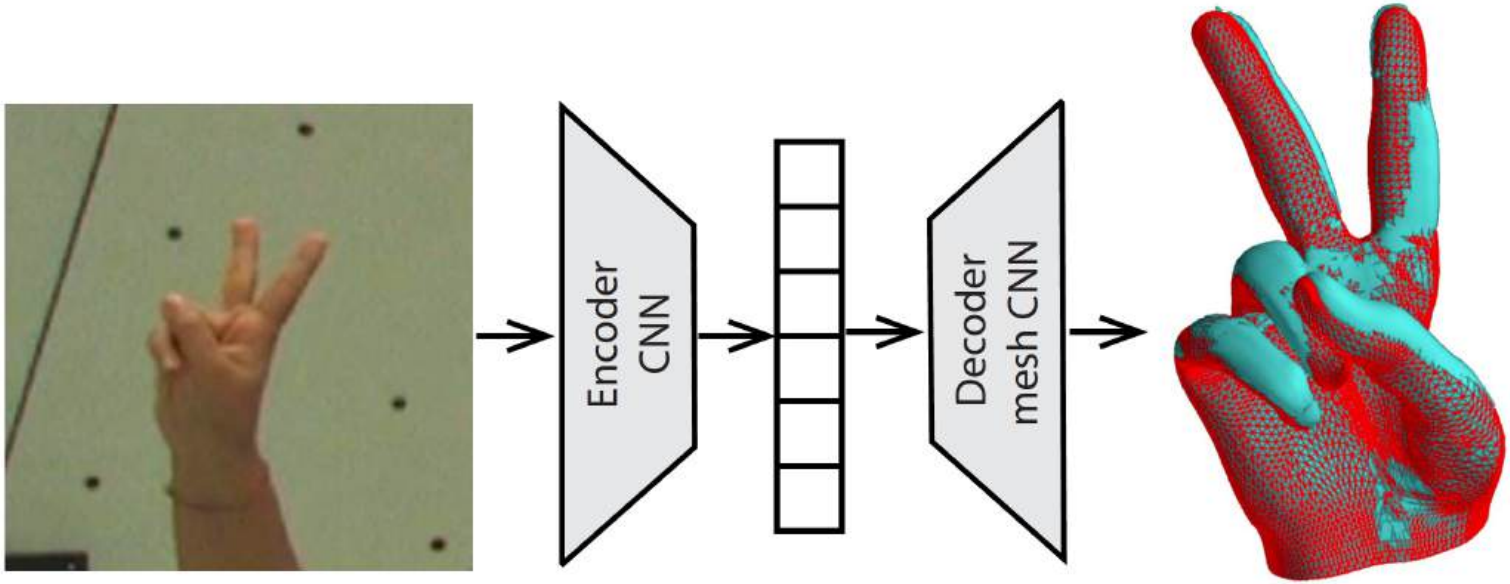

FaceShift 2015
GDC

Shape Analysis & Synthesis

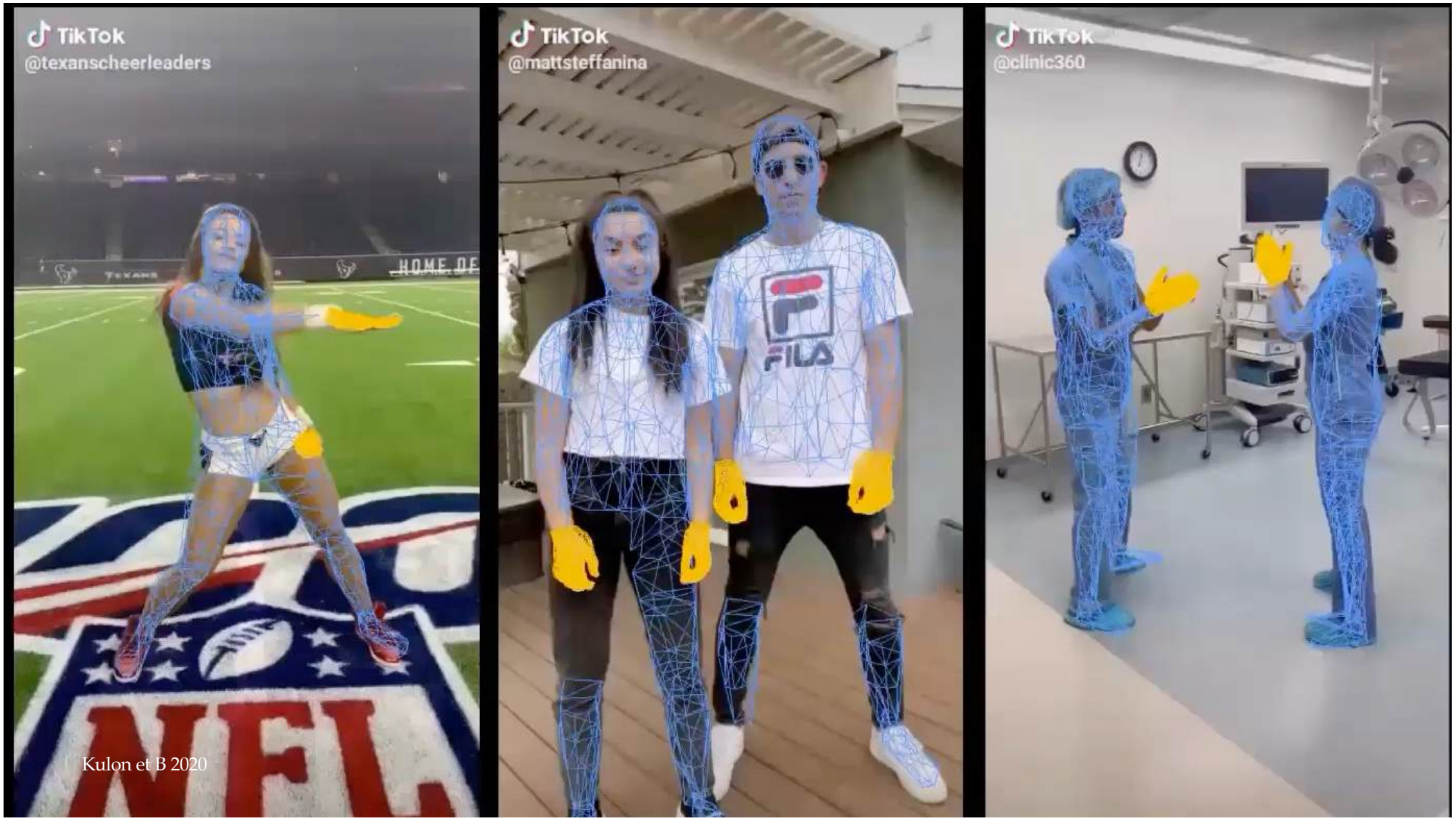


Analysis **Synthesis**

3D Hand Reconstruction



Kulon et B 2020

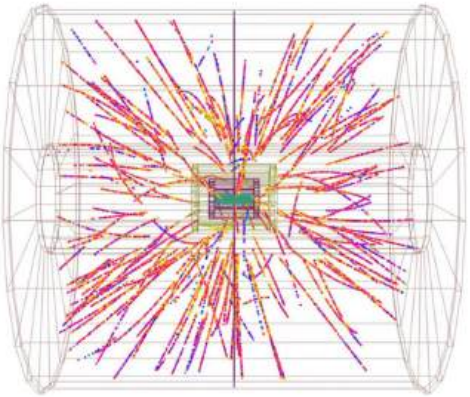


Kulon et B 2020

Snap Acquires Ariel AI To Enhance AR Features

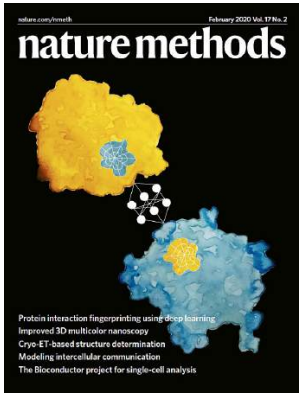
Fake news detection

Navigation



Particle physics

Pure math



Structural biology

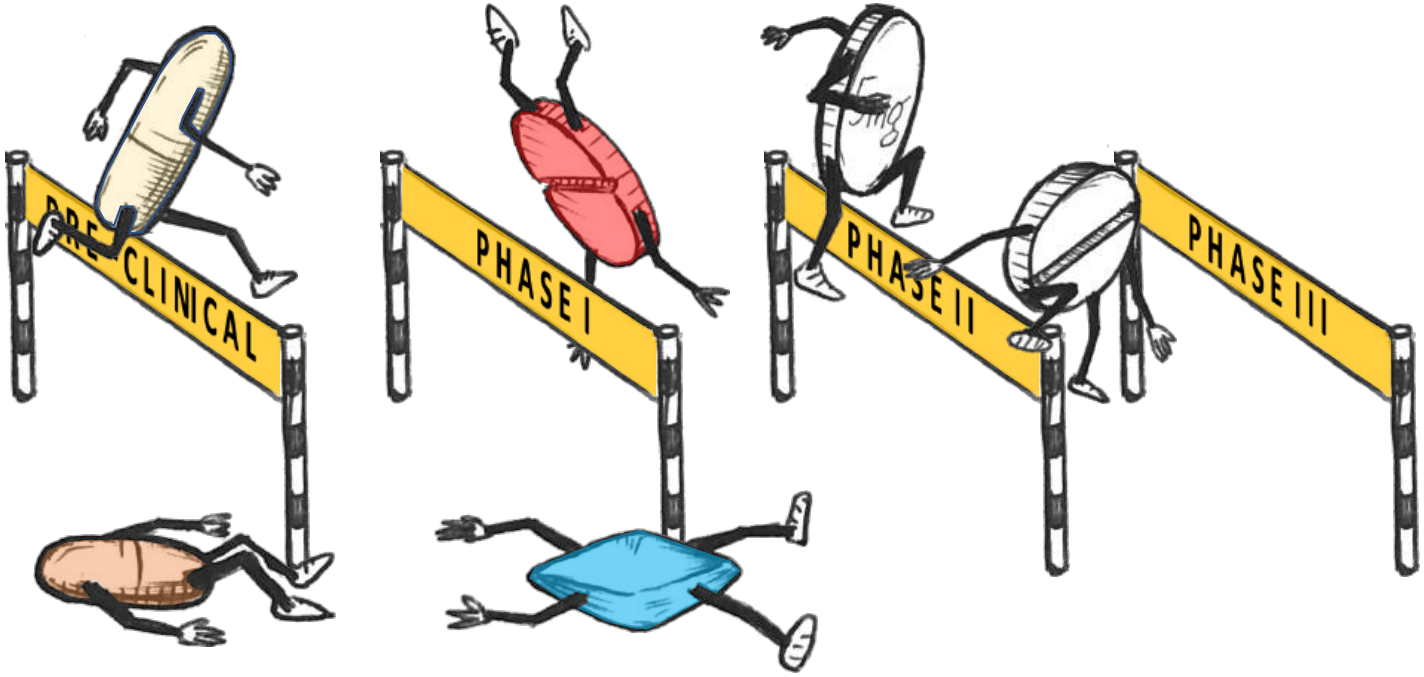
Drug discovery

Vosoghi et al. 2018

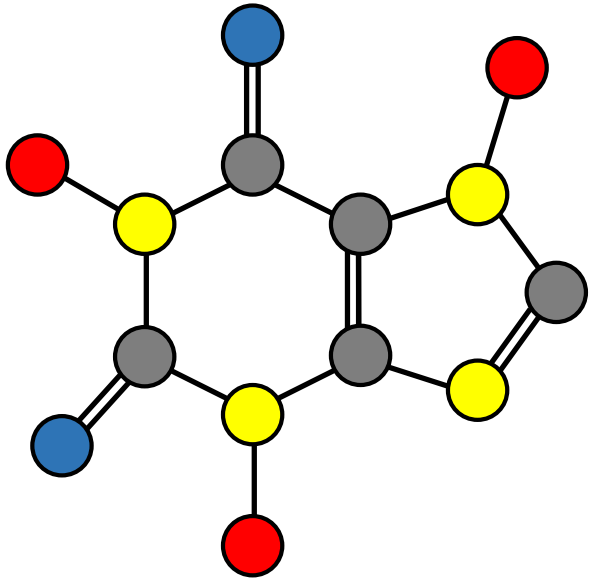
Monti et B 2019

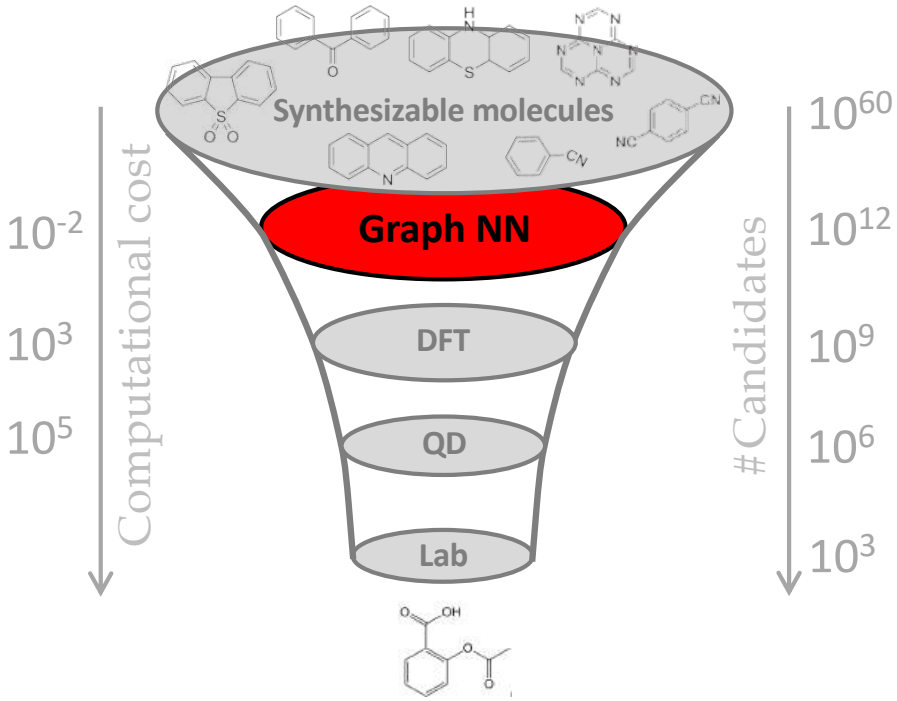
Acquired by Twitter in 2019

Drug Discovery & Design



Virtual Drug Screening

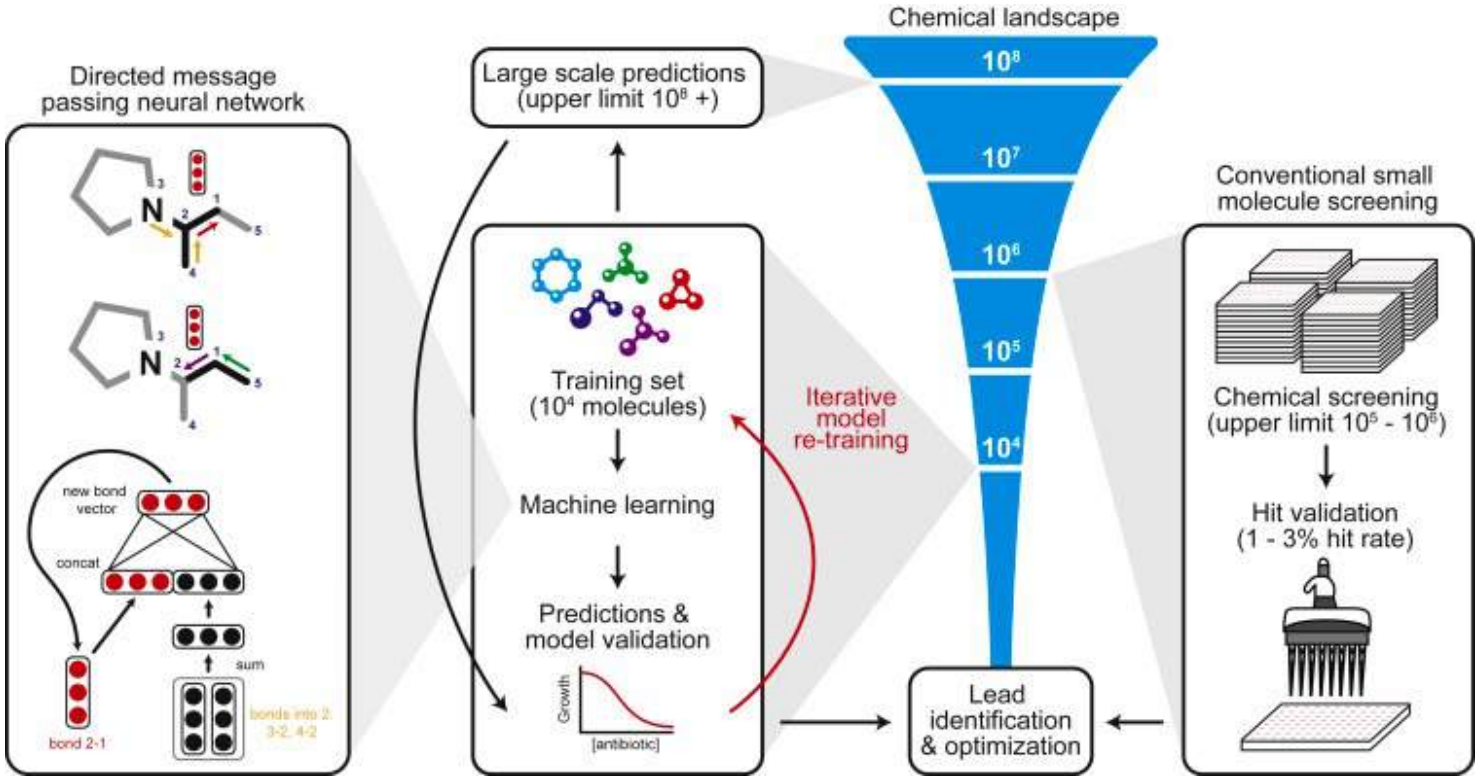




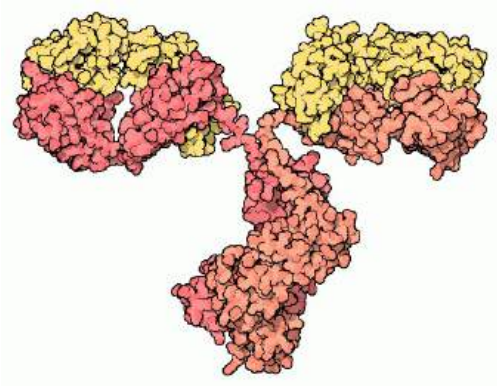
"Computational funnel"

Duvenaud et al. 2015; Gilmer et al. 2017; Jin et al. 2020

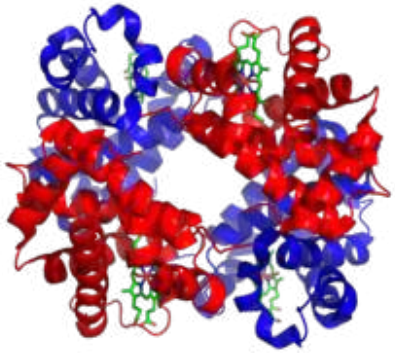
New Antibiotic Discovery



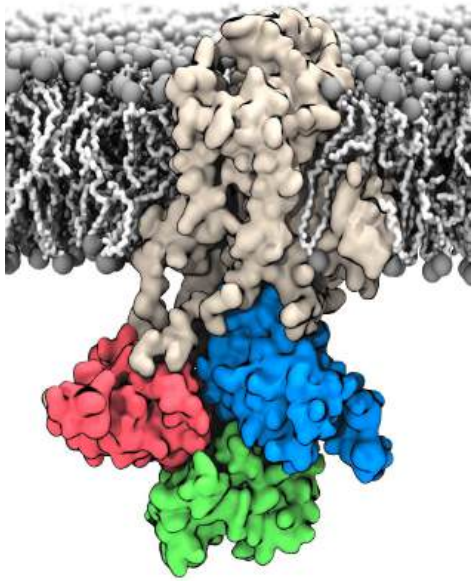
Stokes et al. 2020



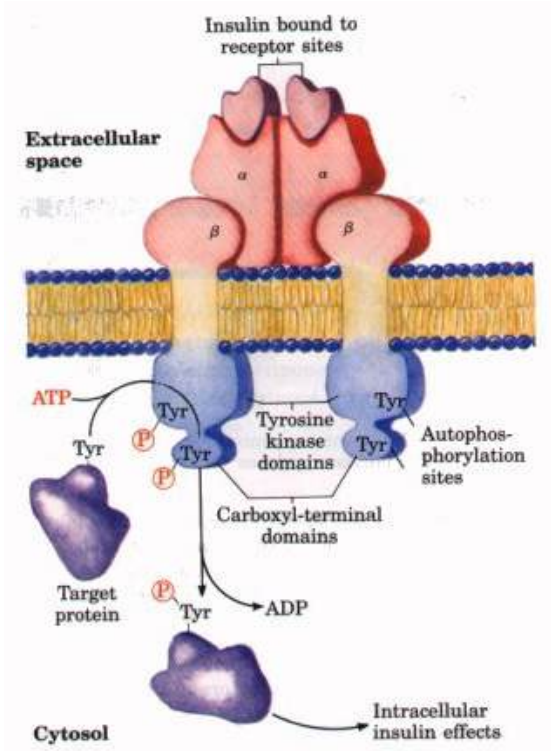
Defense (antibody)



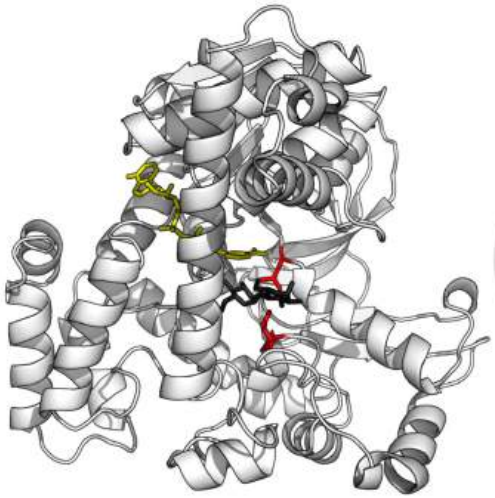
Storage (haemoglobin)

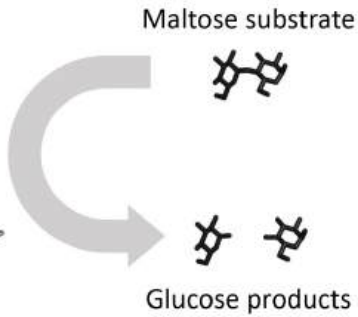


Transport (calcium pump)



Communication (insulin)



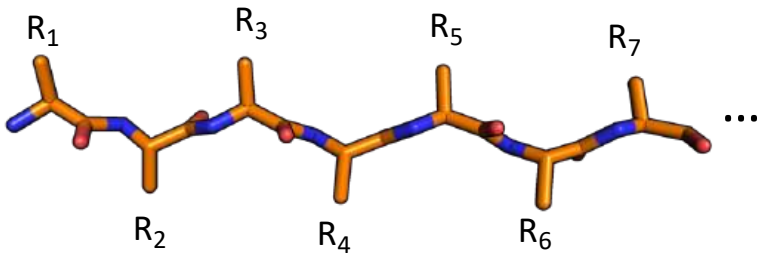


Catalysis (enzyme)

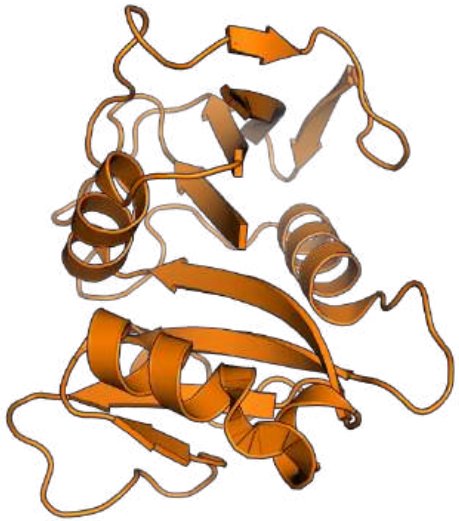
Structure (collagen)

Protein Folding

Protein folding



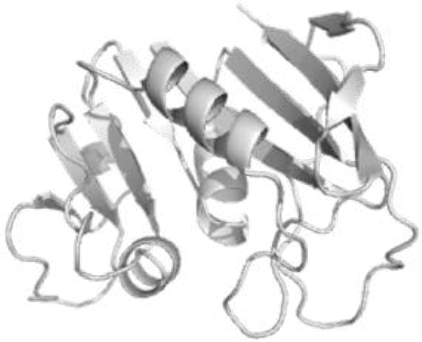
Primary protein structure

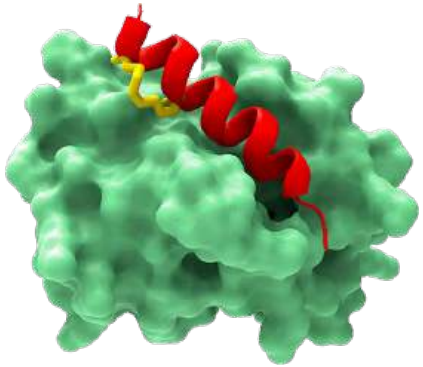


Tertiary protein structure

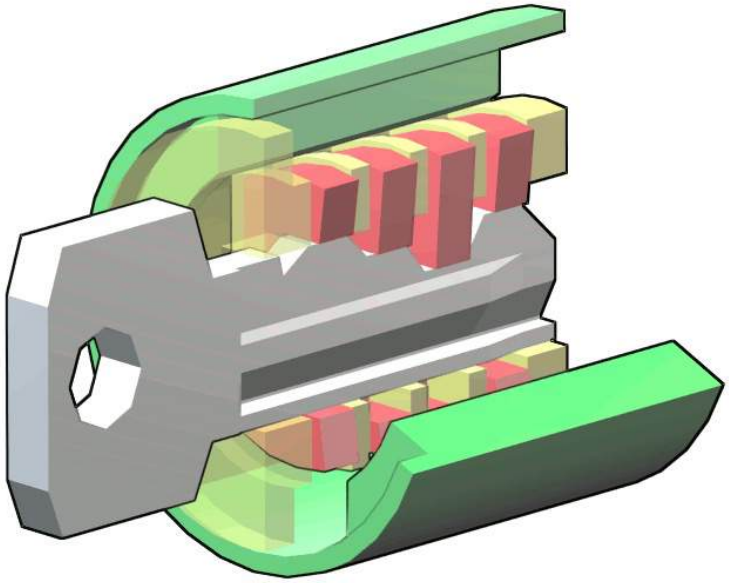
Sequence → Structure → Function

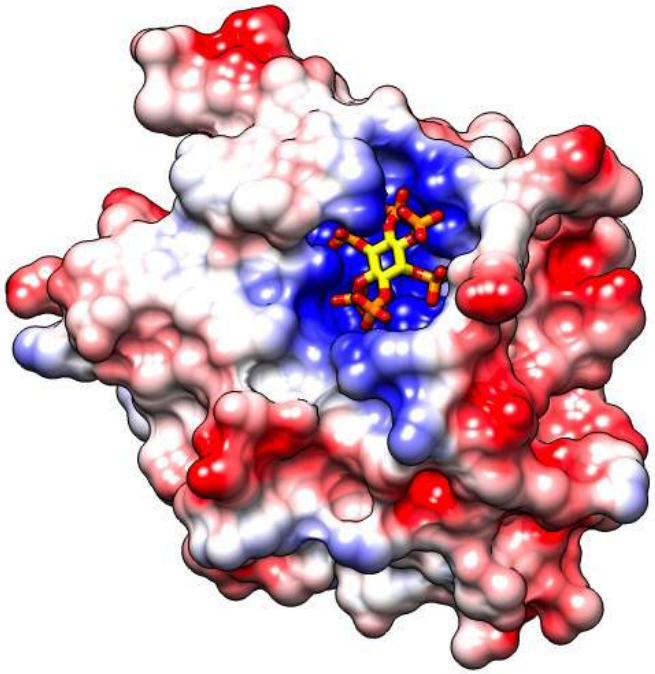
LGCVLAWNTNSK





Lock-Key Metaphor

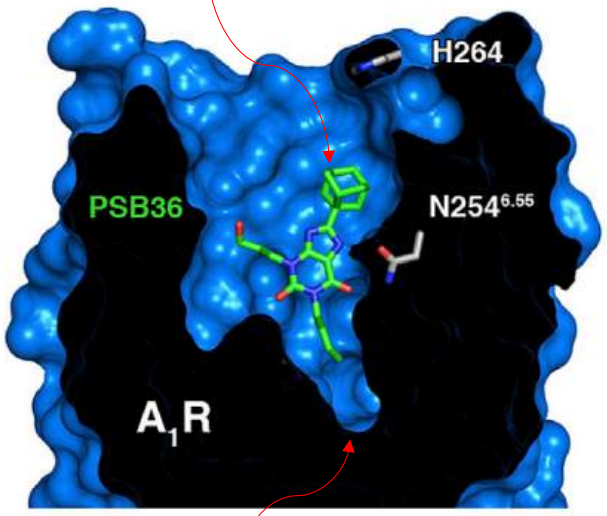




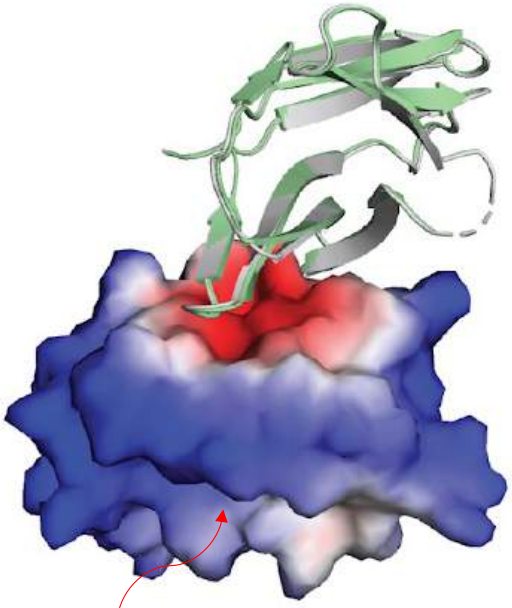
Emil Fischer “Schlüssel-Schloss-Prinzip” 1894

Biologic Drug Design

drug molecule



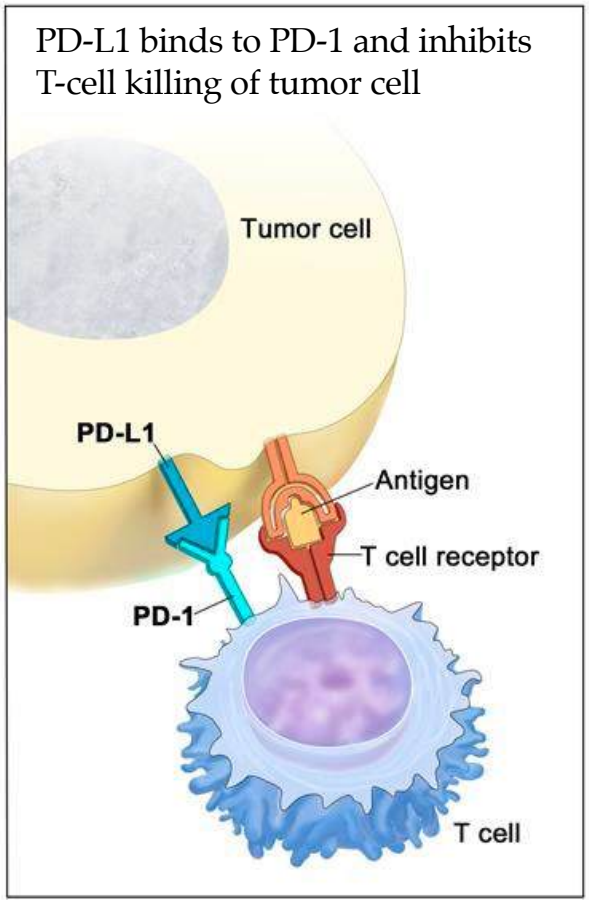
pocket in
protein surface

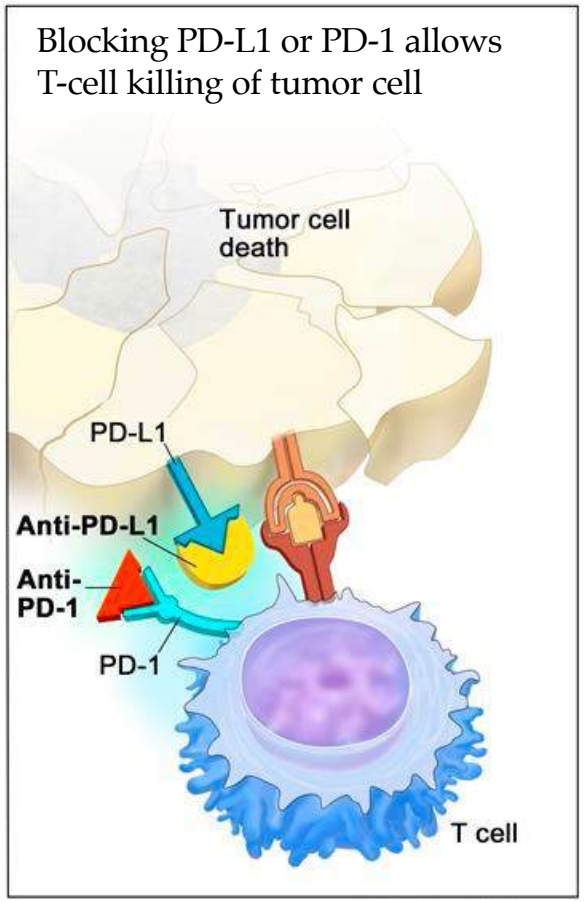


PD-L1 protein

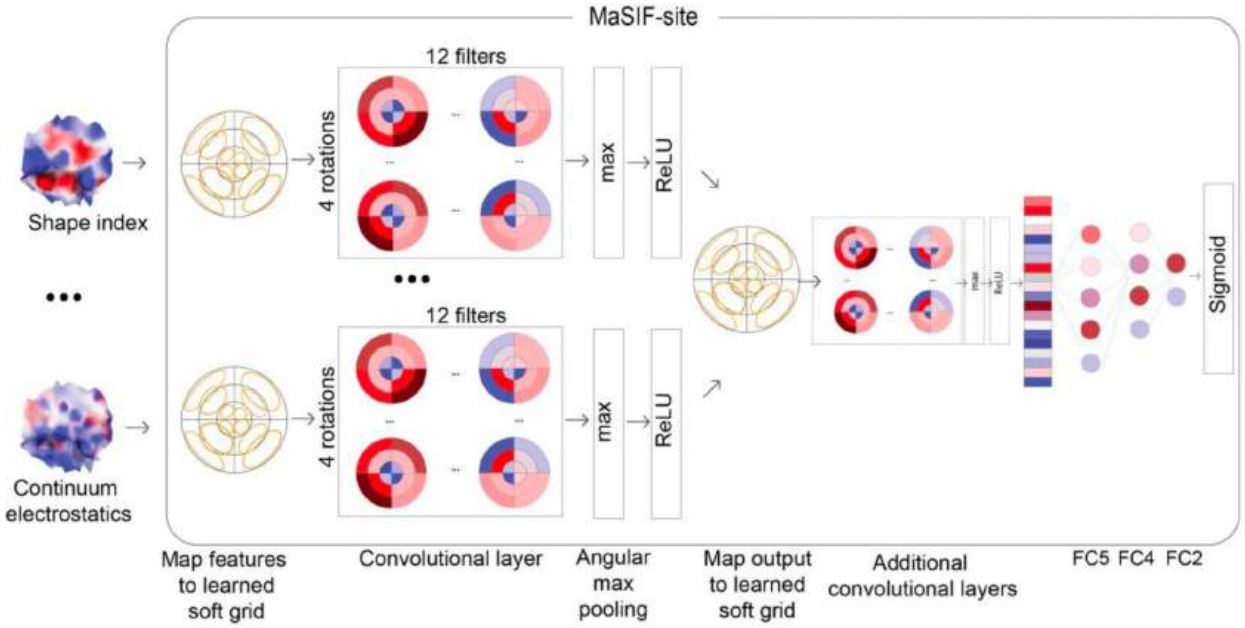
Immunotherapy

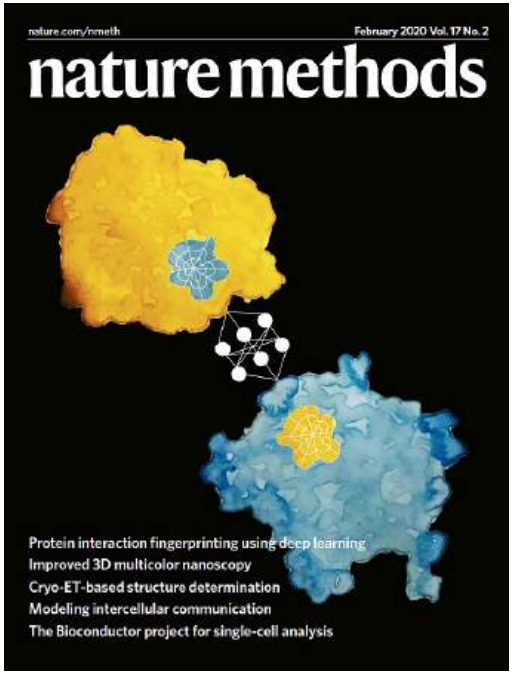
2018 Nobel Prize
PD-proteins role in
immunotherapy





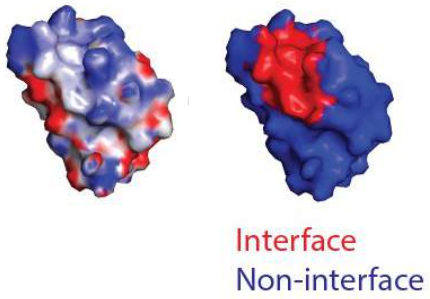
MaSIF: Geometric ML for Protein Function Prediction



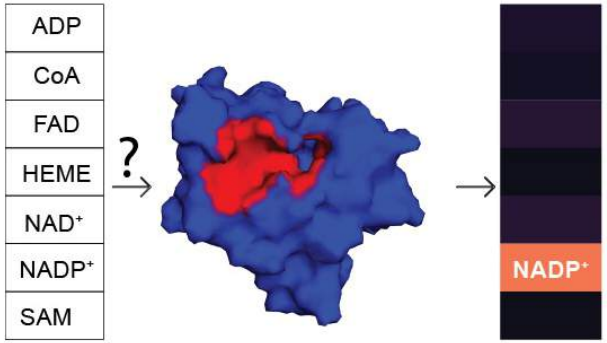


Gainza, Sverrisson et B, Correia 2020

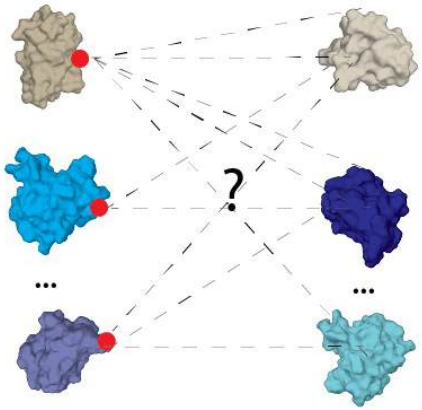
MaSIF Applications



Interface site prediction
MaSIF-site

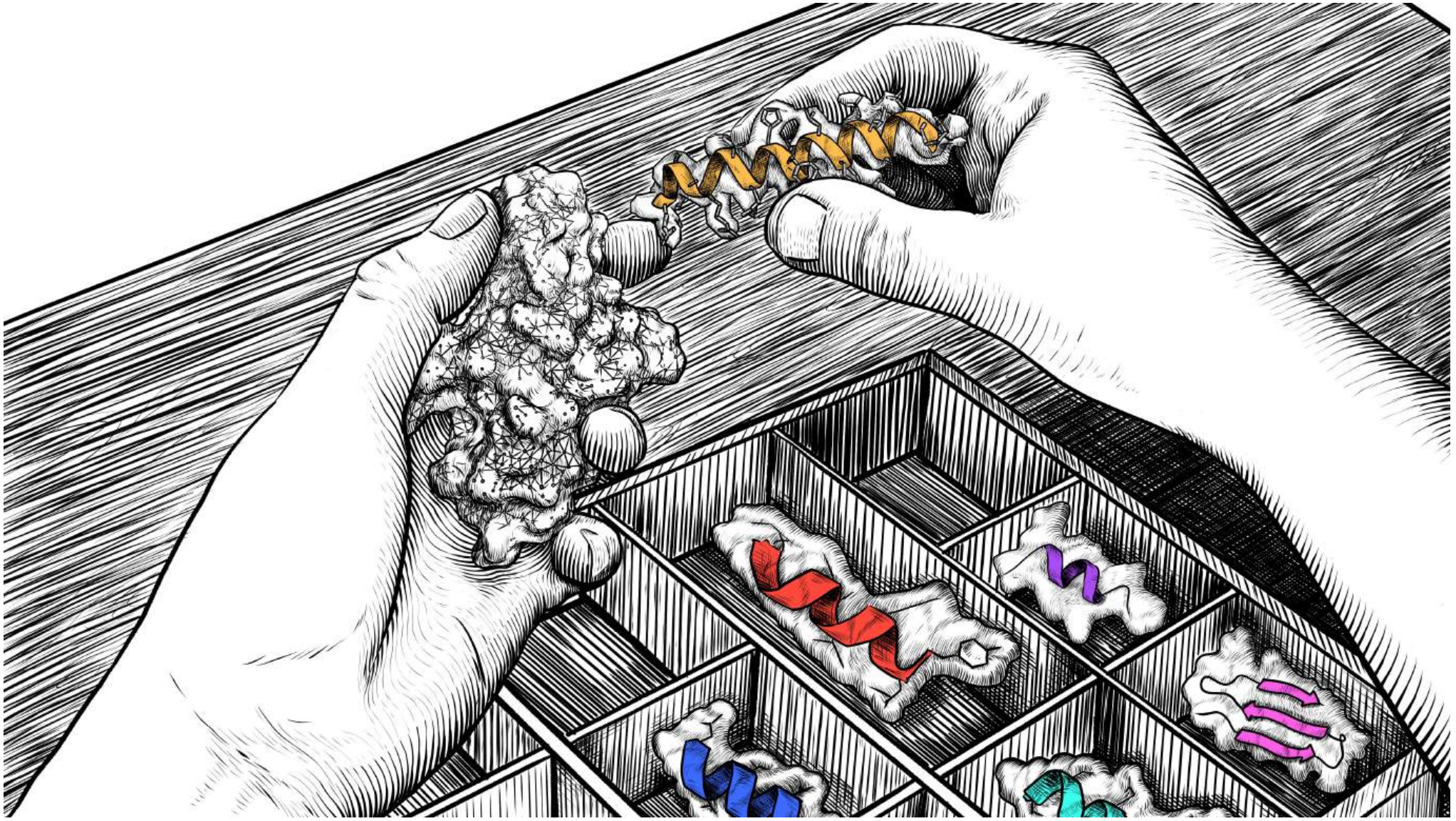


Pocket classification
MaSIF-ligand

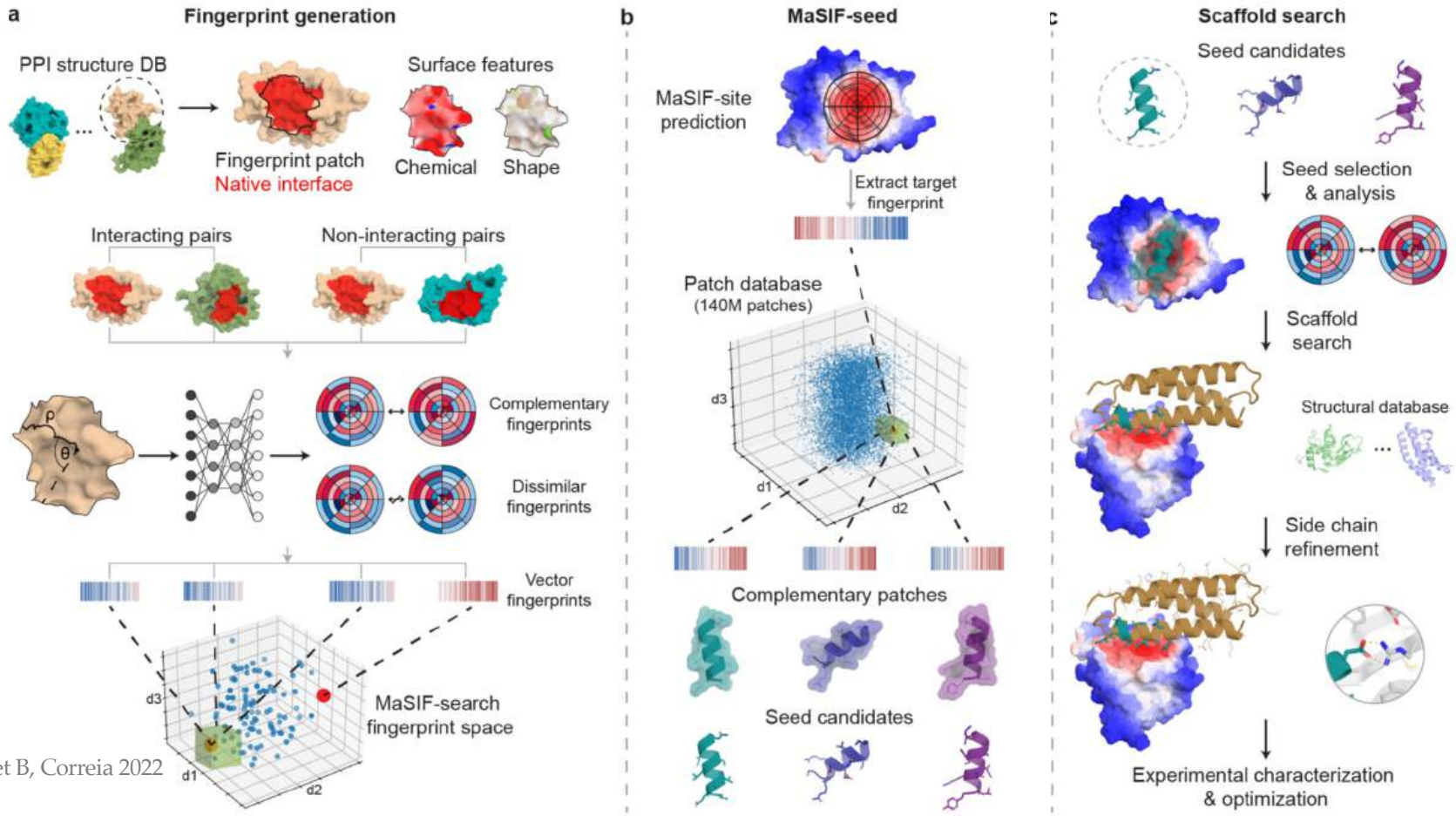


Fast PPI search
MaSIF-search

Gainza, Sverrisson et B, Correia 2020

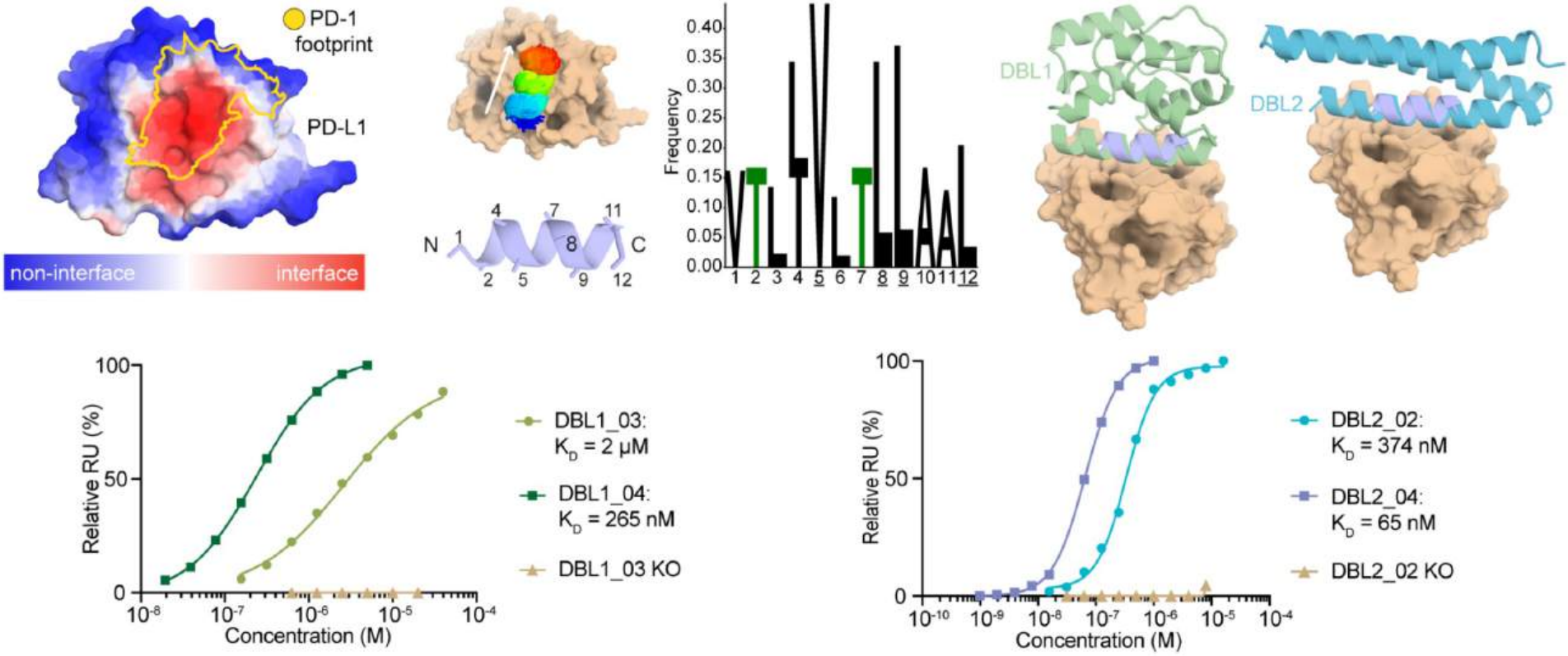


De novo protein design with MaSIF



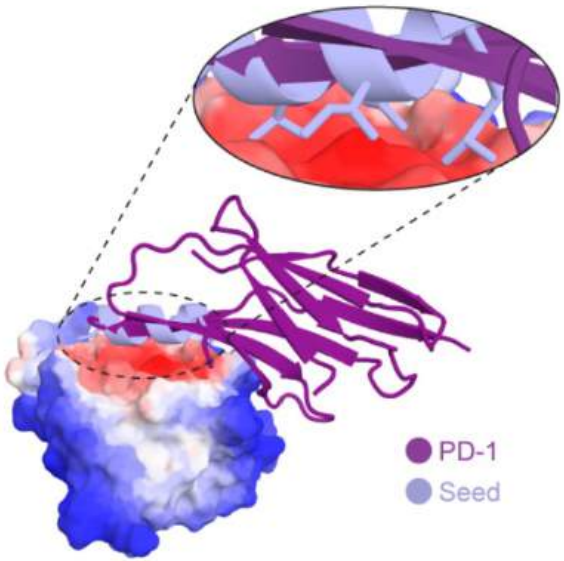
Gainza et B, Correia 2022

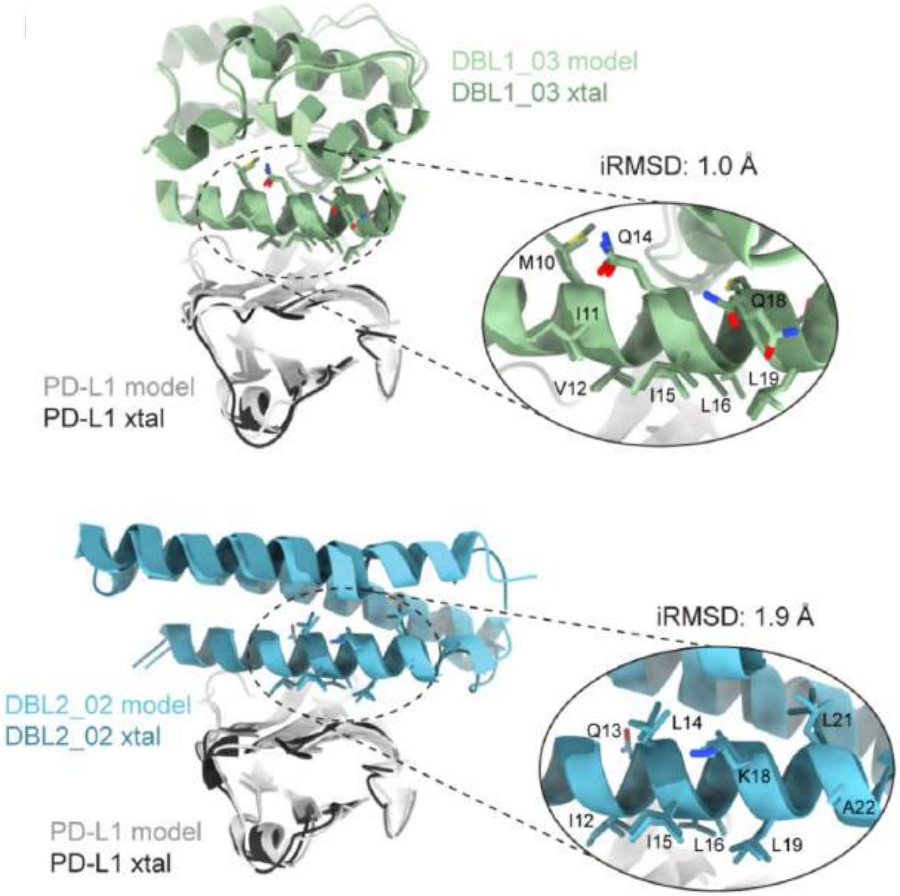
PD-L1 binder design



Gainza et B, Correia 2022

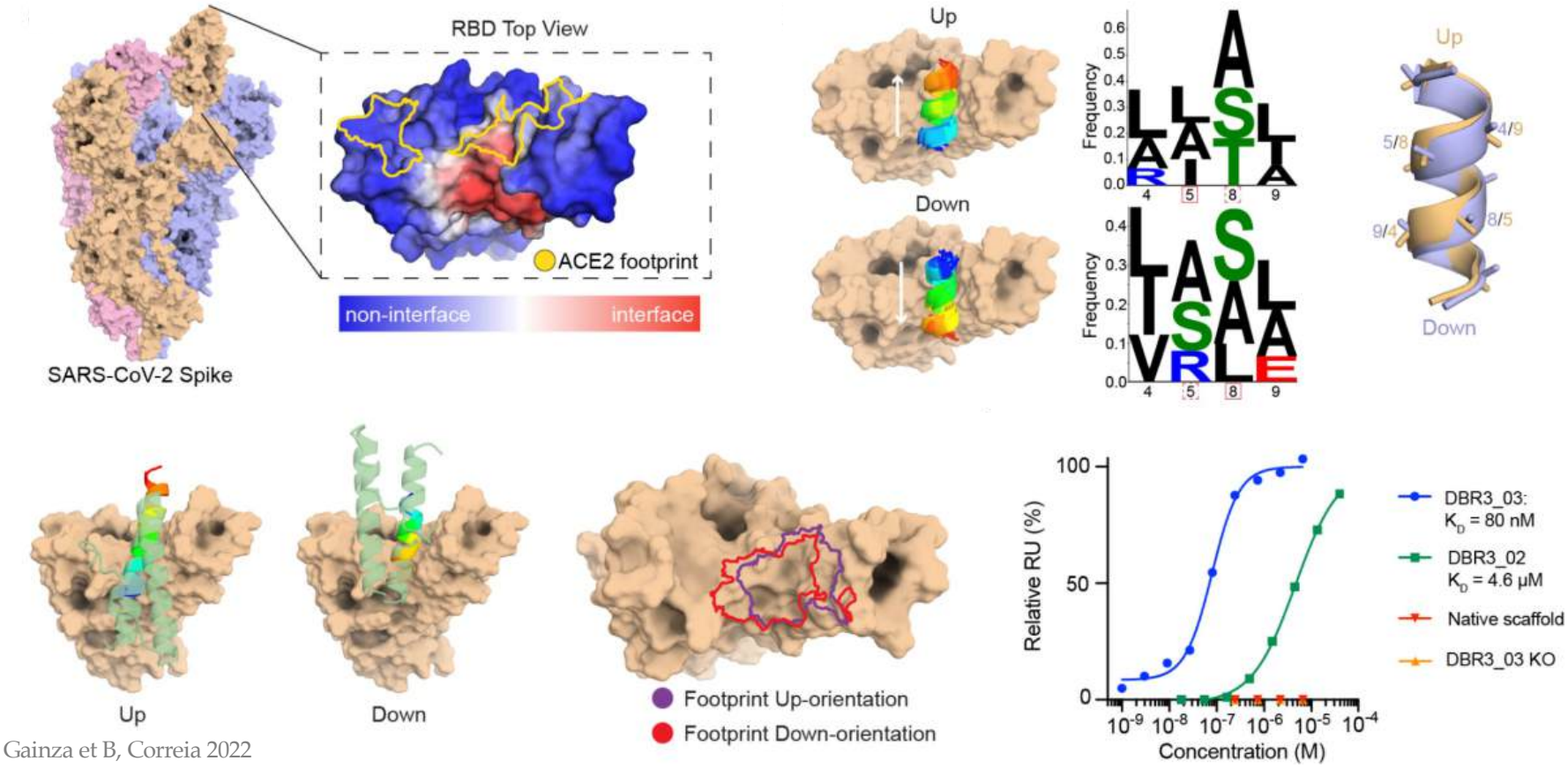
PD-L1 binder structure





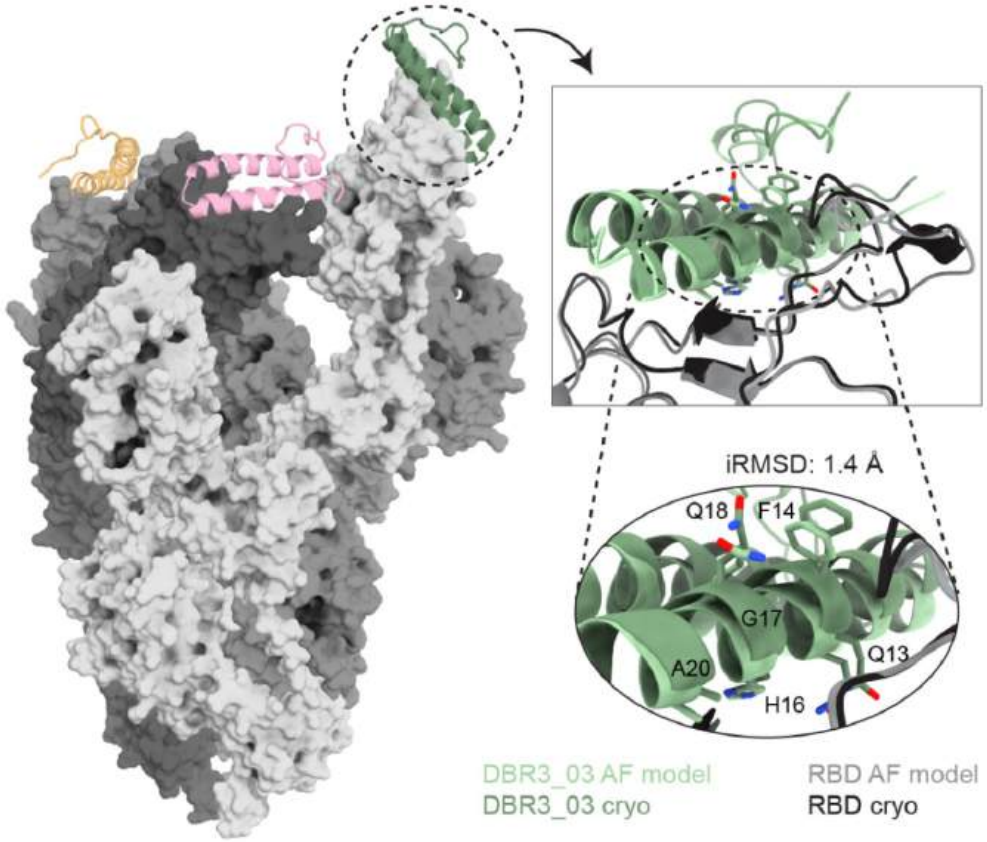
Gainza et B, Correia 2022

SARS-CoV-2 spike binder design



Gainza et B, Correia 2022

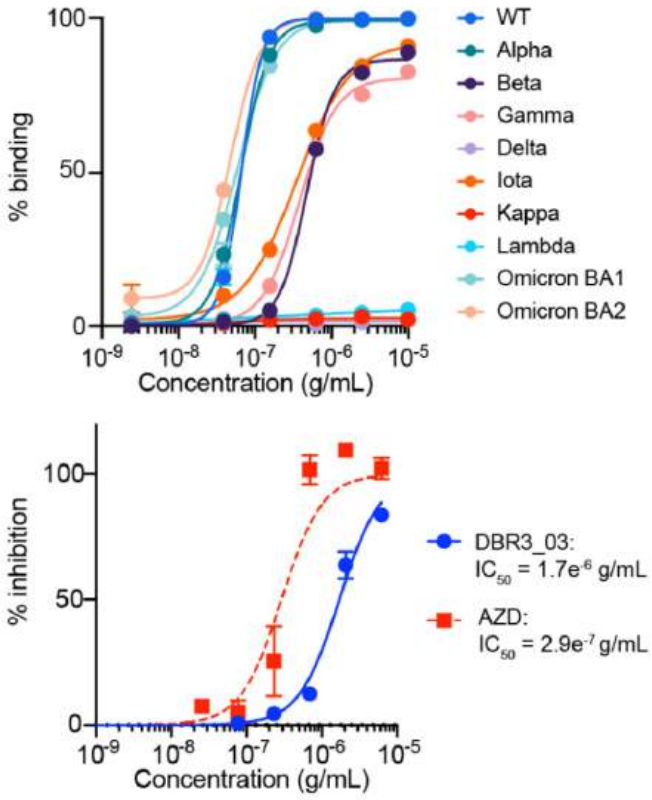
SARS-CoV-2 spike binder function



Gainza et B, Correia 2022

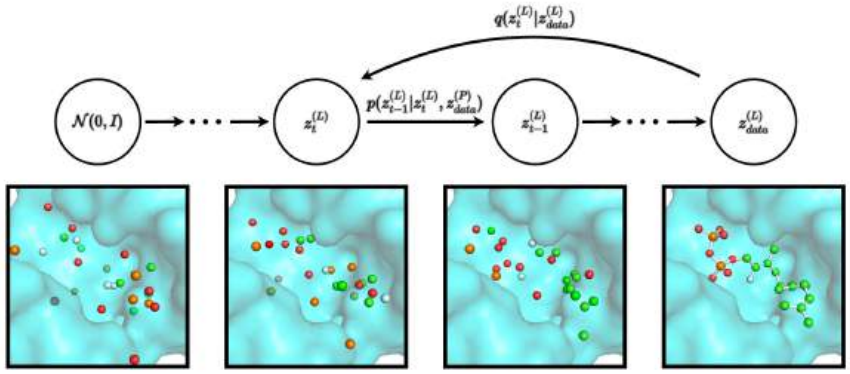
Structure

Binding spike protein in multiple virus variants



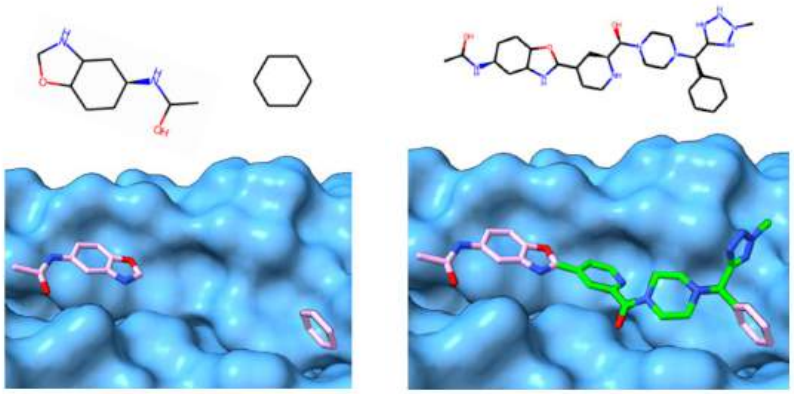
Pseudovirus neutralisation

Structure-Based Drug Design



Equivariant diffusion model conditioned on target pocket 3D structure

Schneuing et B, Welling, Correia 2022

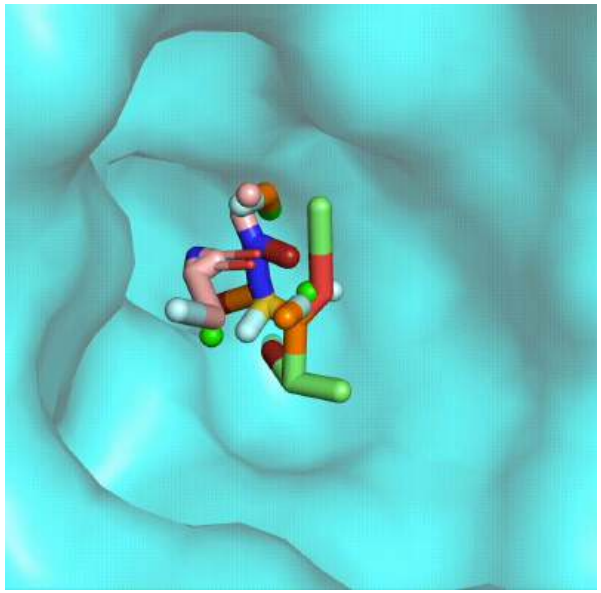


DiffLinker: Impainting of molecular fragments conditioned on target pocket

Igashov et B, Welling, Correia 2022

“Painting of an astronaut riding a dog on the Moon”

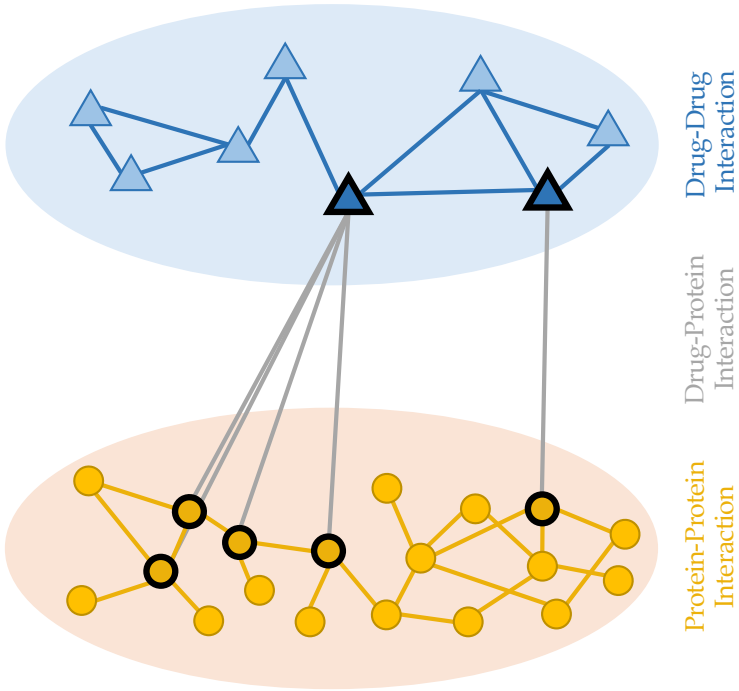
DALL-E 2023 (prompt by B)



“Drug-like molecule binding a protein pocket”

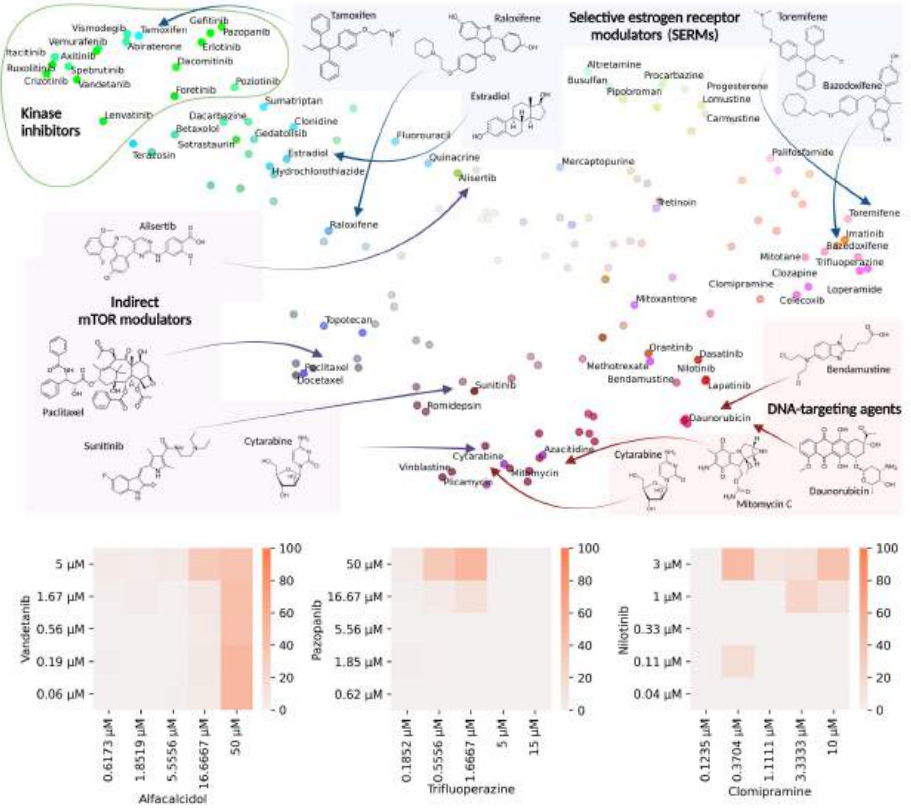
Schneuing et Welling, B, Correia 2022

Drug Repositioning



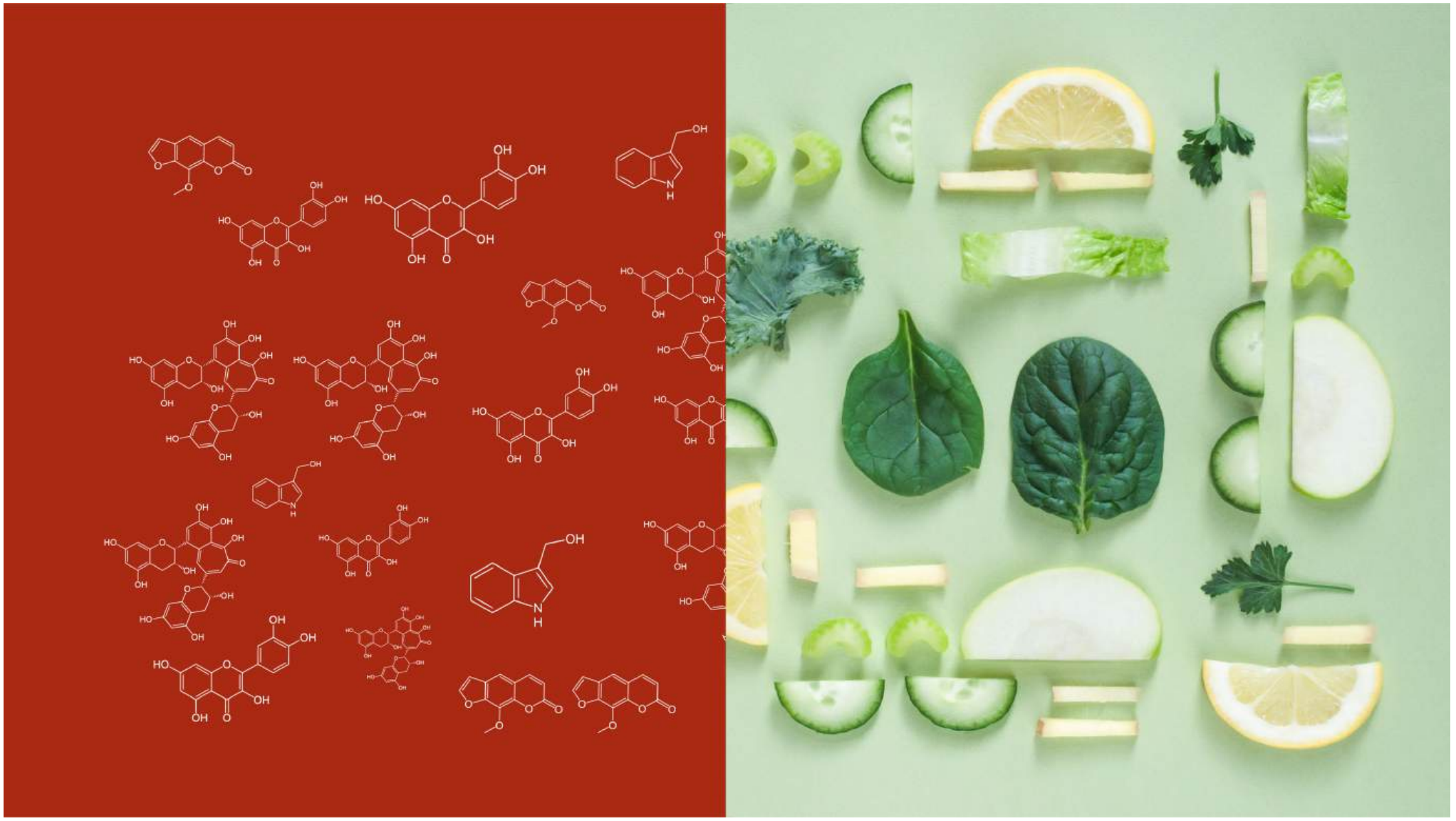
Drug side effect prediction

Zitnik et al. 2018



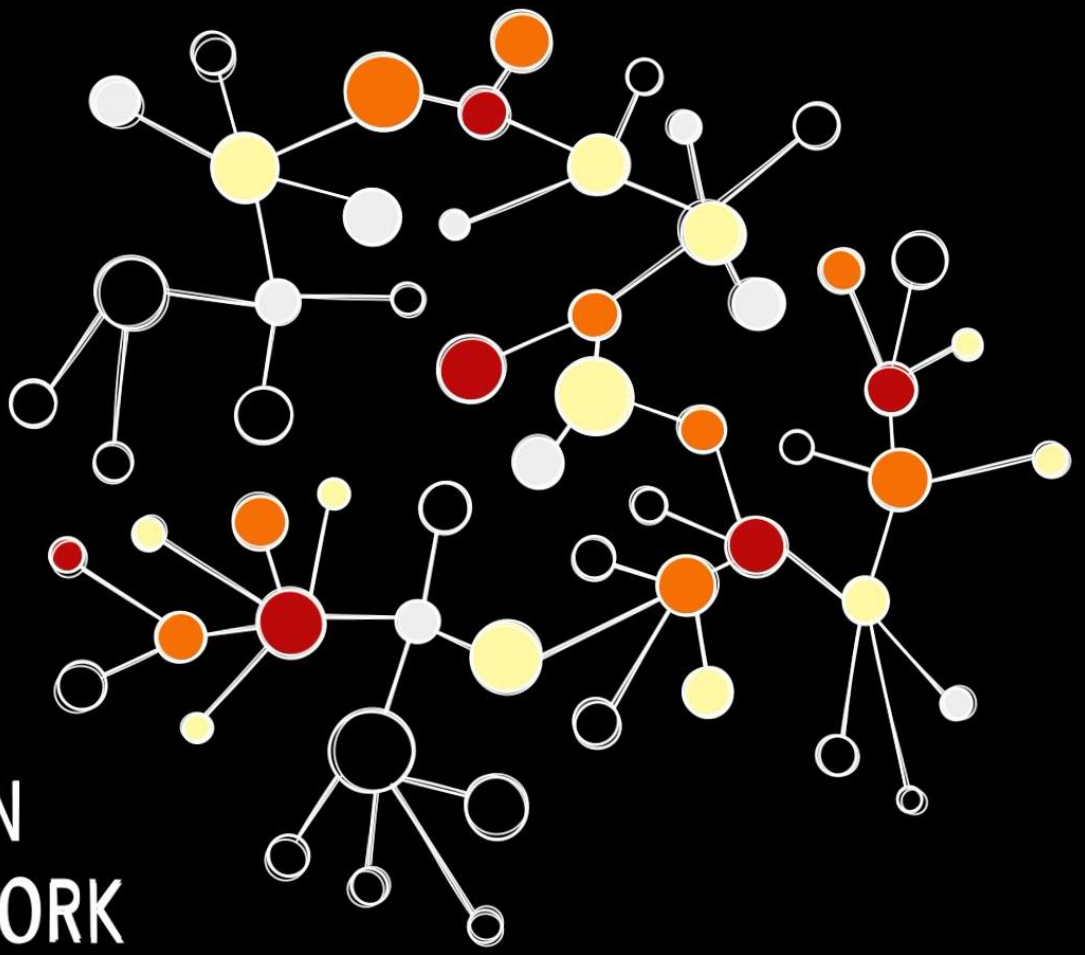
Drug synergy prediction

Bertin et B, Bengio 2021 (RECOVER & Gates Foundation)



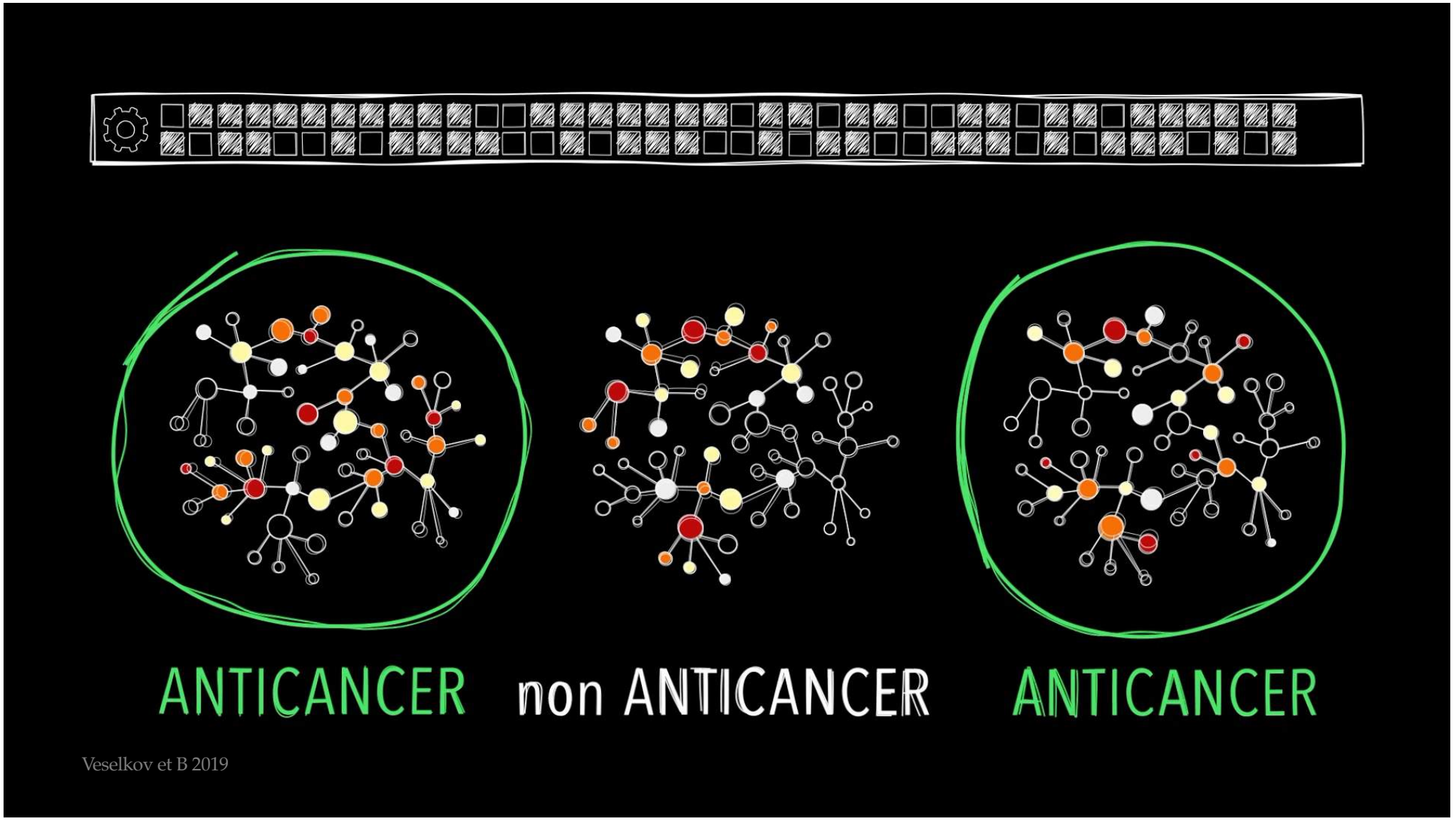

Veselkov et B 2019

Hyperfoods



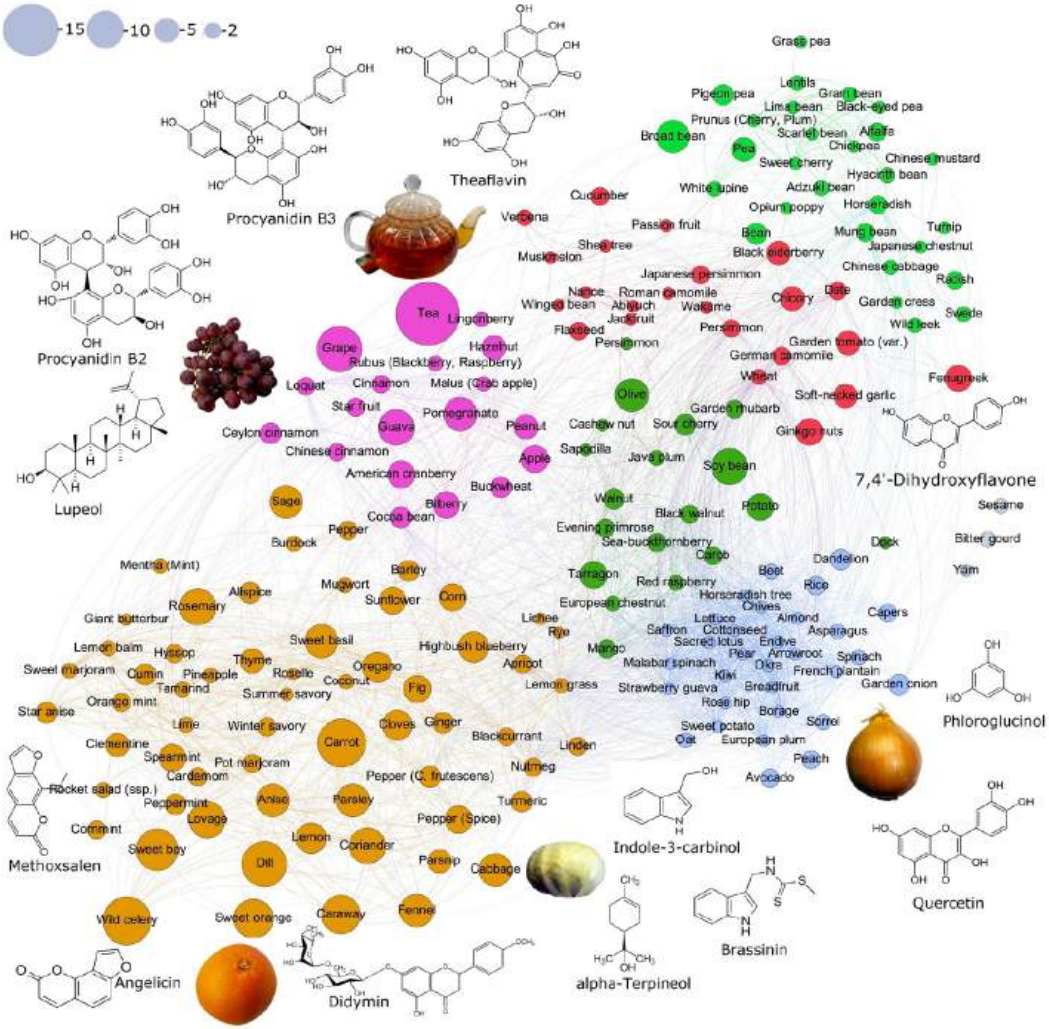
PROTEIN-PROTEIN
INTERACTION NETWORK

Veselkov et B 2019

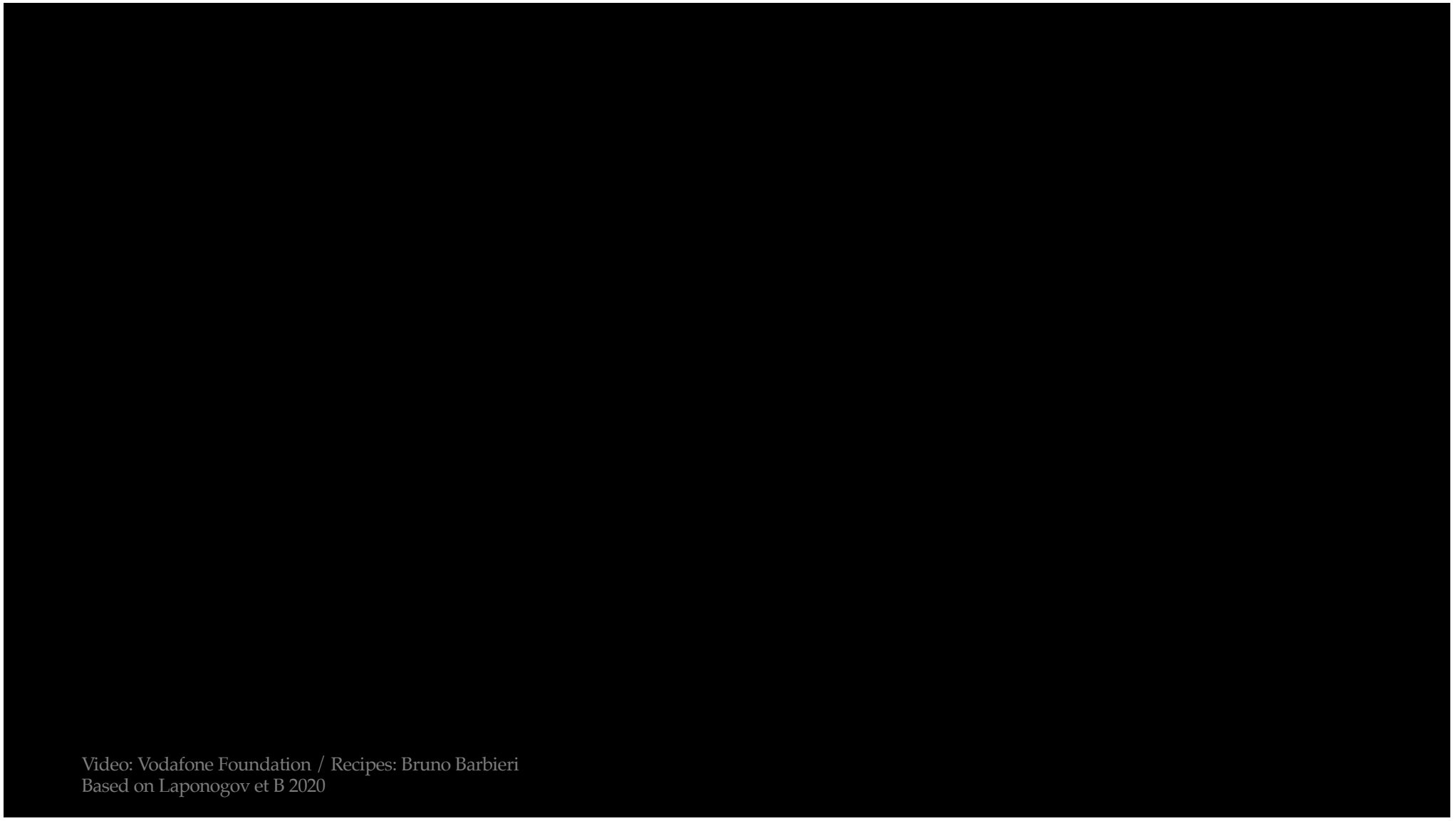


Veselkov et B 2019

Hyperfoods



Veselkov et B 2019



Video: Vodafone Foundation / Recipes: Bruno Barbieri
Based on Laponogov et B 2020

“The knowledge of certain principles easily compensates the lack of knowledge of certain facts”

—Claude Adrien Helvétius


Thank you!