UNIVERSITÄT
HEIDELBERG

Geometric Deep Learning

L5, Structural Bioinformatics

WiSe 2023/24, Heidelberg University

Overview

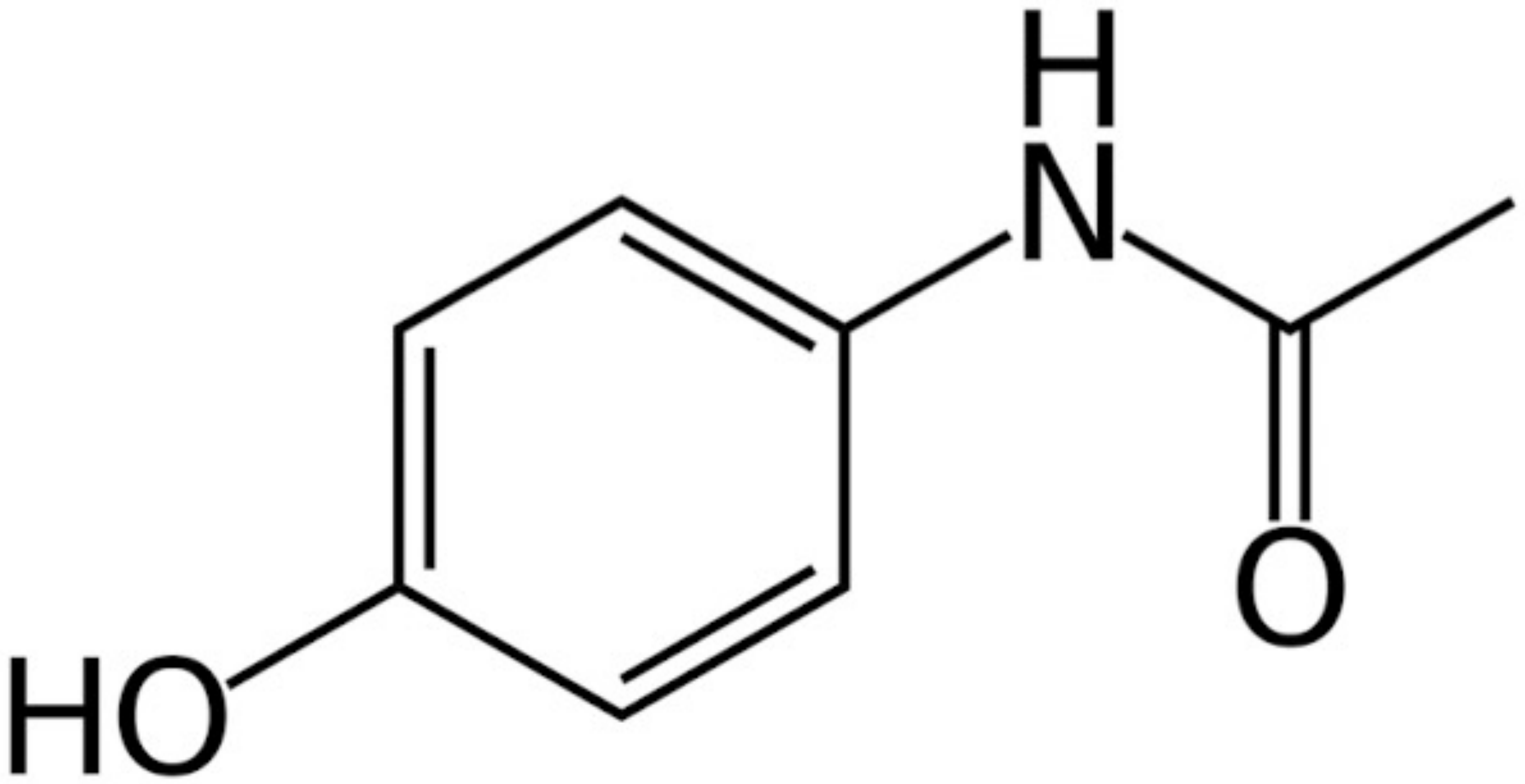
1. Sets and where to find them
2. Graph Neural Networks (GNNs)
3. Geometry and Symmetries
4. Geometric GNNs
5. Outlook to Applications

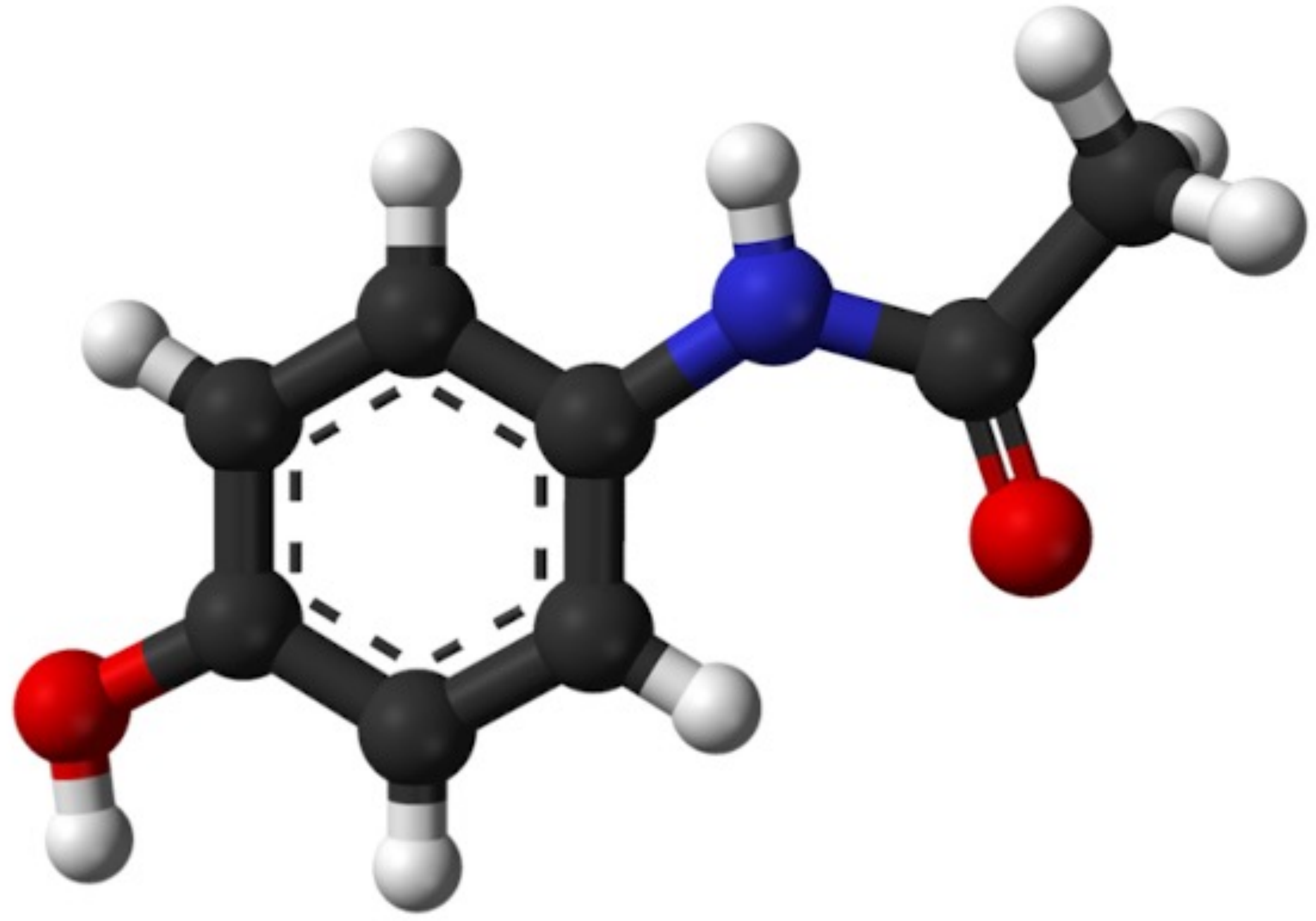
1. Sets and where to find them

Outline of the road ahead

Incorporate relational and then geometric information

$C_8H_9NO_2$





Deep Sets

GNNs

Geometric GNNs

❗ Two problems for molecules:

1. Variable length

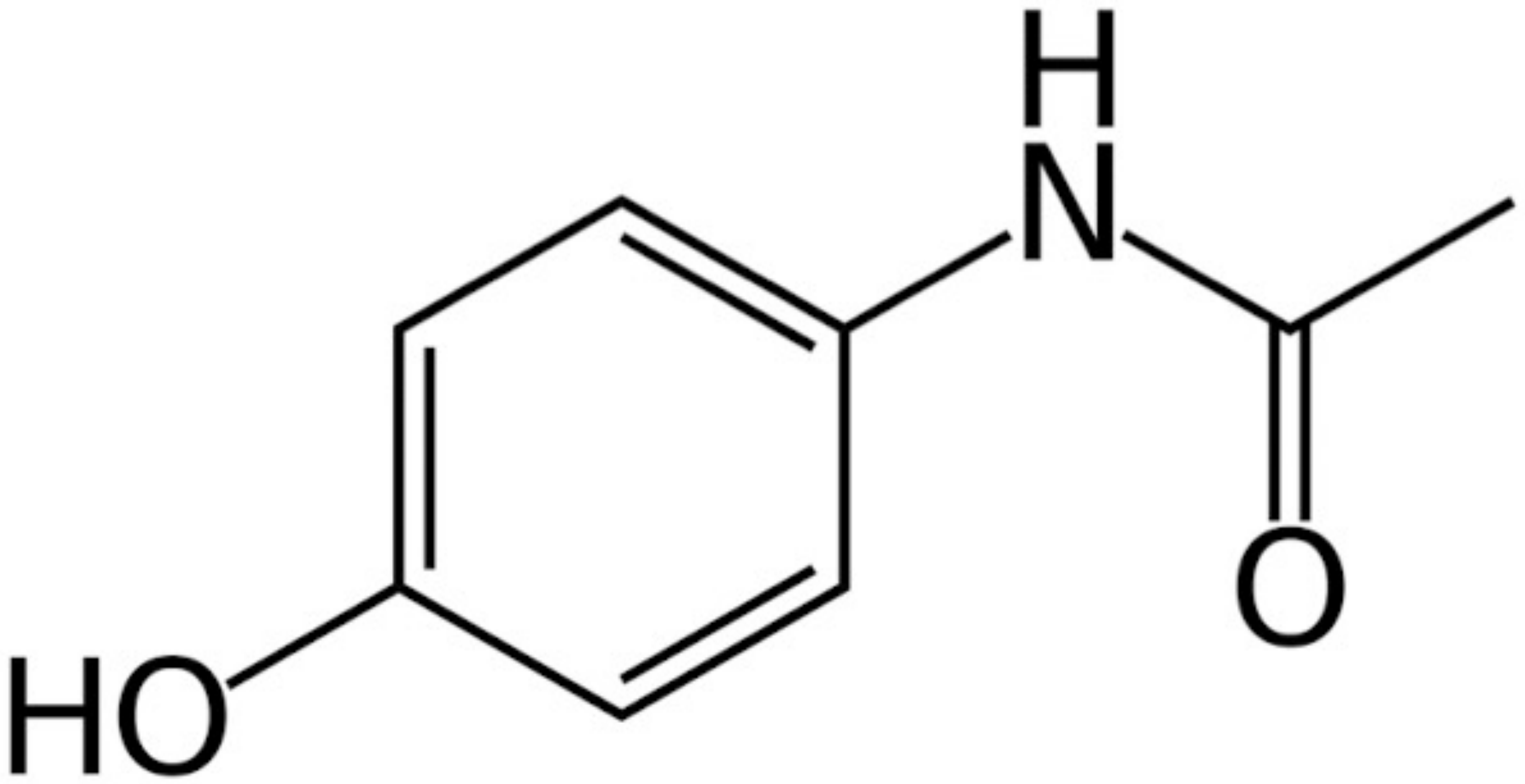
2. Geometric information

Outline of the road ahead

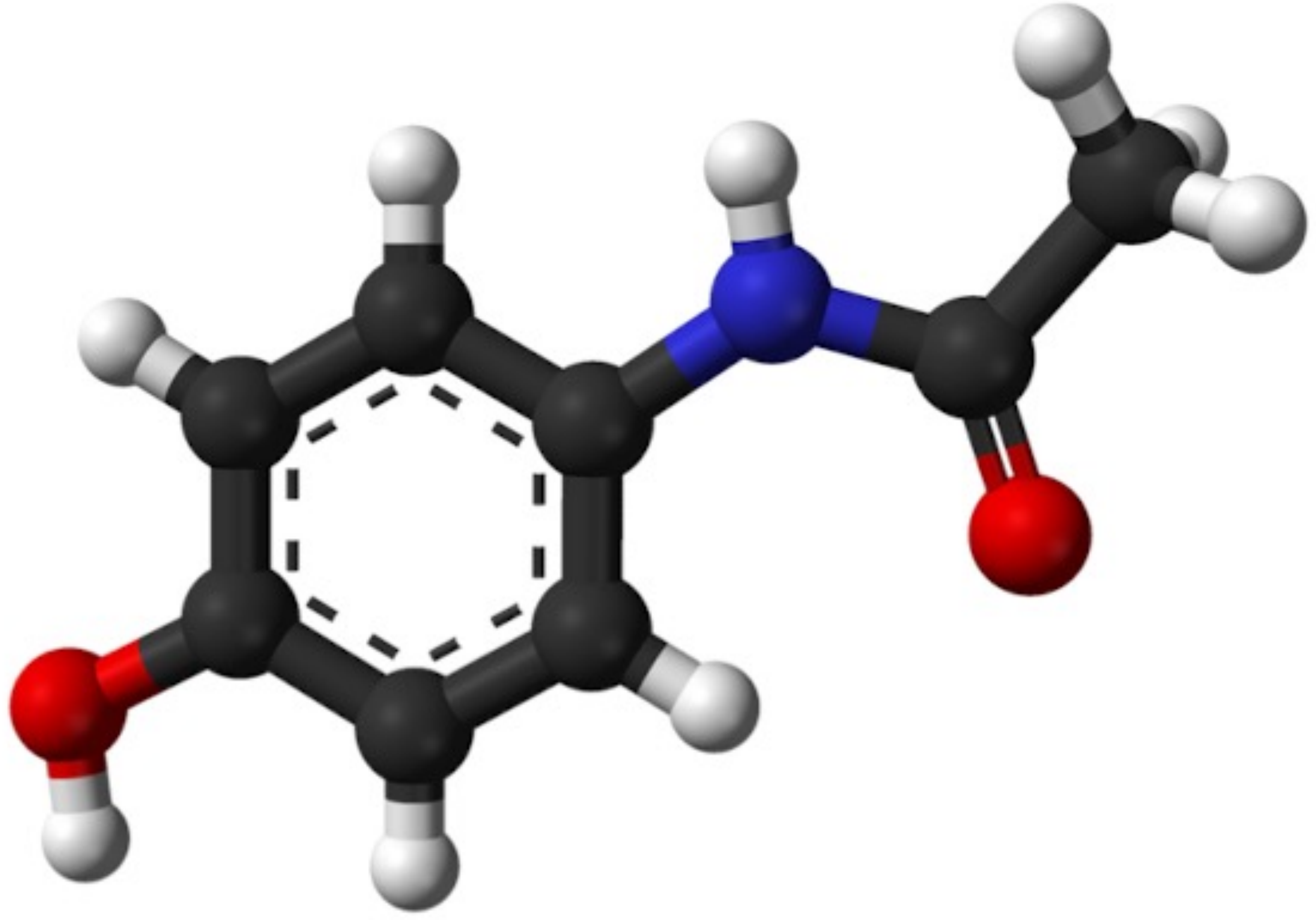
Incorporate relational and then geometric information

$C_8H_9NO_2$

Deep Sets



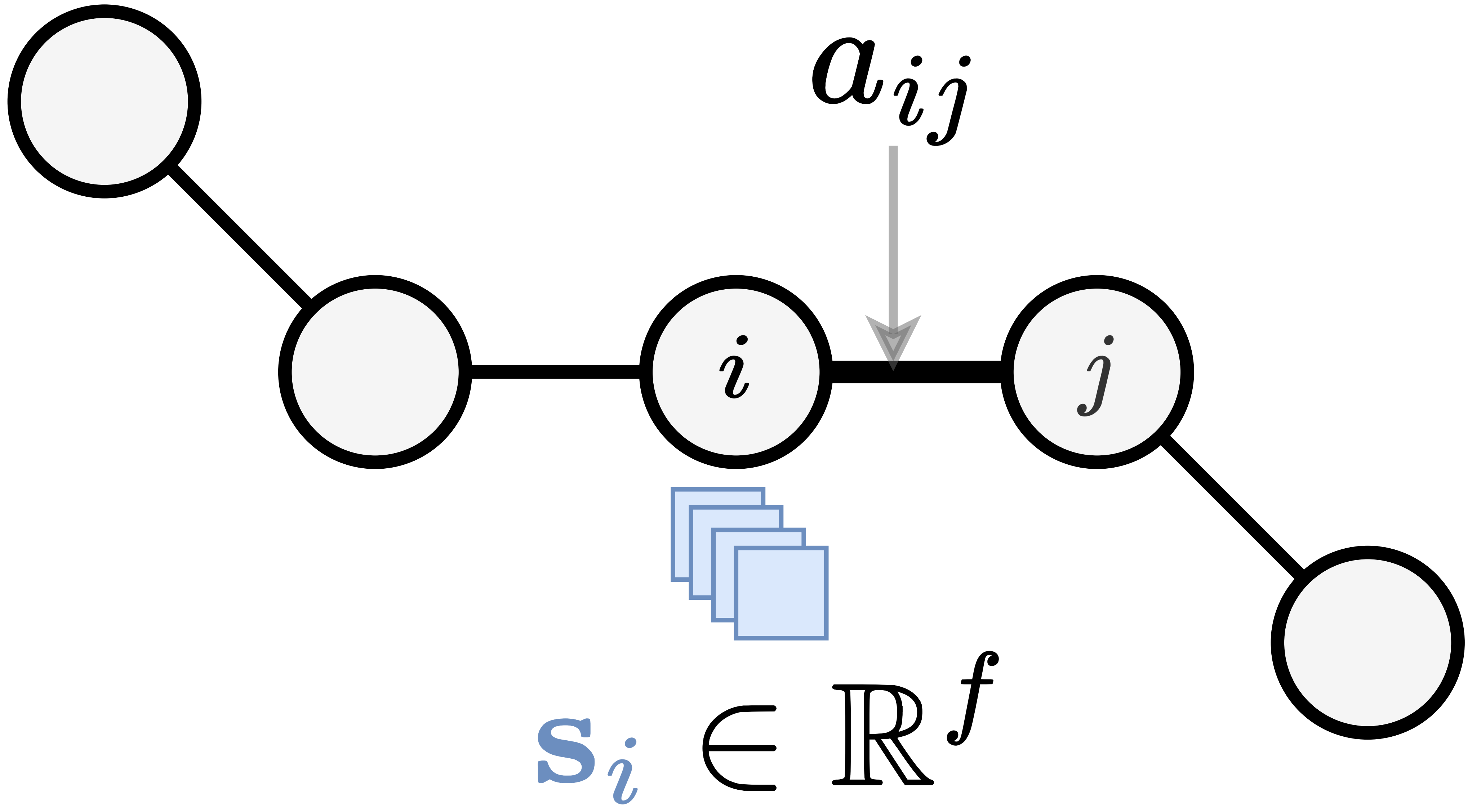
GNNs



Geometric GNNs

Deep Sets

Ignore relational information for now



E.g. atom type

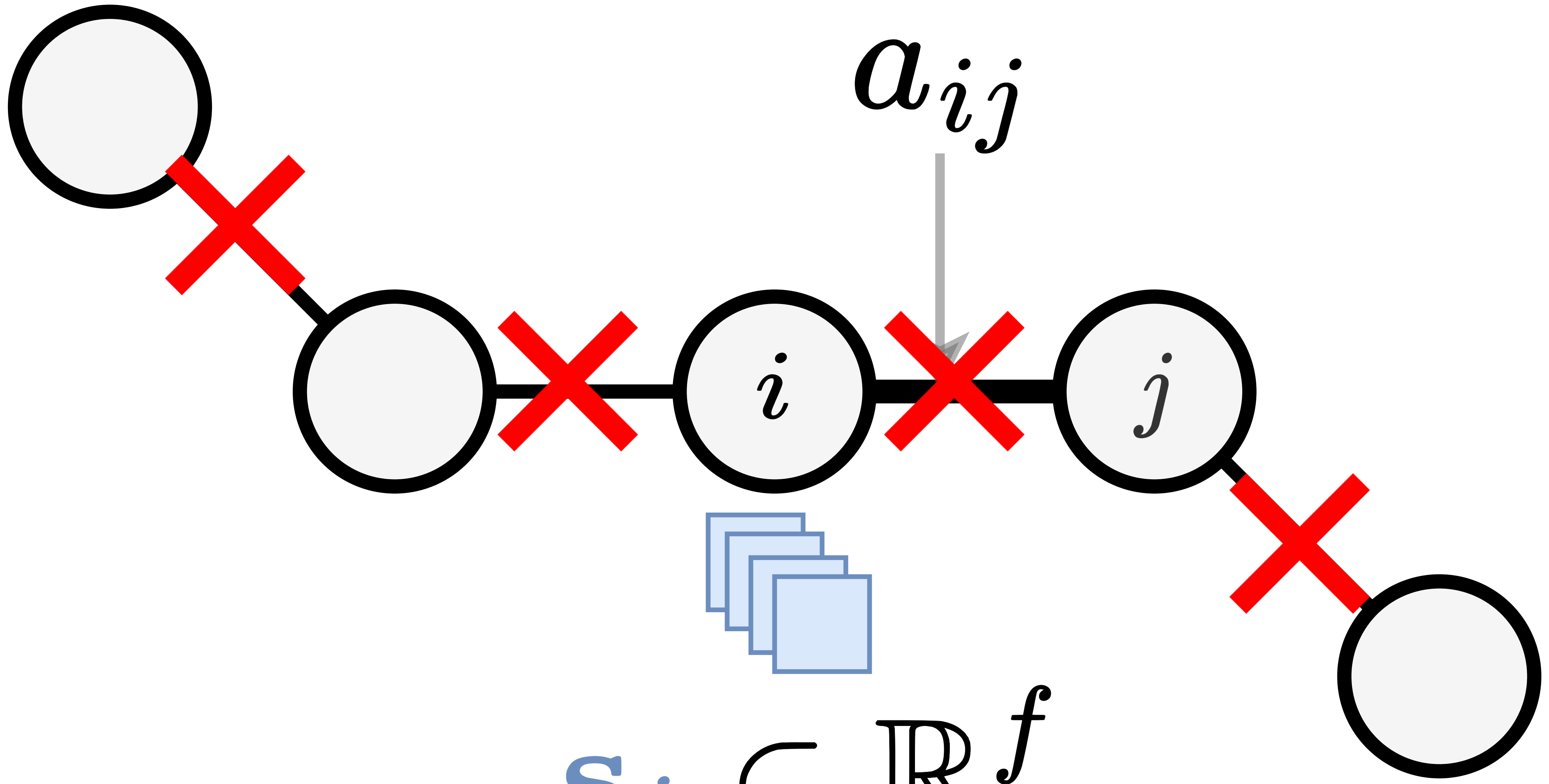
$$\mathcal{G} = (\boldsymbol{A}, \boldsymbol{S})$$

Scalar features $\in \mathbb{R}^{n \times f}$

$n \times n$ adjacency matrix

Deep Sets

Ignore relational information for now



$\mathbf{s}_i \in \mathbb{R}^f$

E.g. atom type

$$\mathcal{G} = (\boldsymbol{A}, \boldsymbol{S})$$

Scalar features $\in \mathbb{R}^{n \times f}$

$n \times n$ adjacency matrix

Deep Sets

Ignore relational information for now

$C_8H_9NO_2$

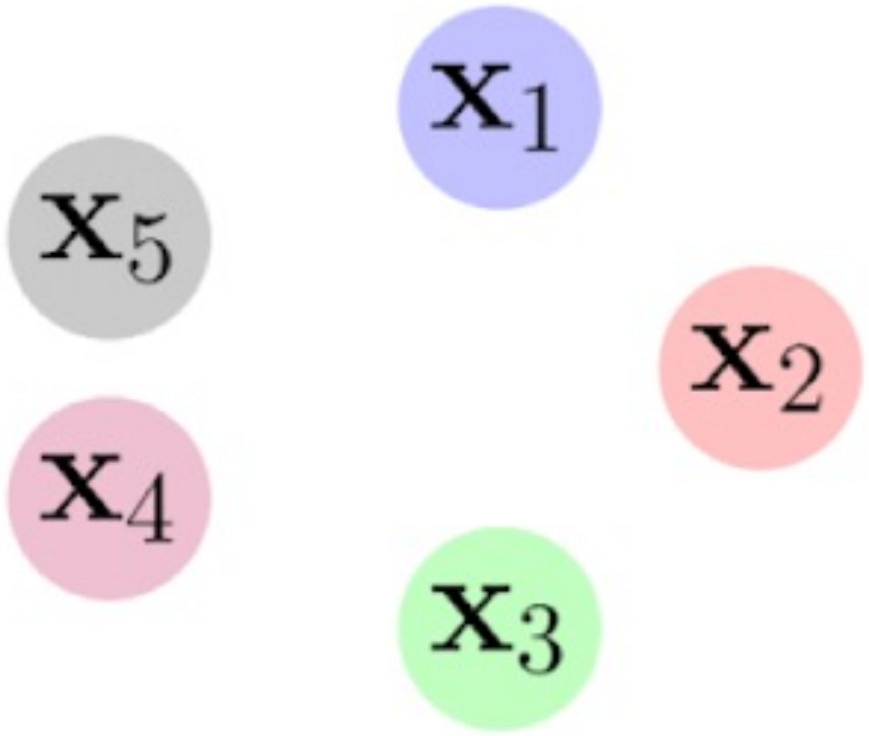
Deep Sets

Ignore relational information for now

$C_8H_9NO_2$

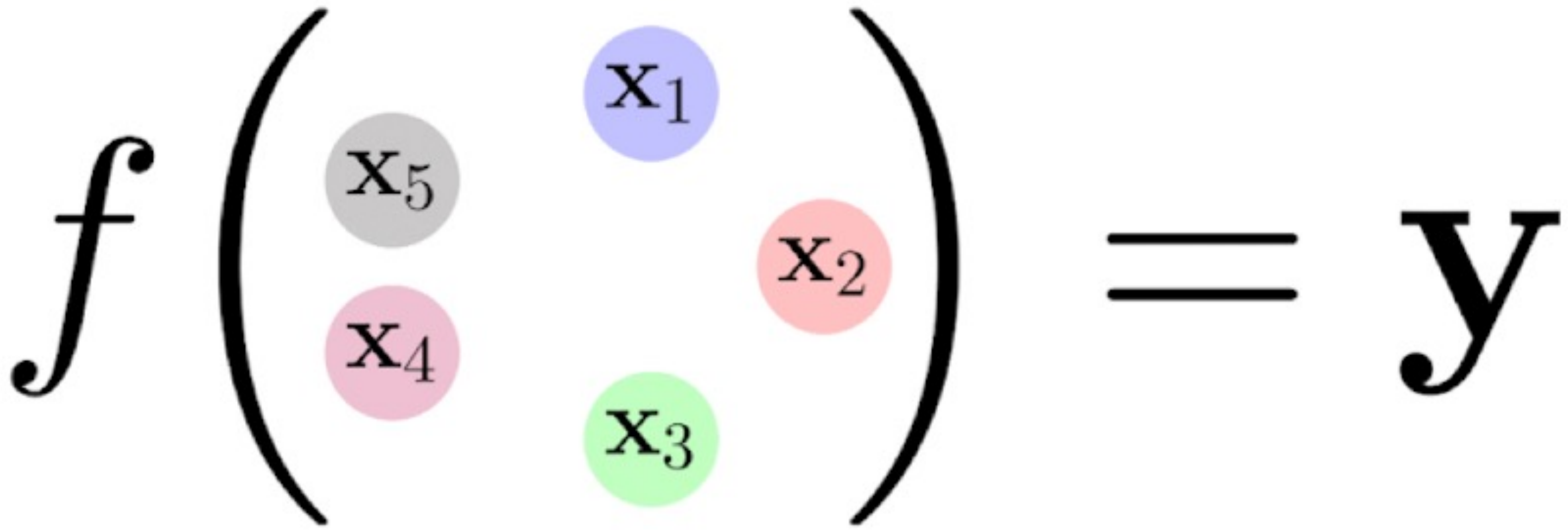
Deep Sets

How do we want our network to behave?



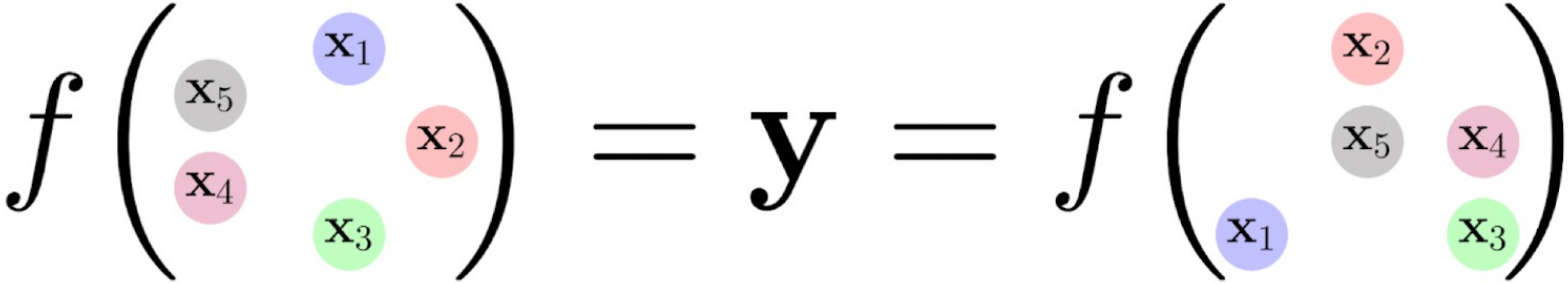
Deep Sets

How do we want our network to behave?



Deep Sets

How do we want our network to behave?



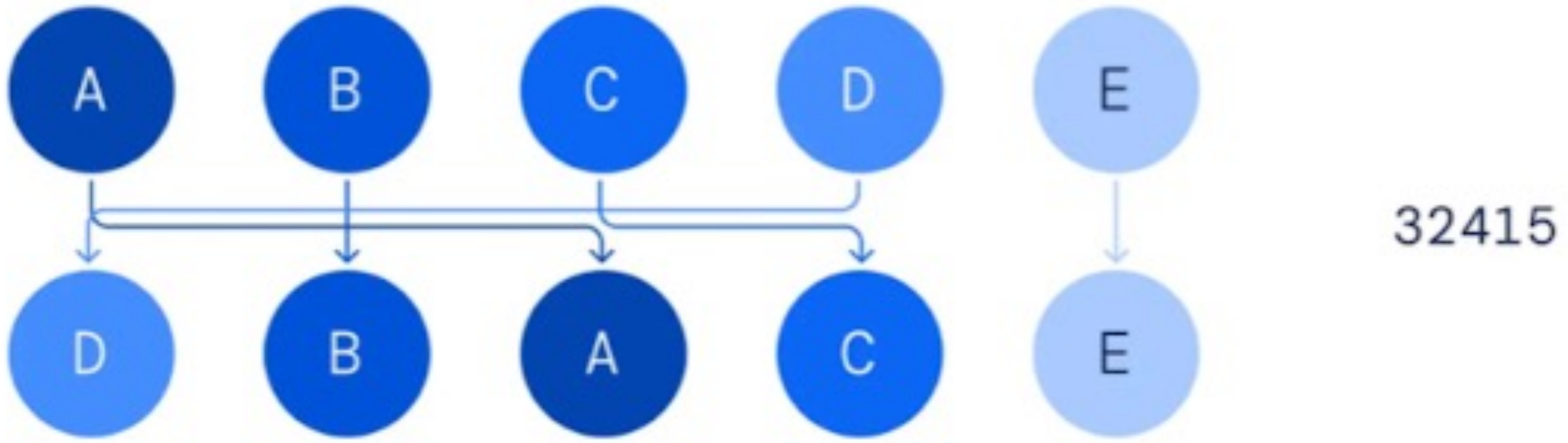
Permutations and Permutation Matrices

Formalising our intuition

It will be useful to think about operators that **change** the node order

Such operations are known as **permutations** (there are $n!$ of them)

e.g. a permutation (2, 4, 1, 3) maps $\mathbf{x}_1 \leftarrow \mathbf{x}_2, \mathbf{x}_2 \leftarrow \mathbf{x}_4, \mathbf{x}_3 \leftarrow \mathbf{x}_1, \mathbf{x}_4 \leftarrow \mathbf{x}_3$



Permutations and Permutation Matrices

Formalising our intuition

Within linear algebra, each permutation defines a $|\mathcal{V}| \times |\mathcal{V}|$ **matrix**

- Such matrices are called **permutation matrices**
- They have exactly one 1 in every row and column, zeroes elsewhere
- Their effect when left-multiplied is to permute rows of $\mathbf{X}$, like so:

$$\mathbf{P}_{(2,4,1,3)}\mathbf{X} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} - & \mathbf{x}_1 & - \\ - & \mathbf{x}_2 & - \\ - & \mathbf{x}_3 & - \\ - & \mathbf{x}_4 & - \end{bmatrix} = \begin{bmatrix} - & \mathbf{x}_2 & - \\ - & \mathbf{x}_4 & - \\ - & \mathbf{x}_1 & - \\ - & \mathbf{x}_3 & - \end{bmatrix}$$

Permutations and Permutation Matrices

Formalising our intuition

Want: functions $f(\mathbf{X})$ over sets that will not depend on the order

Equivalently: applying a permutation matrix shouldn't modify result!

We arrive at a very useful notion of **permutation invariance**.
$f(\mathbf{X})$ is permutation *invariant* if, for *all* permutation matrices $\mathbf{P}$:

$$f(\mathbf{PX}) = f(\mathbf{X})$$

Deep Sets

How do we want our network to behave?

A very generic form is the *Deep Sets* model (Zaheer *et al.*, NeurIPS'17):

$$f(\mathbf{X}) = \phi\left(\bigoplus_{i\in\mathcal{V}} \psi(\mathbf{x}_i)\right)$$

where ψ and ϕ are (learnable) functions, e.g. MLPs.

The **sum** aggregation is *critical*!
(other choices possible, e.g. **max** or **avg**)

We will use $\oplus$ to denote *any* permutation-invariant operator.

Deep Sets

How do we want our network to behave?

Permutation invariant models are good for set-level outputs

What if we would like answers at the **node** level?
We want to still be able to **identify** node outputs, which a permutation-invariant aggregator would destroy!

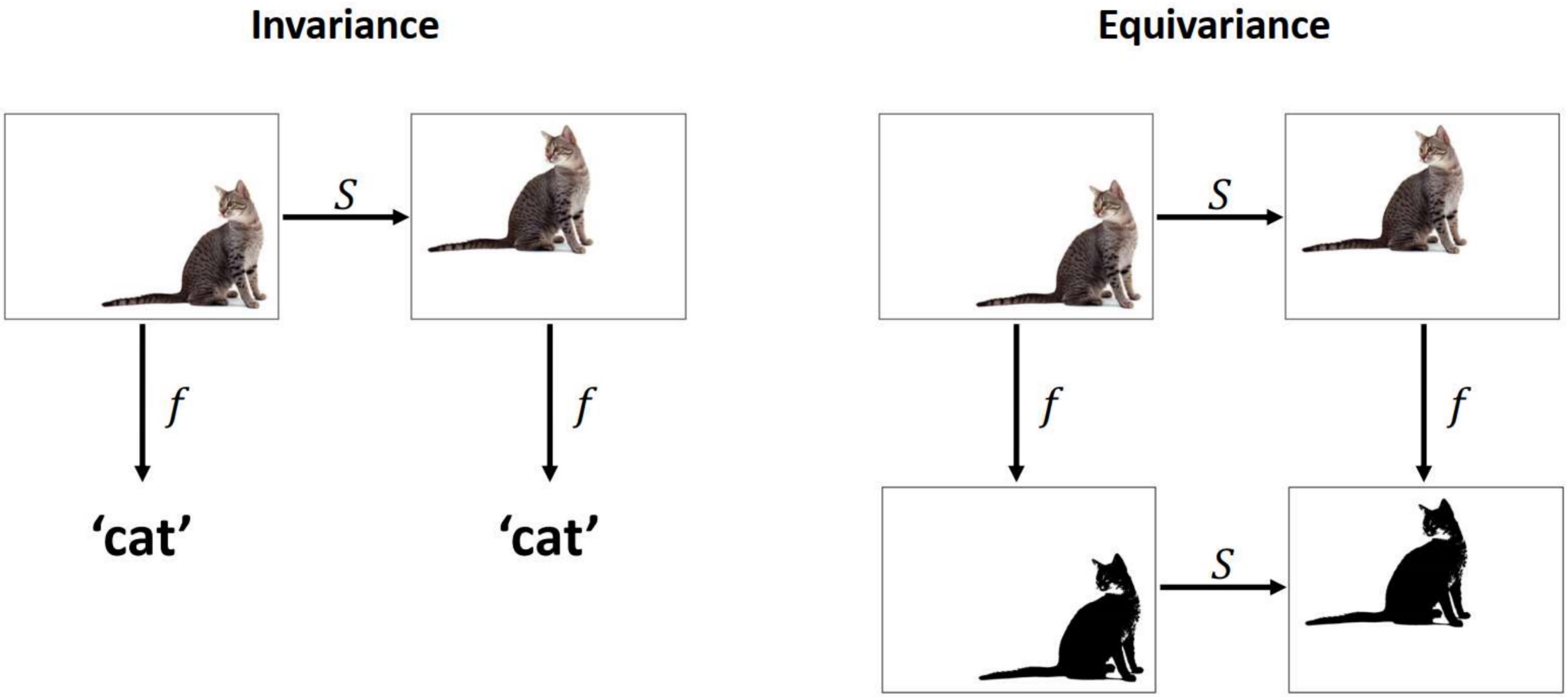
We may instead seek functions that don't **change** the node order i.e. if we permute nodes, it doesn't matter if we do it **before** or **after**!

Accordingly, we say that $\mathbf{F}(\mathbf{X})$ is **permutation equivariant** if, for all permutation matrices $\mathbf{P}$:

$$\mathbf{F}(\mathbf{PX}) = \mathbf{PF}(\mathbf{X})$$

Inductive Bias: Translational In-/Equivariance

Leverage the symmetry of your data



Deep Sets

Analysing the update function

Deep Sets as a *blueprint*: (stacking) **equivariant** function(s), potentially with an **invariant** tail---yields (m)any useful set neural nets!

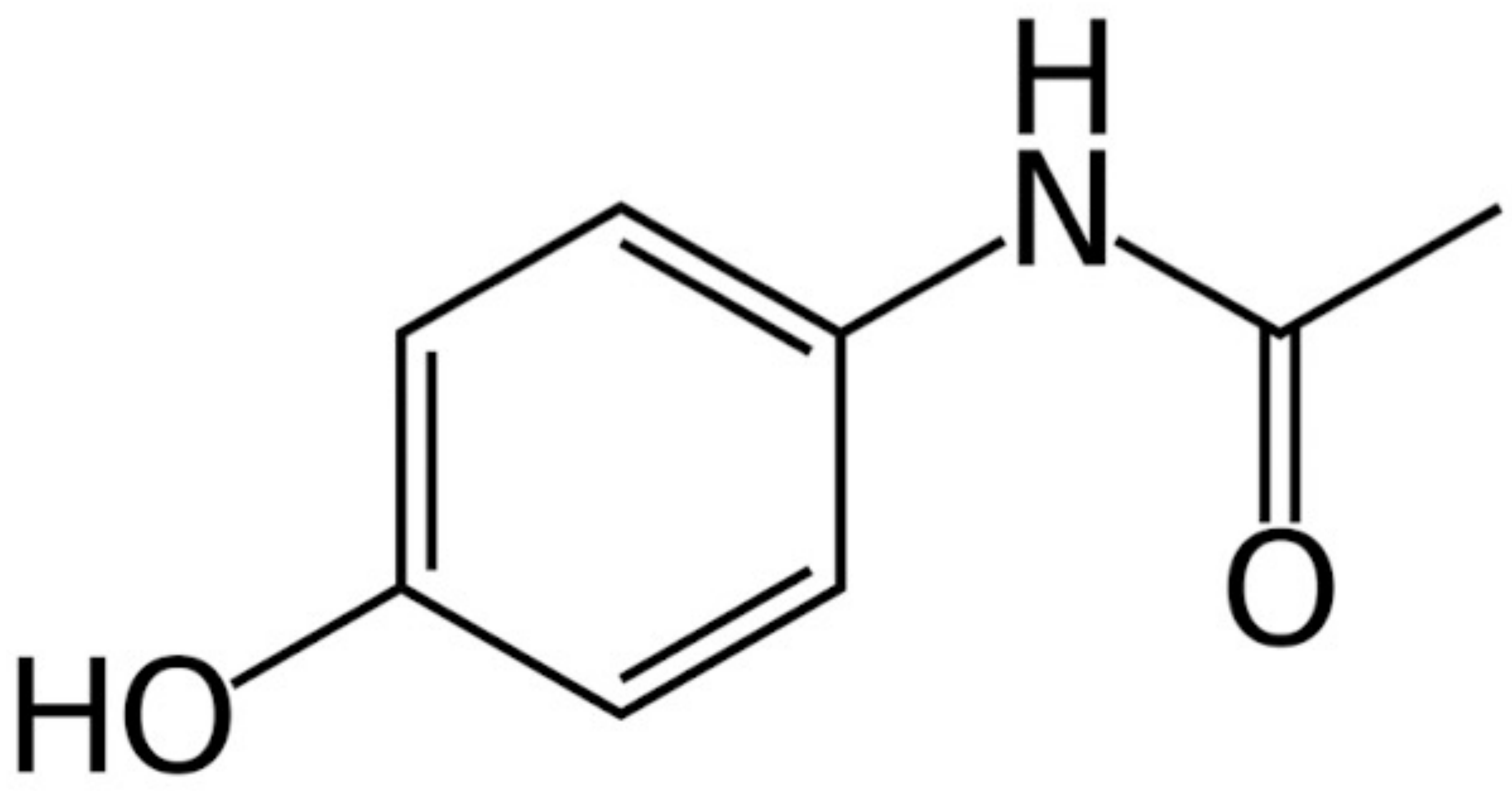
$$f(\mathbf{X}) = \boxed{\phi\left(\bigoplus_{i\in\mathcal{V}} \boxed{\psi(\mathbf{x}_i)}\right)}$$

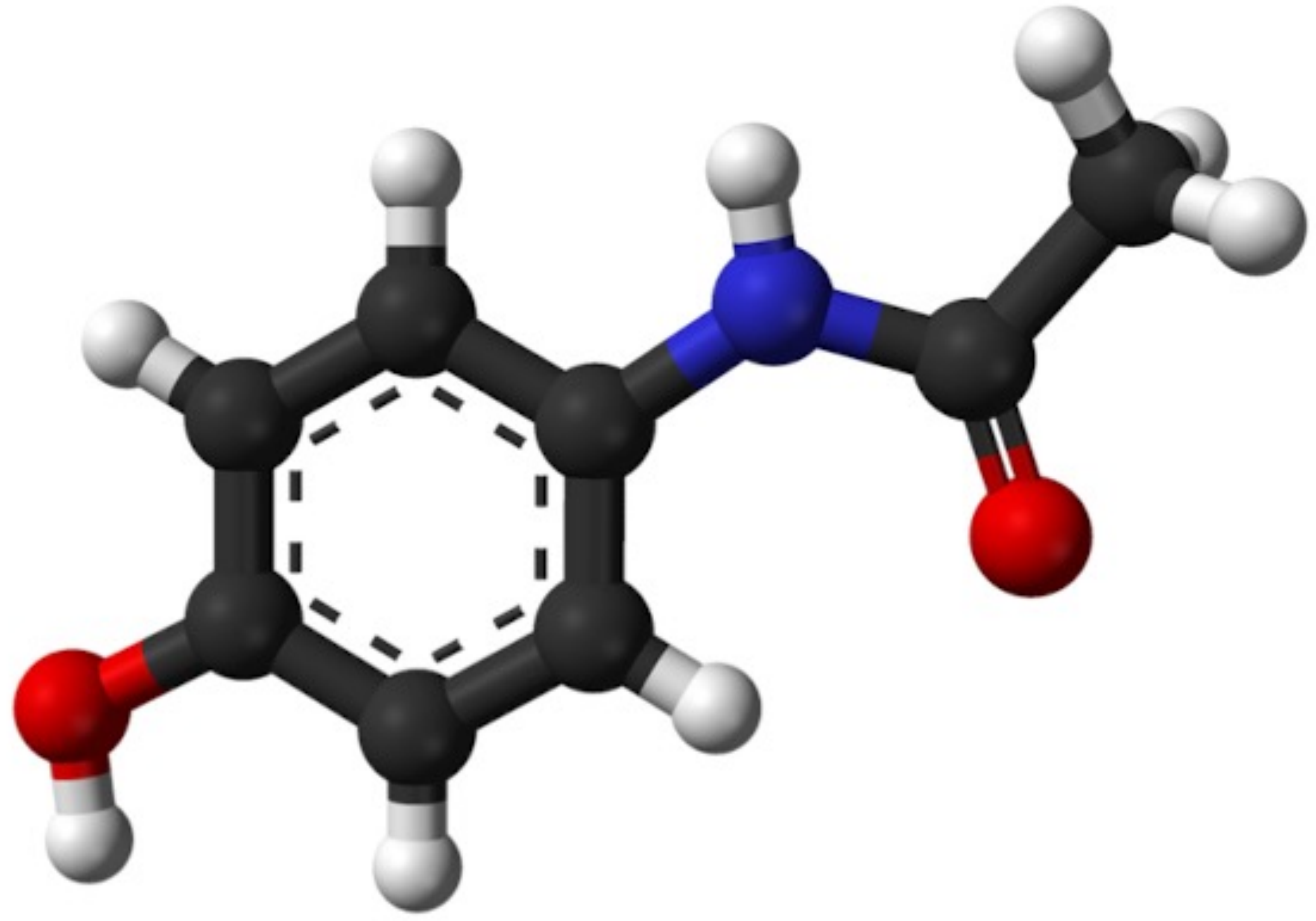
2. Graph Neural Networks (GNNs)

Outline of the road ahead

Incorporate relational and then geometric information

$C_8H_9NO_2$





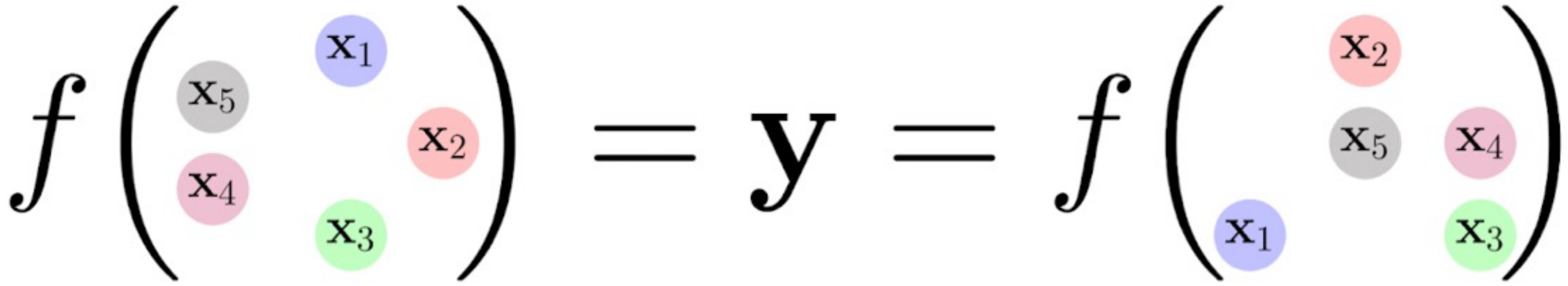
Deep Sets

GNNs

Geometric GNNs

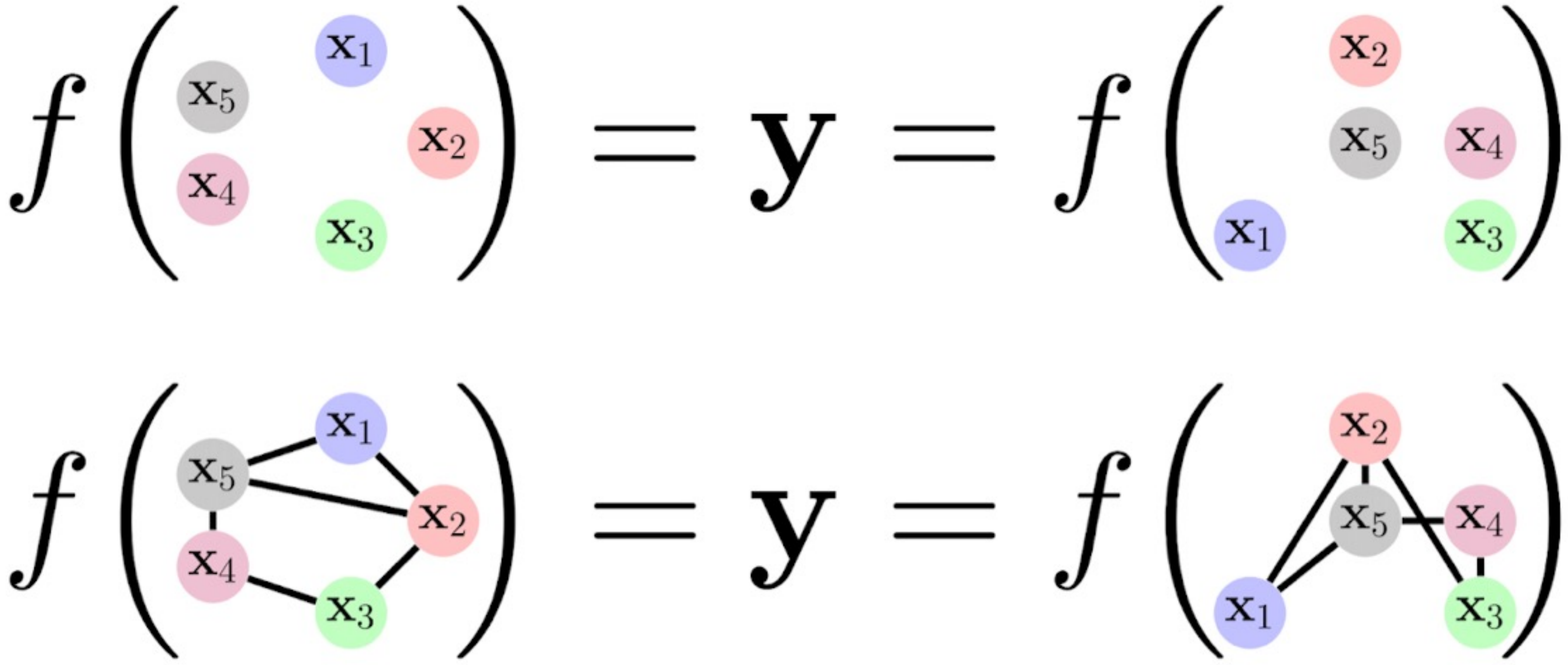
What has changed?

We need to think about the edges as well!



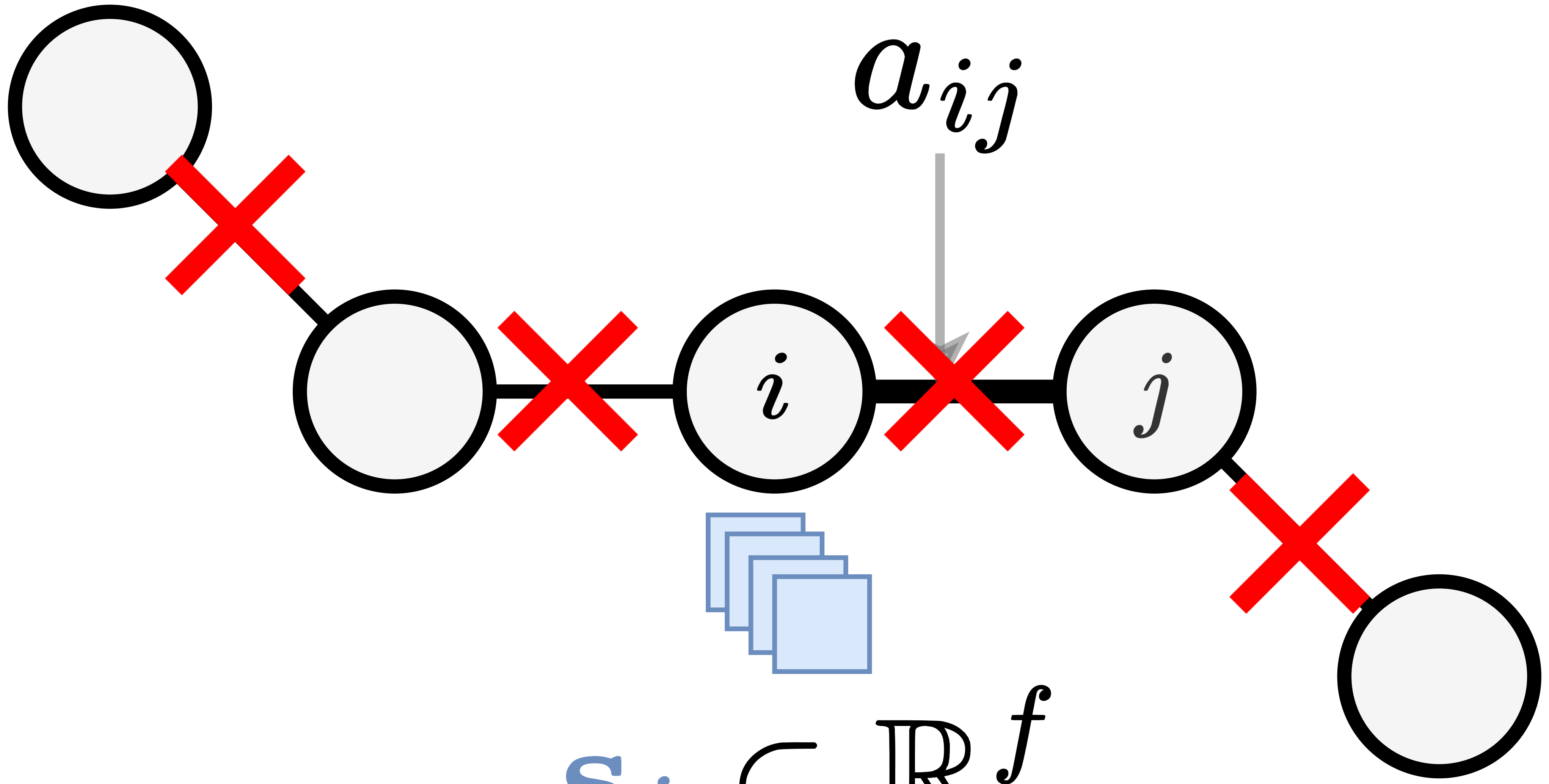
What has changed?

We need to think about the edges as well!



From Sets to Graph

Relational Information is back in the game



$\mathbf{s}_i \in \mathbb{R}^f$

E.g. atom type

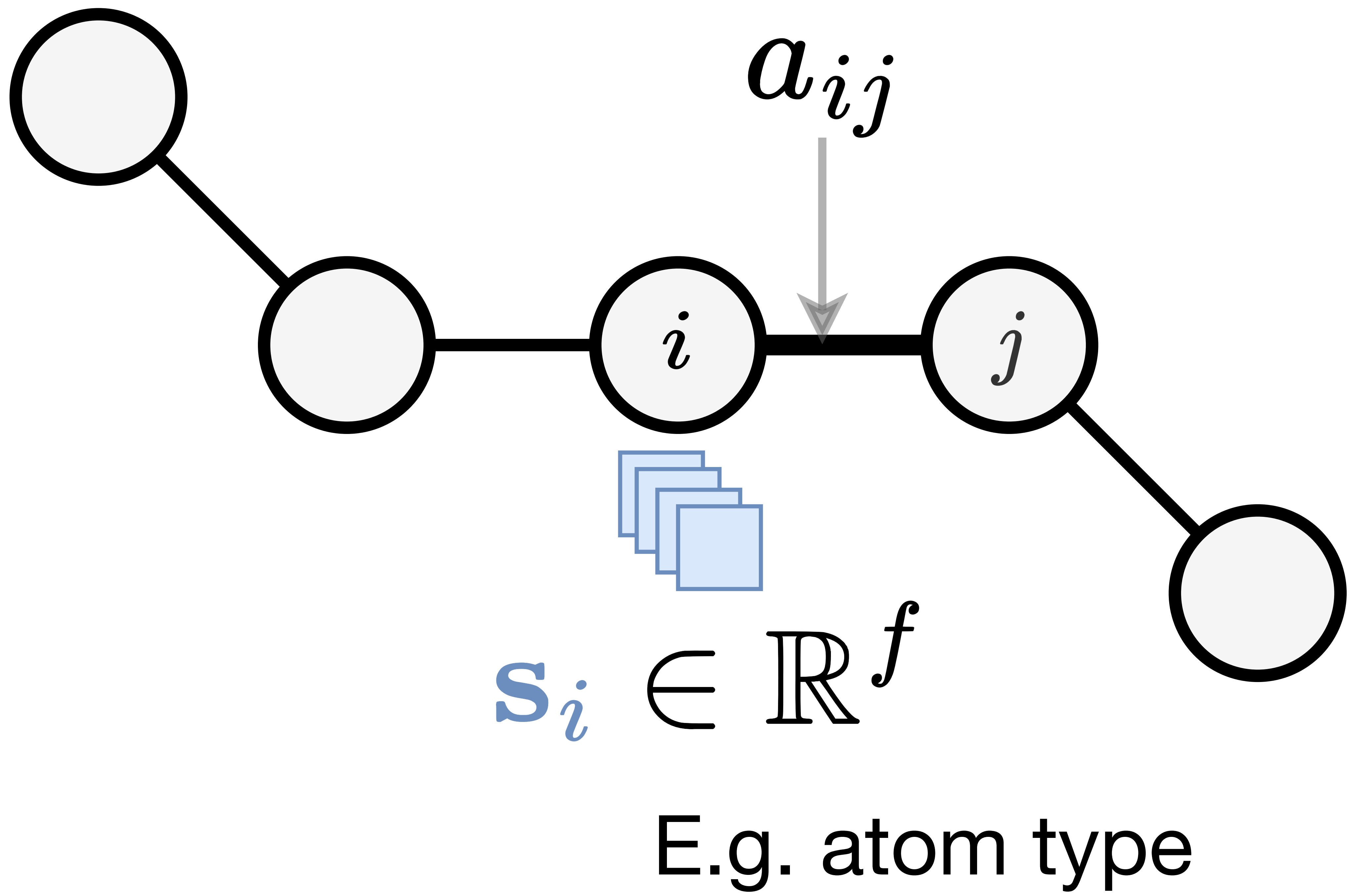
$$\mathcal{G} = (\boldsymbol{A}, \boldsymbol{S})$$

Scalar features $\in \mathbb{R}^{n \times f}$

$n \times n$ adjacency matrix

From Sets to Graph

Relational Information is back in the game



$$\mathcal{G} = (\boldsymbol{A}, \boldsymbol{S})$$

Scalar features $\in \mathbb{R}^{n \times f}$

$n \times n$ adjacency matrix

Message-passing

Tell your neighbour what you know

Main difference: permutations now also accordingly act on the **edges**

We need to appropriately permute both **rows** and **columns** of $\mathbf{A}$
When applying a permutation matrix $\mathbf{P}$, this amounts to $\mathbf{PAP}^\top$

We arrive at updated definitions of suitable functions over graphs:

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

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

Message-passing

Tell your neighbour what you know

On *sets*, we enforced *locality* by transforming every node **in isolation**

Graphs give us a broader context: a node's *neighbourhood*
For a node i, its (1-hop) neighbourhood, $\mathcal{N}_i$, is commonly defined as:

$$\mathcal{N}_i = \{ j : (i,j) \in \mathcal{E} \lor (j,i) \in \mathcal{E} \}$$

Accordingly, we can extract neighbourhood features, $\mathbf{X}_{\mathcal{N}_i}$, like so:

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

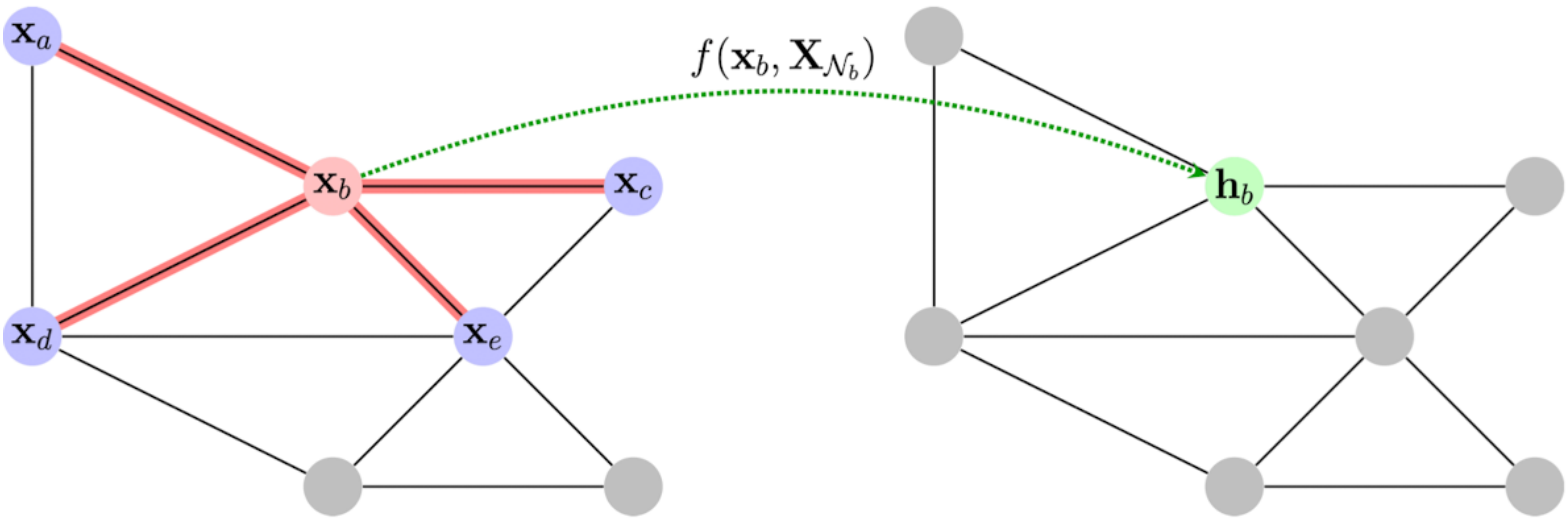
important to not loose identical neighbours, would happen with simple set

and define a *local* function, $f(\mathbf{x}_i, \mathbf{X}_{\mathcal{N}_i})$, operating over them.

($\mathbf{X}_{\mathcal{N}_i}$ is a *multiset*; cf. {{ ... }} notation)

Message-passing

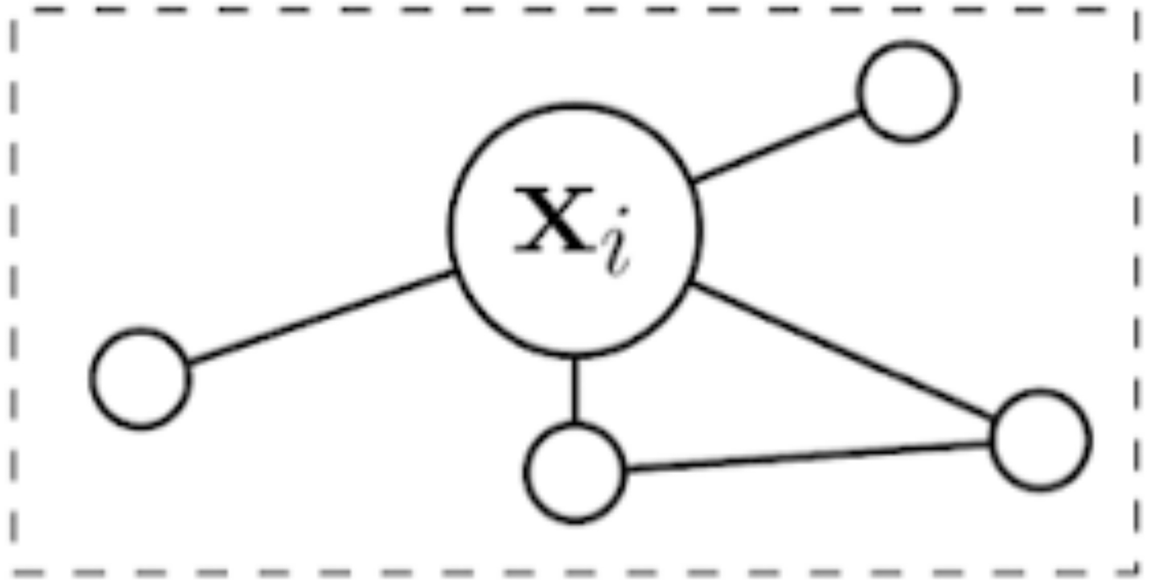
Tell your neighbour what you know



$$\mathbf{X}_{\mathcal{N}_b} = \{\!\{\mathbf{x}_a, \mathbf{x}_b, \mathbf{x}_c, \mathbf{x}_d, \mathbf{x}_e\}\!\}$$

Message-passing

A general framework for all graph-related predictions

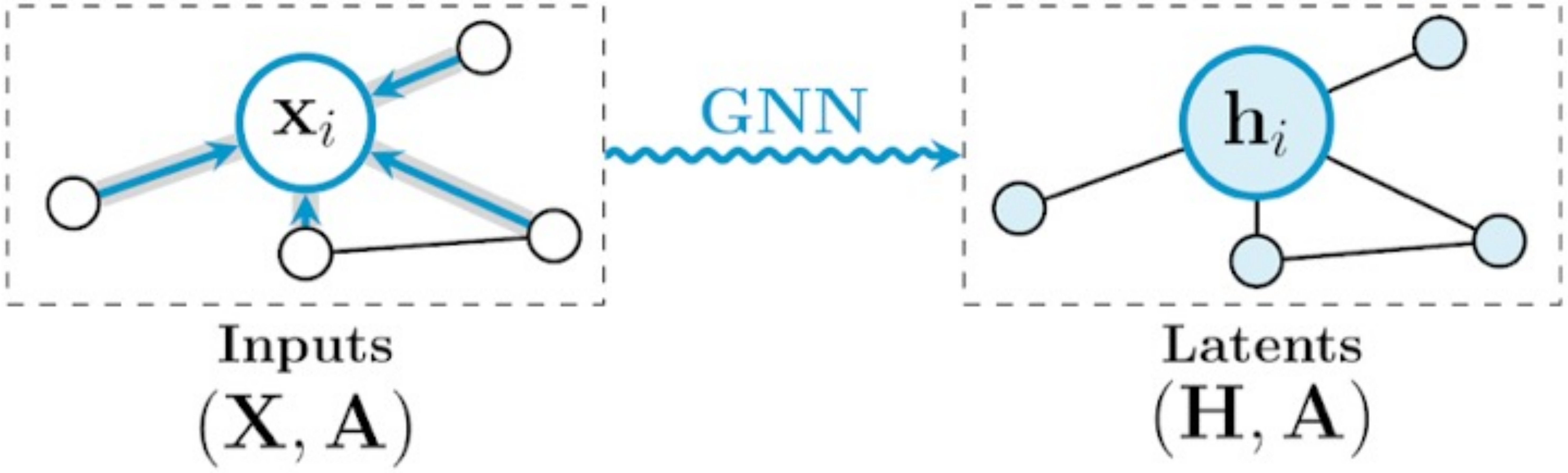


Inputs

$(\mathbf{X}, \mathbf{A})$

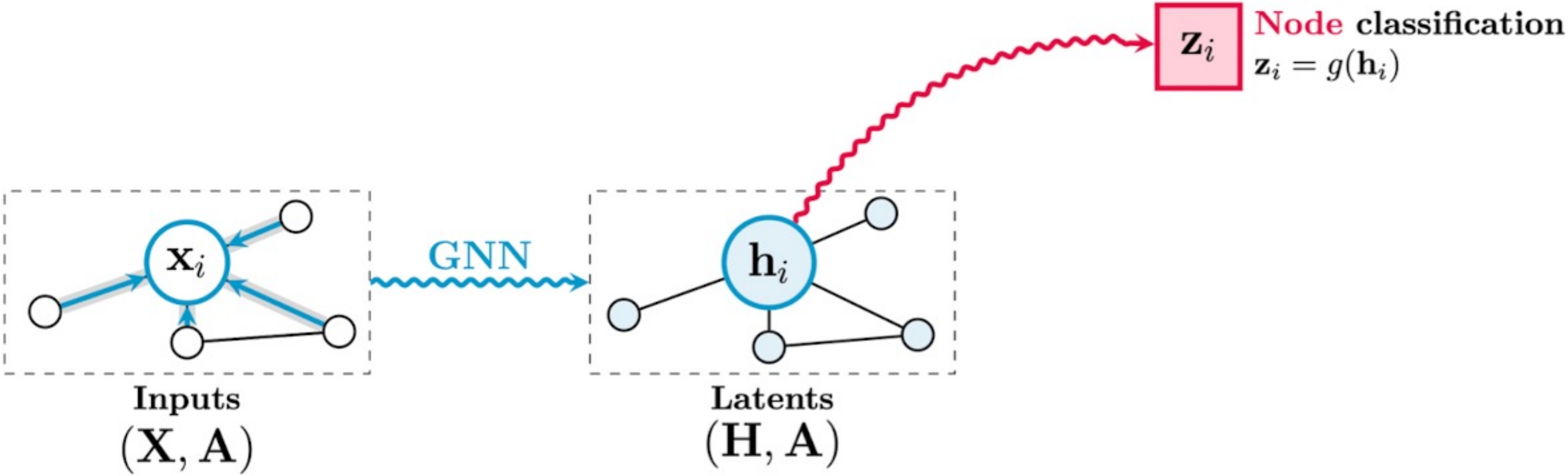
Message-passing

A general framework for all graph-related predictions



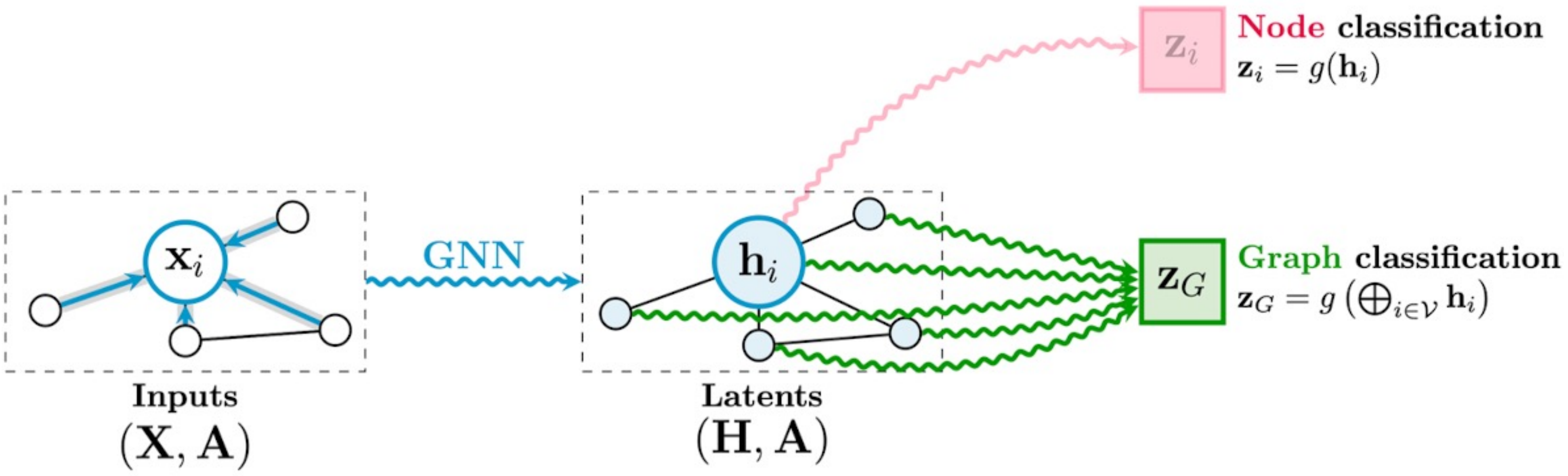
Message-passing

A general framework for all graph-related predictions



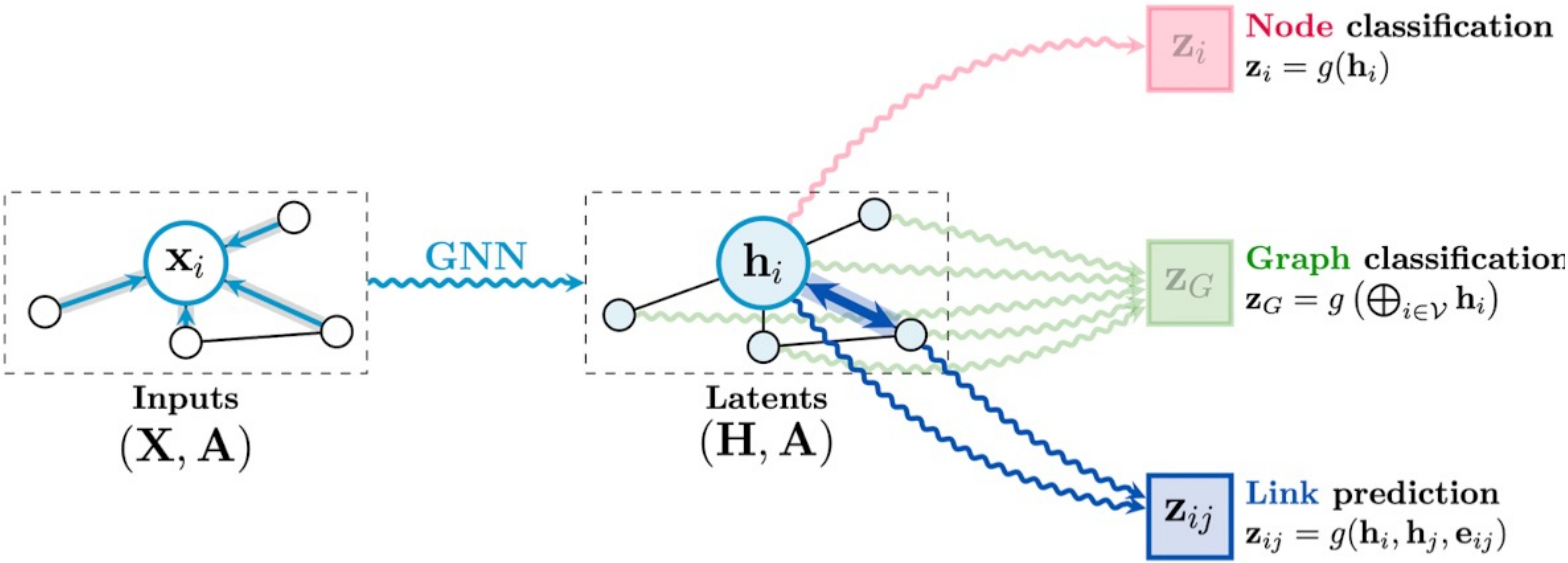
Message-passing

A general framework for all graph-related predictions



Message-passing

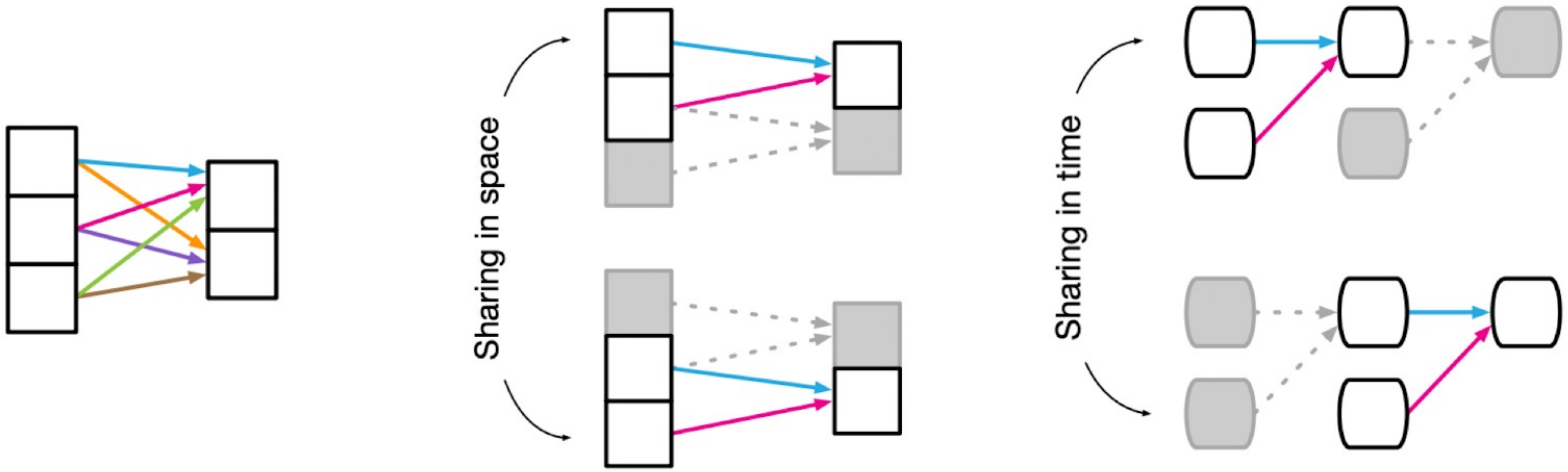
A general framework for all graph-related predictions



Petar Veličković

A unifying framework

Other architectures can be seen as message-passing GNNs!



(a) Fully connected (b) Convolutional (c) Recurrent

How to implement message passing?

Remember our DeepSet insights

As f is supposed to be a local and permutation-invariant function over the neighbourhood features $\mathbf{X}_{\mathcal{N}_i}$, it effectively needs to be a neural network over **sets**, potentially conditioned by $\mathbf{x}_i$.

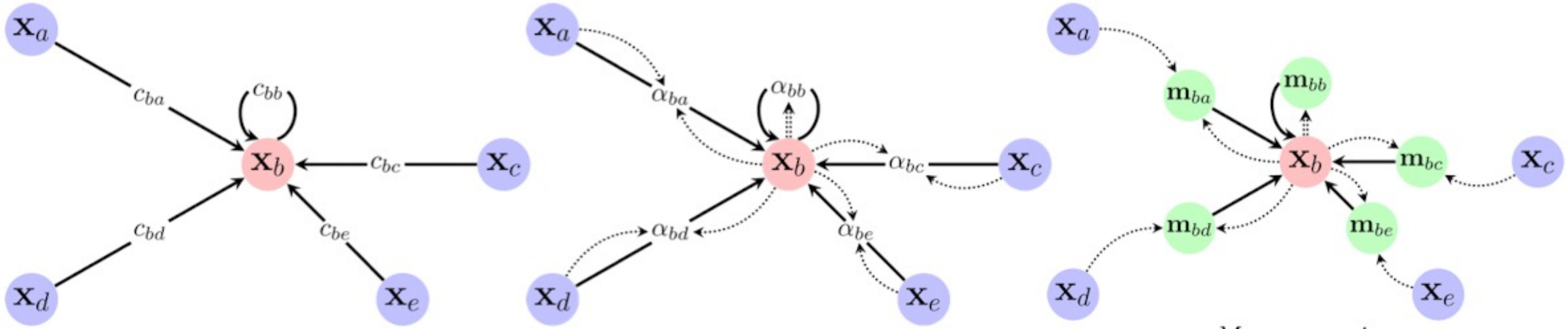
Recalling the Deep Sets model and its universality, we can hence assume the following generic equation (with added conditioning):

$$f(\mathbf{x}_i, \mathbf{X}_{\mathcal{N}_i}) = \phi\left(\mathbf{x}_i, \bigoplus_{j \in \mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$$

Note that this induces several free variables $(\mathcal{N}_i, \bigoplus, \phi, \psi)$

The classic landscape

One architecture per community



Convolutional *Attentional* *Message-passing*

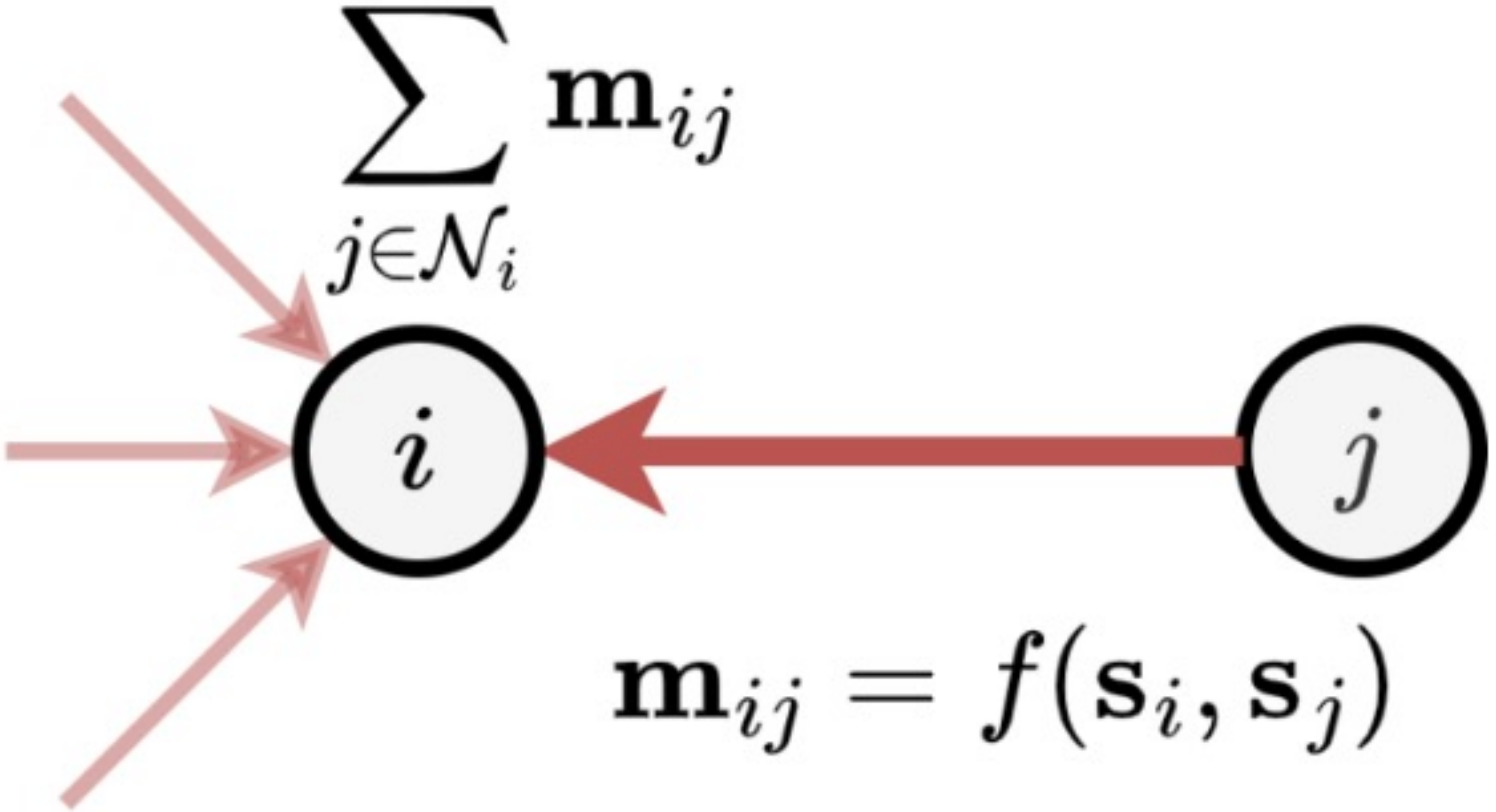
$$\mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j\in\mathcal{N}_i} c_{ij}\psi(\mathbf{x}_j)\right) \qquad \mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j\in\mathcal{N}_i} a(\mathbf{x}_i, \mathbf{x}_j)\psi(\mathbf{x}_j)\right) \qquad \mathbf{h}_i = \phi\left(\mathbf{x}_i, \bigoplus_{j\in\mathcal{N}_i} \psi(\mathbf{x}_i, \mathbf{x}_j)\right)$$

Message-passing

Tell your neighbour what you know

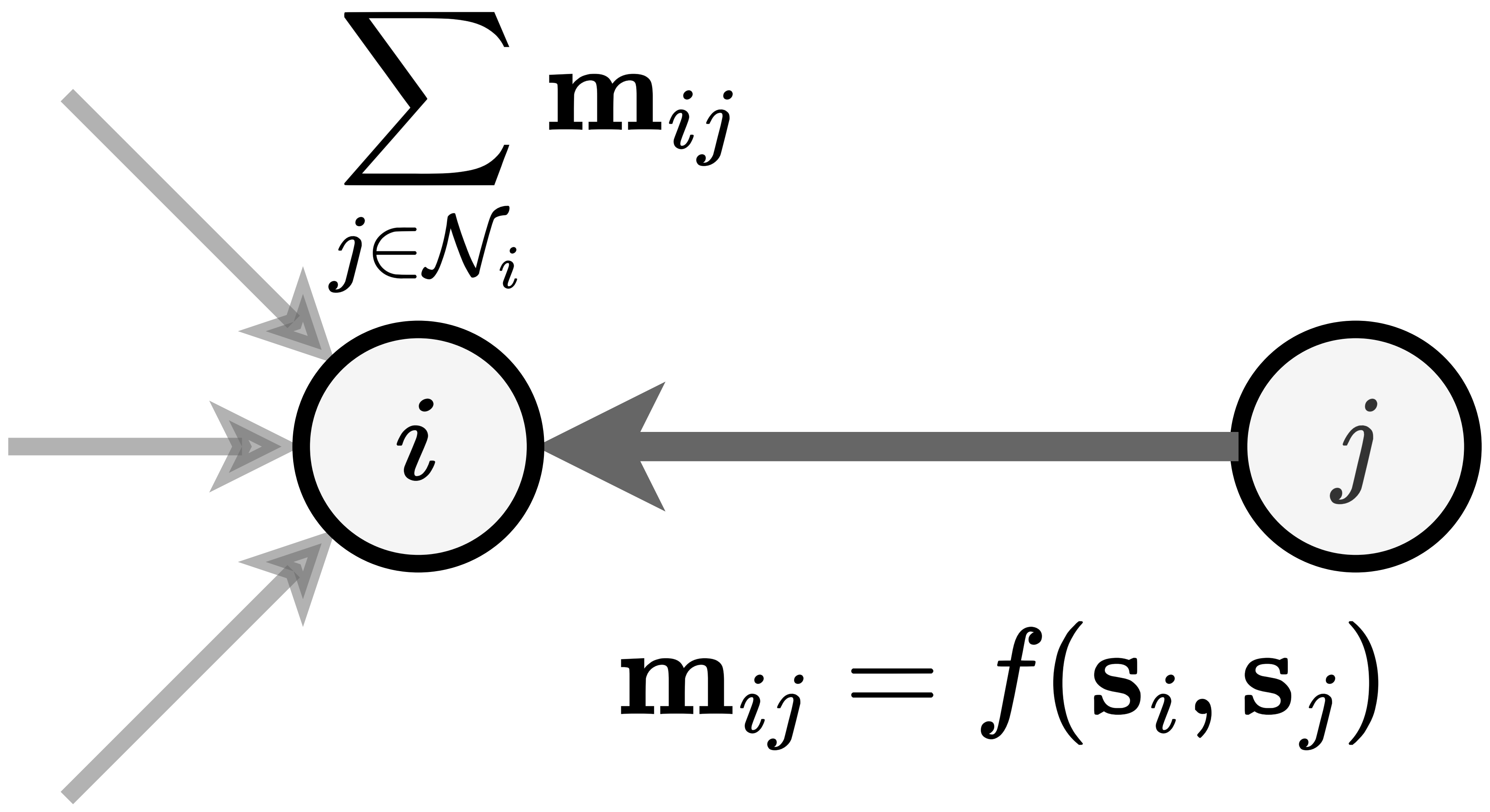
$$\boldsymbol{m}_i^{(t)} = \textsc{Agg}\left(\{\!\{(\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}) \mid j \in \mathcal{N}_i\}\!\}\right)$$

$$\boldsymbol{s}_i^{(t+1)} = \textsc{Upd}\left(\boldsymbol{s}_i^{(t)}\ ,\ \boldsymbol{m}_i^{(t)}\right)$$



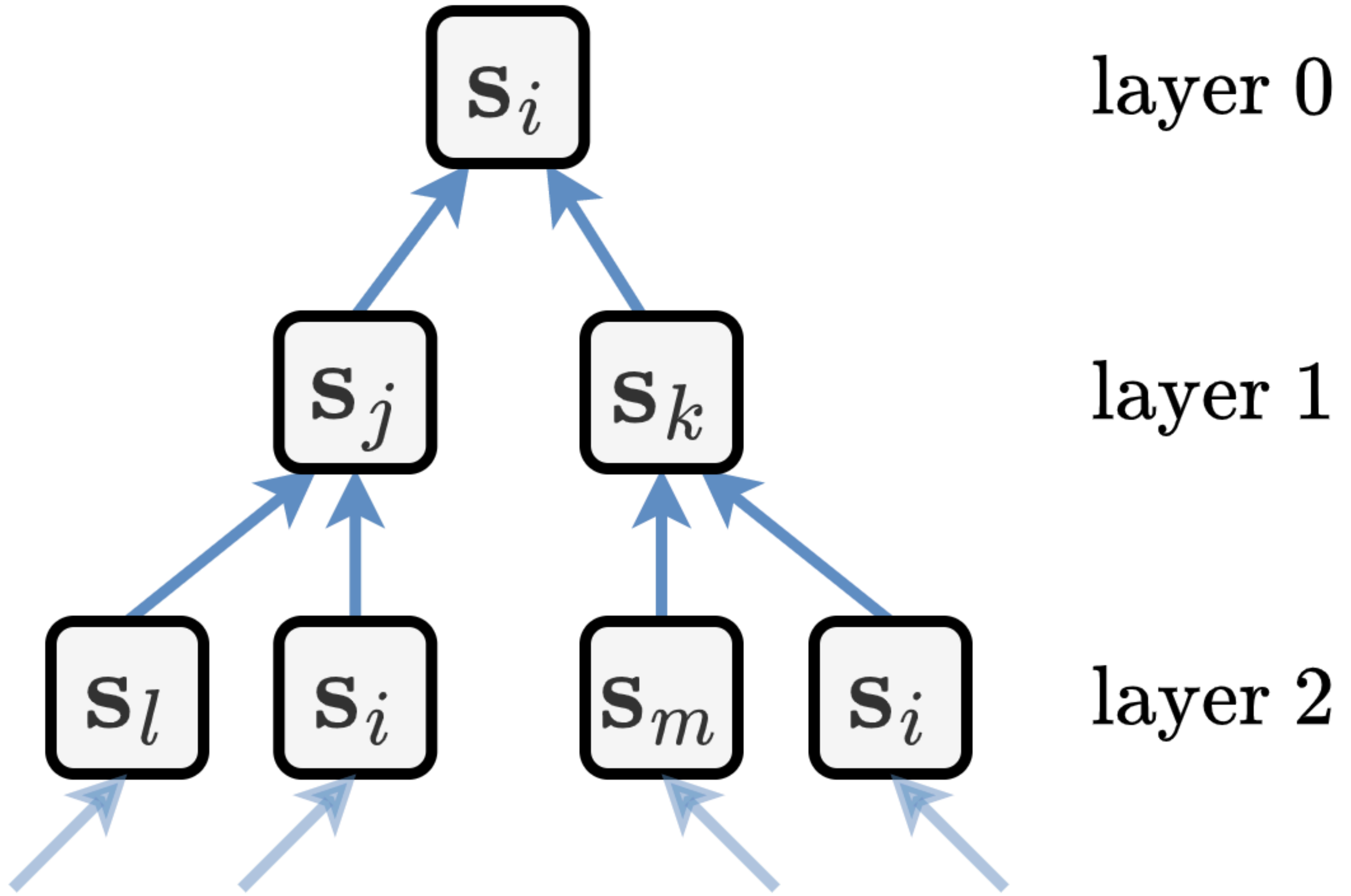
Normal Graph Neural Networks

Message passing updates node features using local aggregation



$$\boldsymbol{m}_i^{(t)} := \text{AGG}\left(\{\!\{(\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}) \mid j \in \mathcal{N}_i\}\!\}\right),$$

$$\boldsymbol{s}_i^{(t+1)} := \text{UPD}\left(\boldsymbol{s}_i^{(t)}, \boldsymbol{m}_i^{(t)}\right),$$



Computation tree:
Message passing gathers & propagates features beyond local neighbourhoods.

Chaitanya Joshi

Fun Fact: Chemistry is crucial for GNNs

Many GNN advances came from computational chemistry

In fact, it can be argued that computational chemists invented the first general-purpose GNNs!

- ChemNet (Kireev *et al.*, CICS'95)
- Baskin *et al.* (CICS'97)
- Molecular Graph Networks (Merkwirth and Lengauer, CIM'05)

This drive continued well into the 2010s:

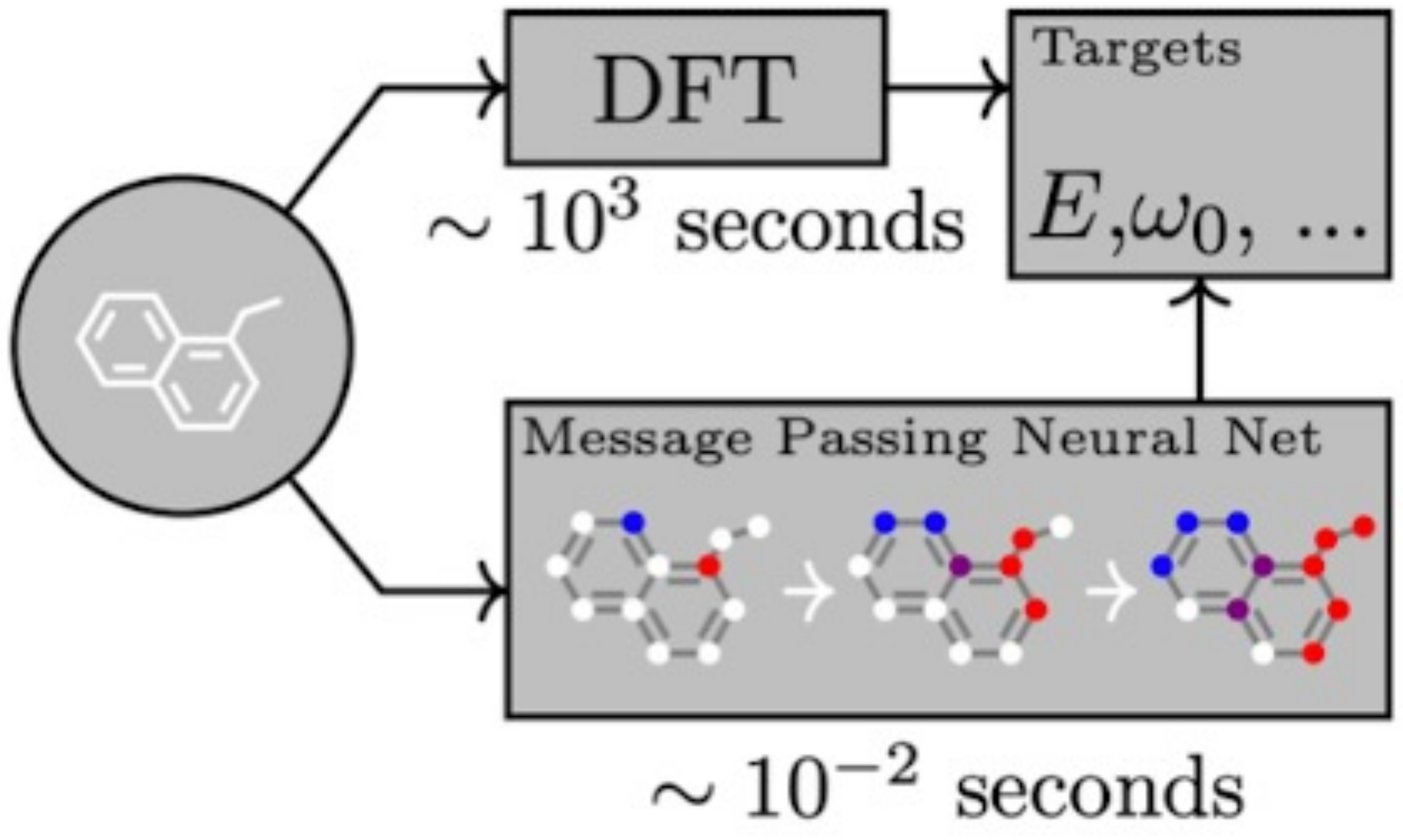
- Molecular fingerprinting GNNs (Duvenaud *et al.*, NeurIPS'15)
- GNNs for quantum chemistry (Gilmer *et al.*, ICML'17)

The classic landscape

One architecture per community

In this work, Gilmer *et al.* tackle head-on the task of quantum property predictions from small-molecule datasets (such as QM9)

Their **target**: replace expensive DFT simulations with learnt GNN models

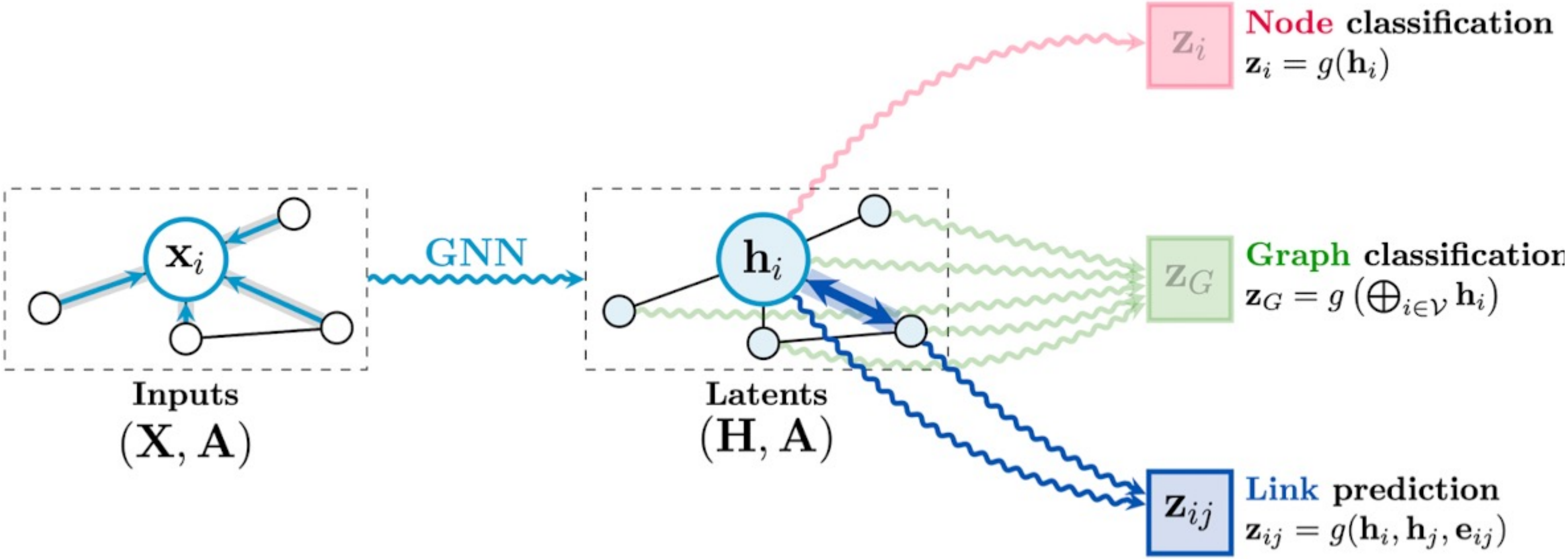


Contribution is also *theoretical*: Categorise **all** existing GNNs at the time into the *MPNN* framework

This framework was generic enough to reach *chemical accuracy* on 11 out of 13 of the tasks within QM9, after a thorough architecture scan.

What if we want other information?

So far we only considered node features



Extending the message passing framework

Edges and Graphs can have features too

Update **edge** features (using graph + relevant nodes)

$$\mathbf{h}_{uv} = \psi\left(\mathbf{x}_u, \mathbf{x}_v, \mathbf{x}_{uv}, \mathbf{x}_{\mathcal{G}}\right)$$

Update **node** features (using updated relevant edges + graph)

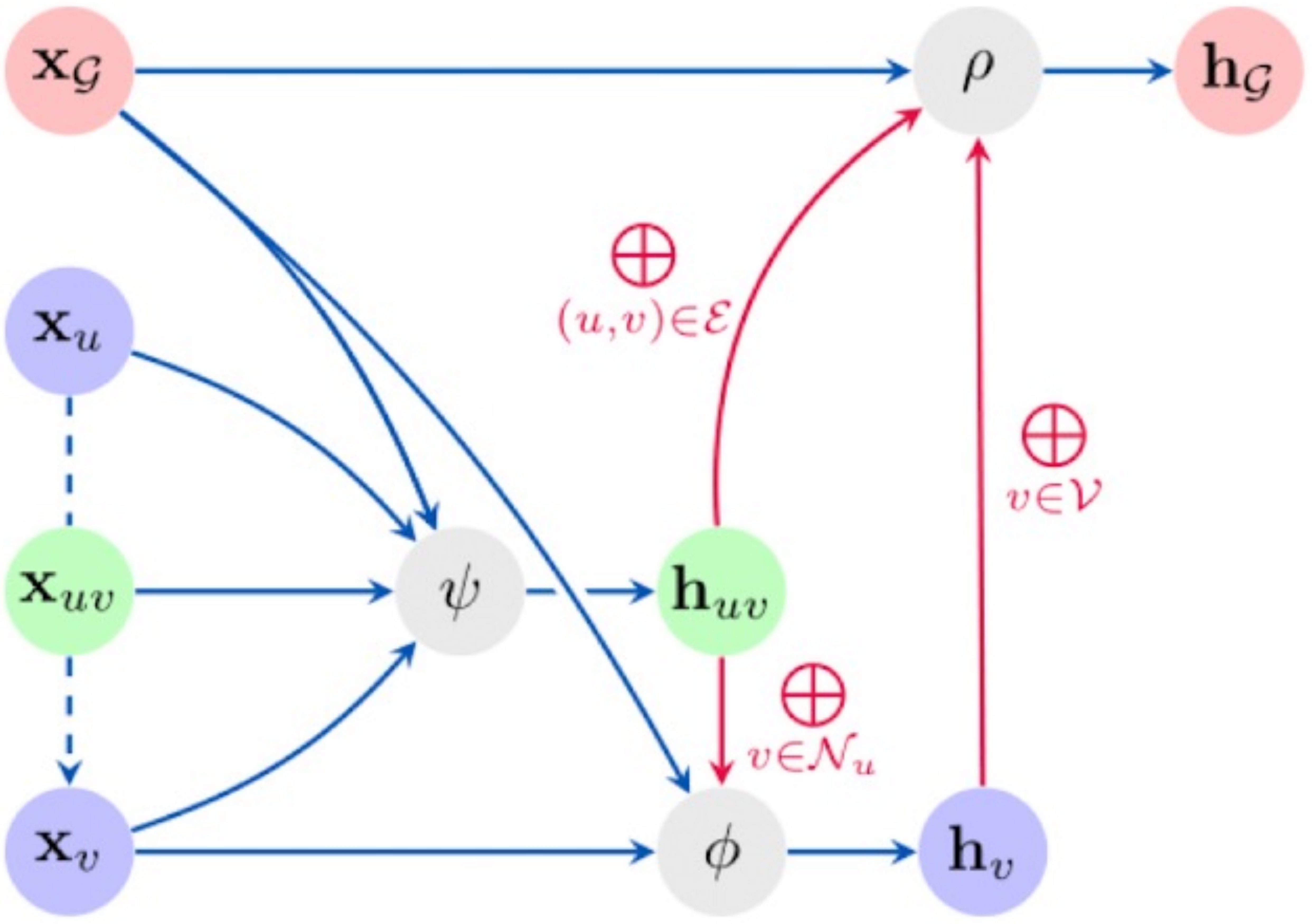
$$\mathbf{h}_u = \phi\left(\mathbf{x}_u, \bigoplus_{u \in \mathcal{N}_v} \mathbf{h}_{vu}, \mathbf{x}_{\mathcal{G}}\right)$$

Update **graph** features (using updated nodes + edges)

$$\mathbf{h}_{\mathcal{G}} = \rho\left(\bigoplus_{u \in \mathcal{V}} \mathbf{h}_u, \bigoplus_{(u,v) \in \mathcal{E}} \mathbf{h}_{uv}, \mathbf{x}_{\mathcal{G}}\right)$$

Make it as complex as you like!

All architectures discussed are special cases

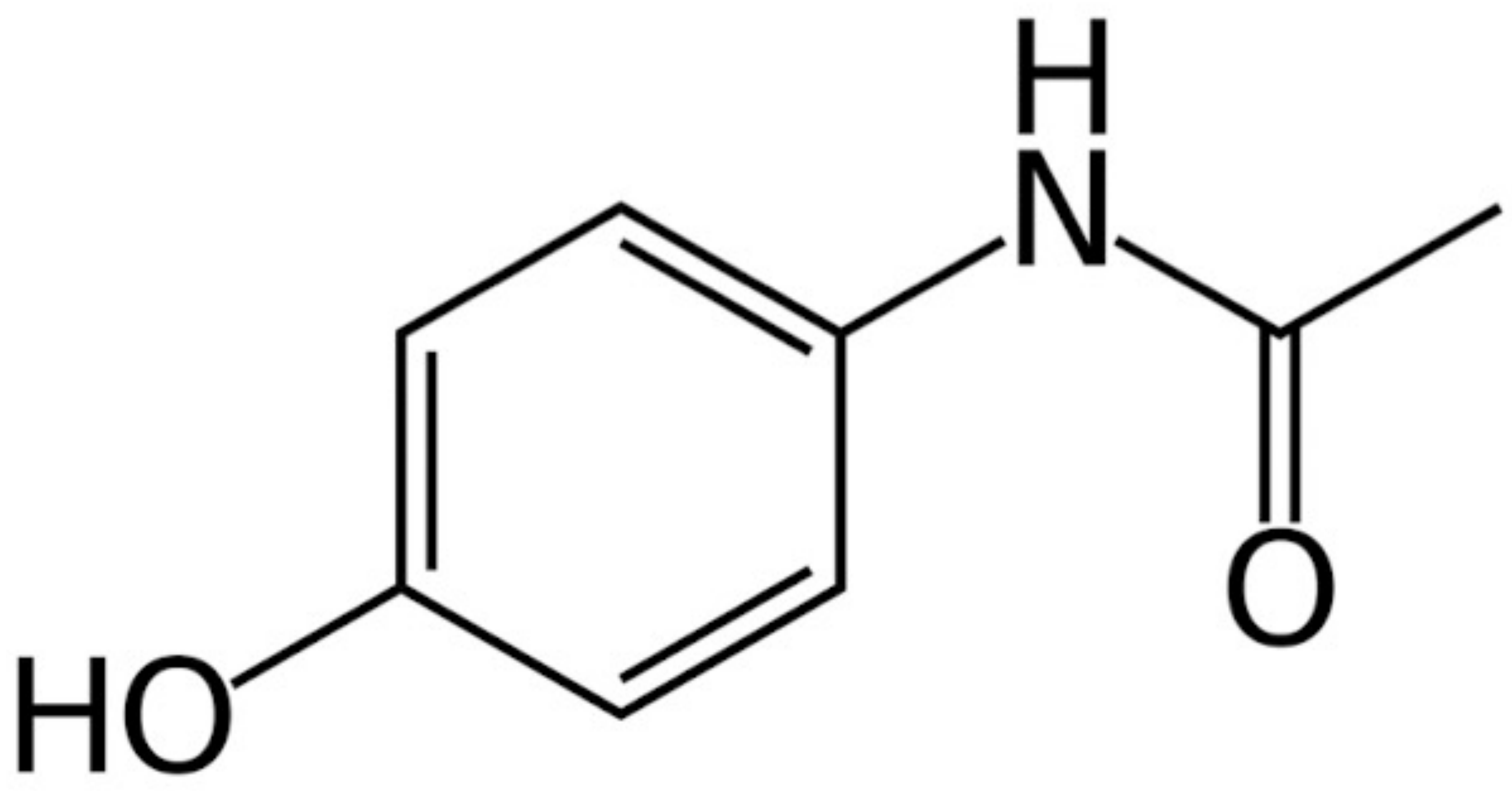


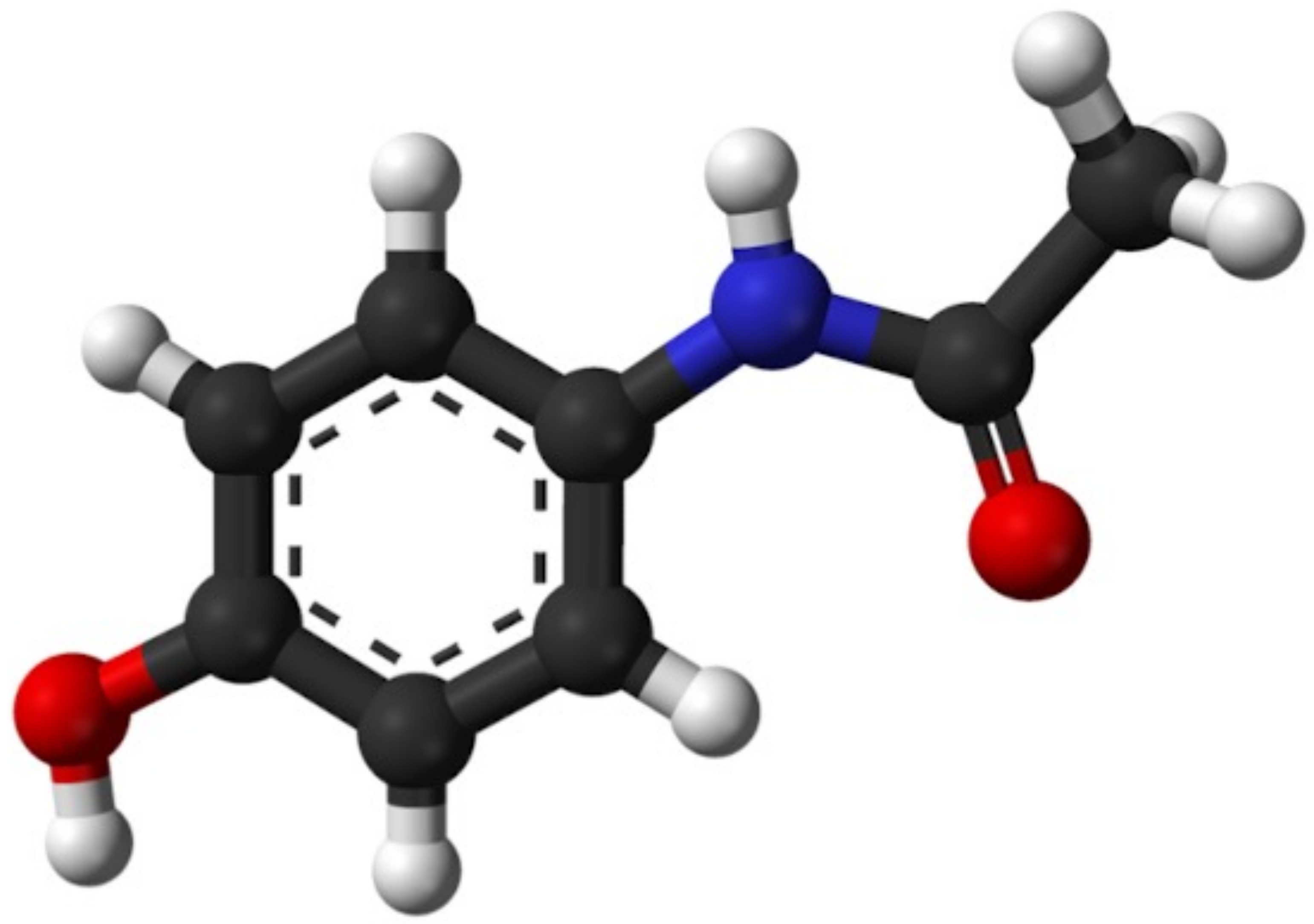
3. Geometry and Symmetry

Outline of the road ahead

Incorporate relational and then geometric information

$C_8H_9NO_2$





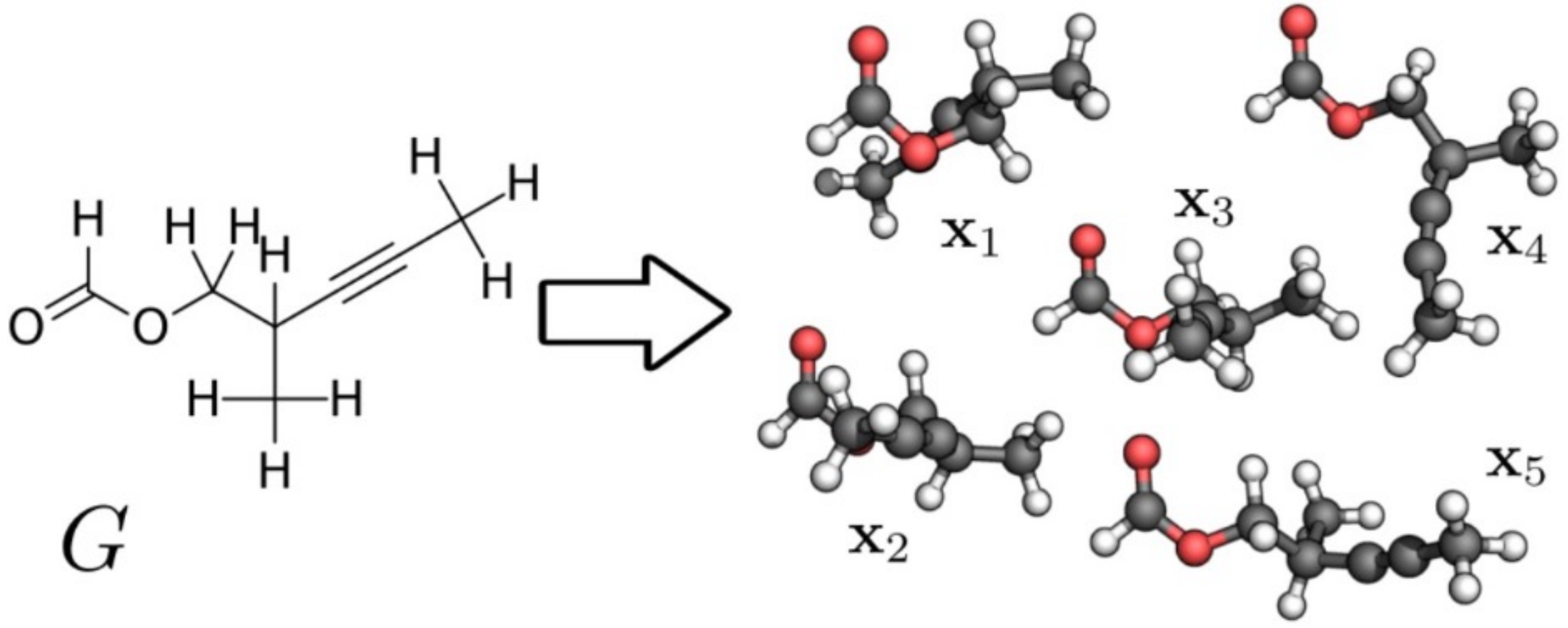
Deep Sets

GNNs

Geometric GNNs

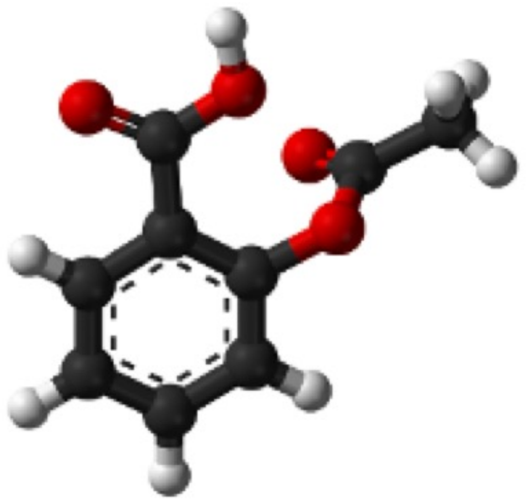
Applying our framework to molecules

Is there more to a structure than the 2D representation?

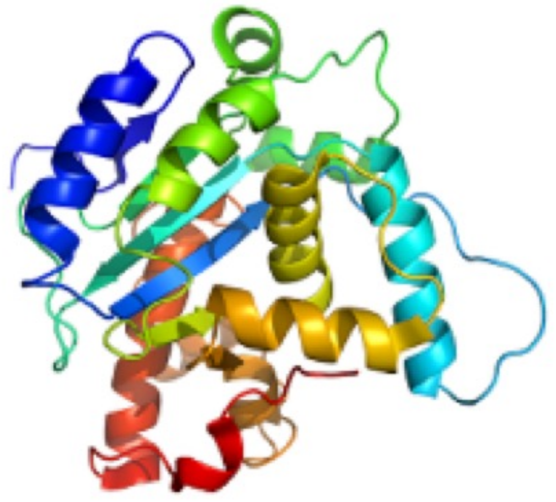


Simm, Gregor NC, and José Miguel Hernández-Lobato. "A generative model for molecular distance geometry." *ICML 2020*

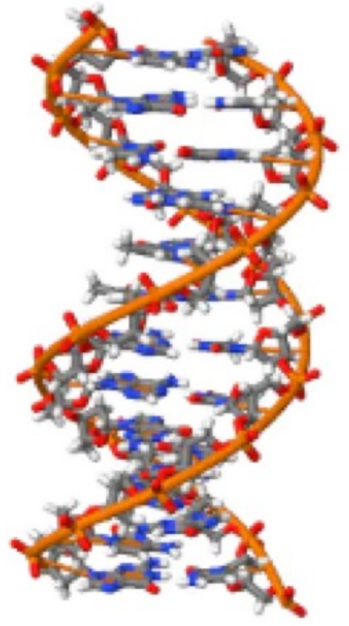
Systems with geometric & relational structure



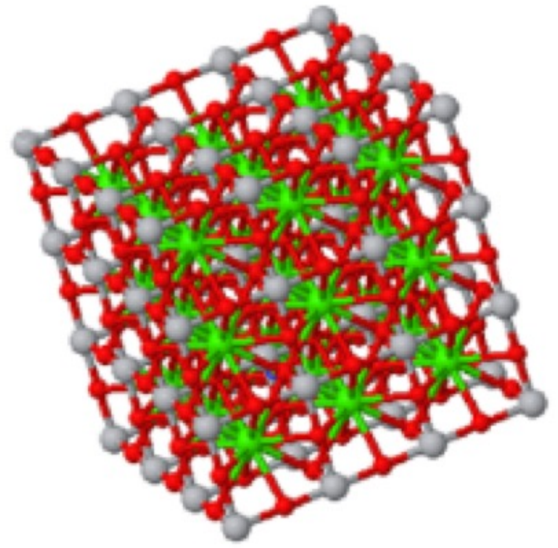
Small Molecules



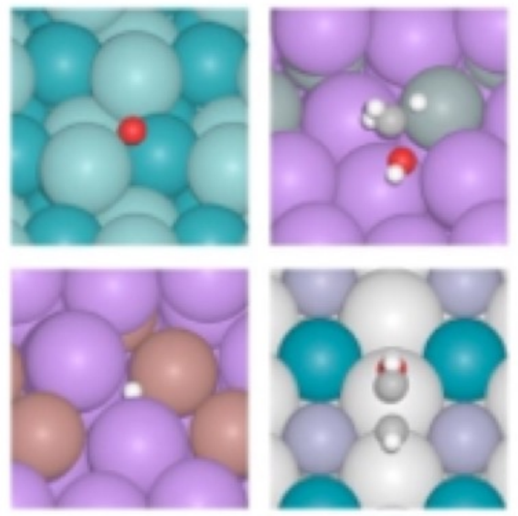
Proteins



DNA/RNA



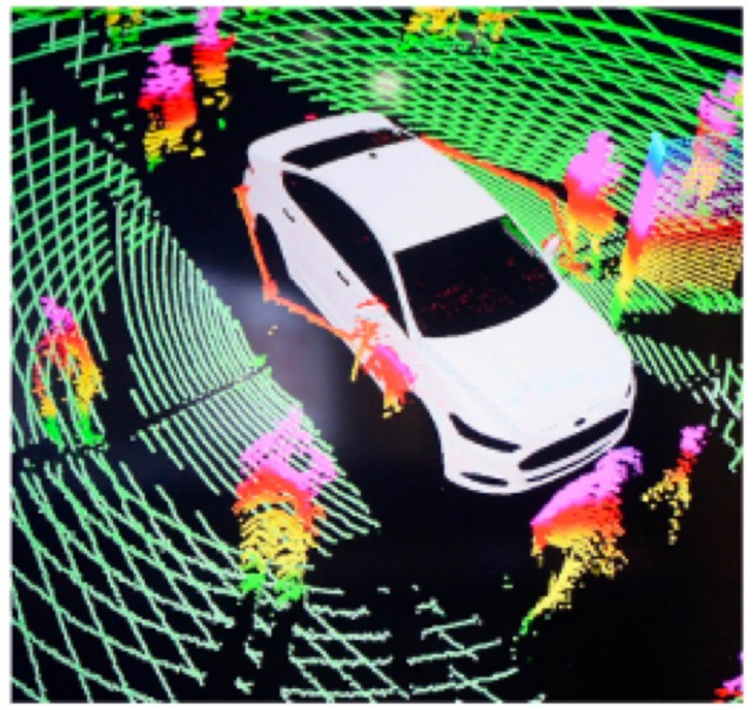
Inorganic Crystals



Catalysis Systems

Transportation & Logistics

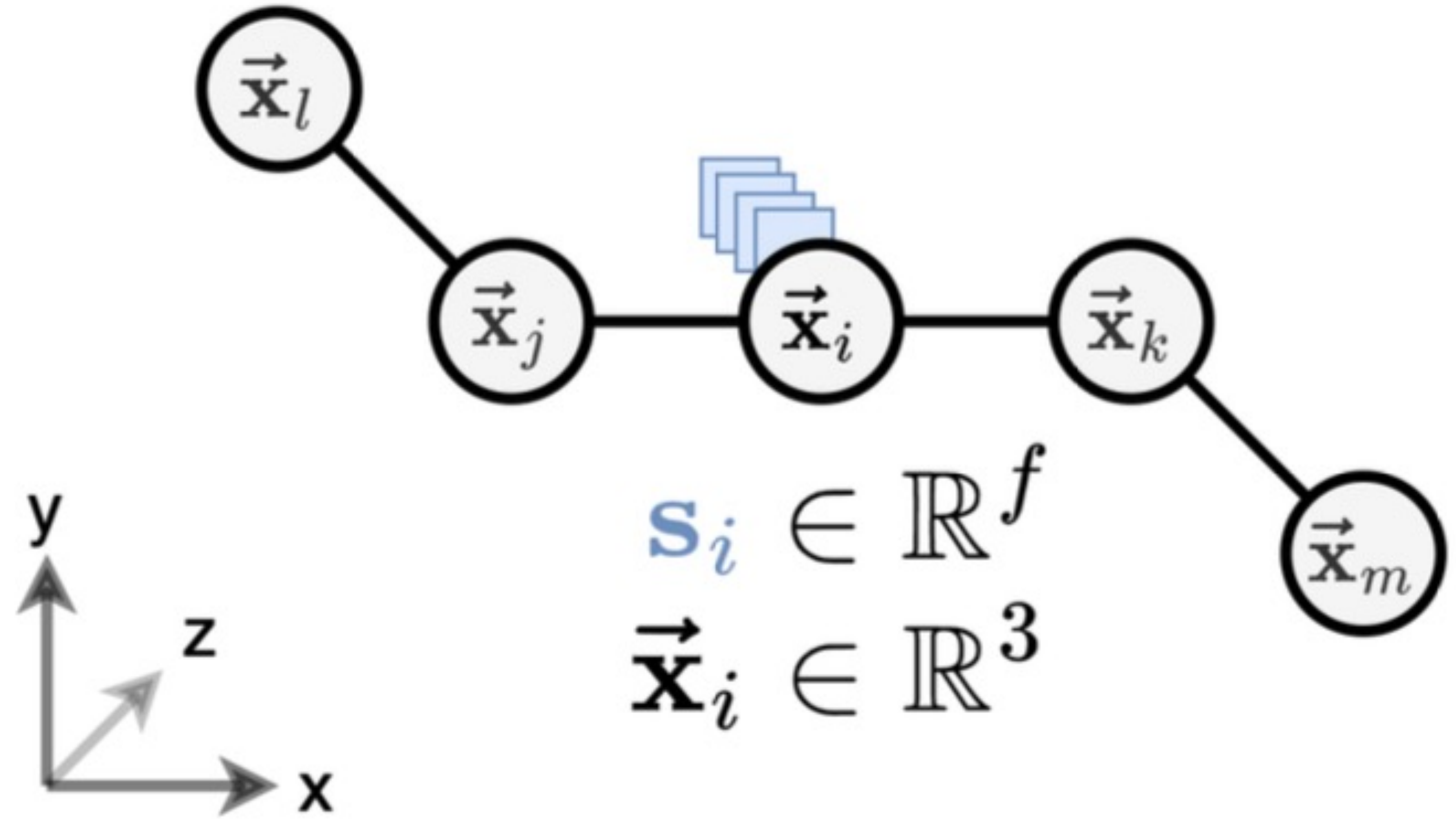
Robotic Navigation



3D Computer Vision

Geometric Graphs

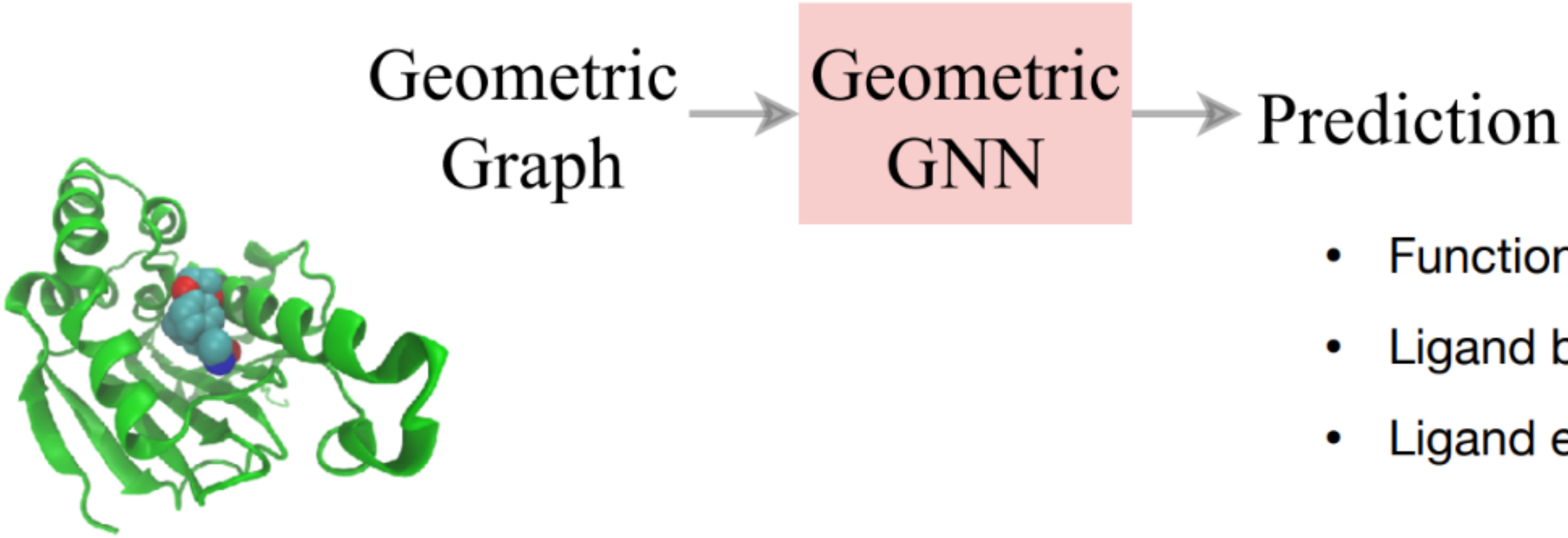
A graph G=(A,S,X) embedded in Euclidean space



- A: an $n \times n$ adjacency matrix.
- S ∈ $R^{n \times f}$: scalar features.
- X ∈ $R^{n \times d}$: tensor features, e.g., coordinates.

Why geometric GNNs?

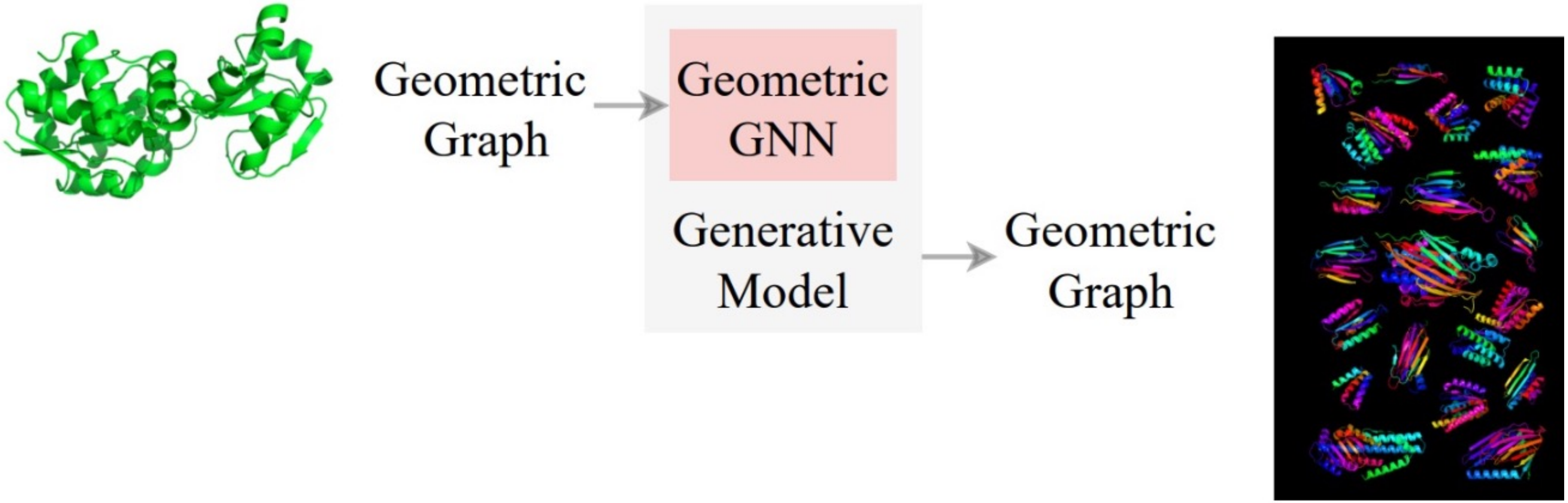
Supervised Learning: Predict functional properties



- Functional properties?
- Ligand binding affinity?
- Ligand efficacy?

Why geometric GNNs?

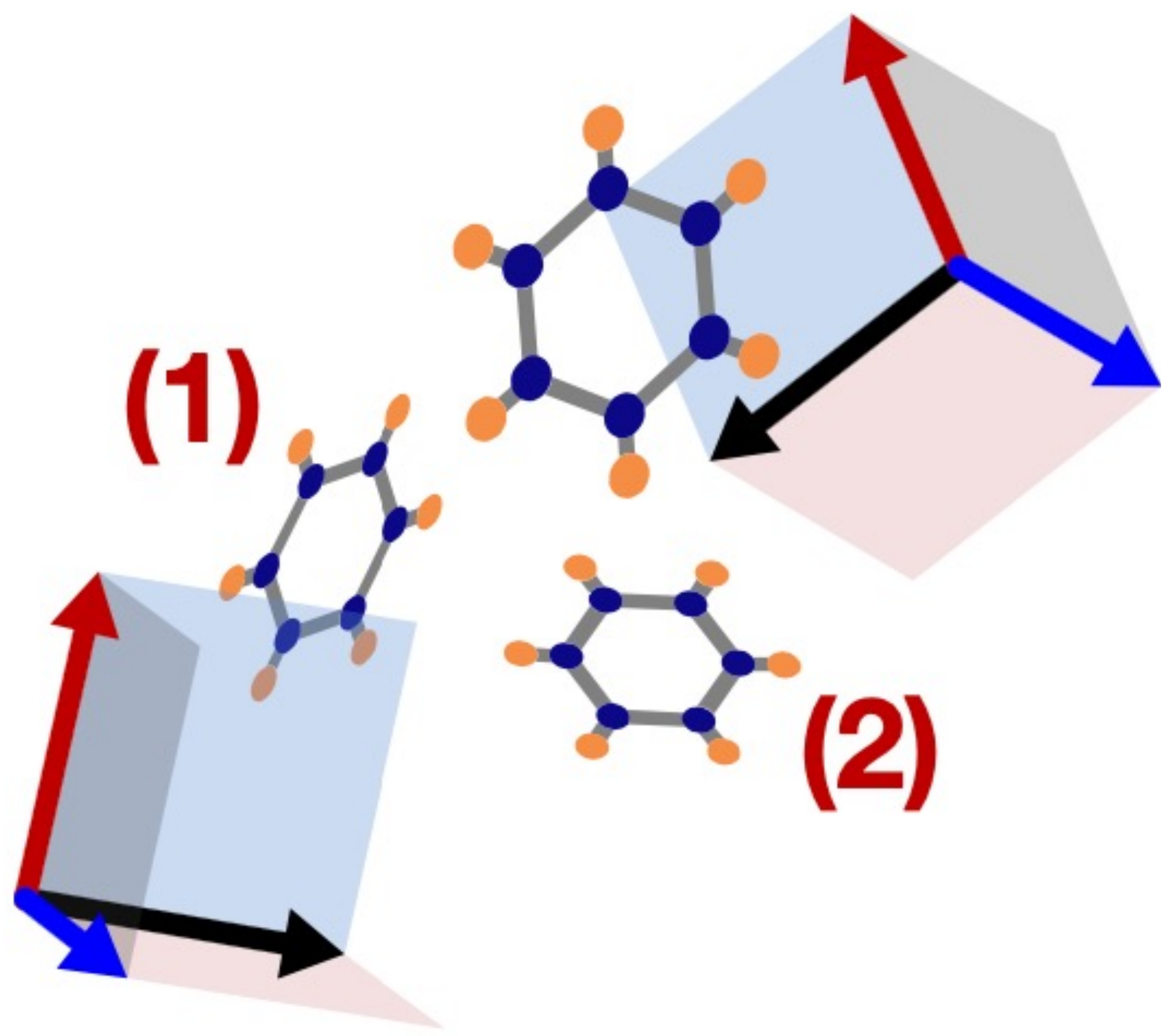
Generative Modelling (L6): Design new molecules



How to deal with geometric graphs?

The problem of symmetry in input and output

- To describe geometric graphs we use coordinate systems
  - (1) and (2) use different coordinate systems to describe the **same** molecular geometry.
- We can describe the transform between coordinate systems with symmetries of Euclidean space
  - 3D rotations, translations

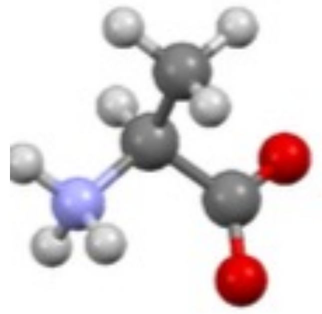


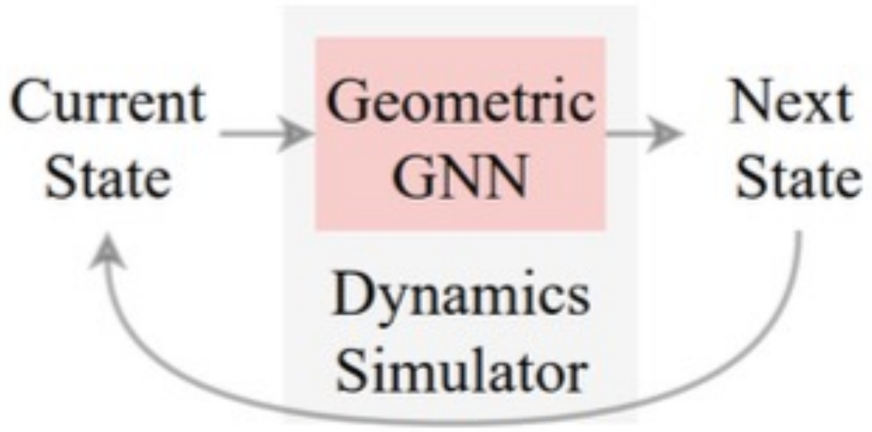
- **However, output of traditional GNNs given (1) and (2) as completely different!**

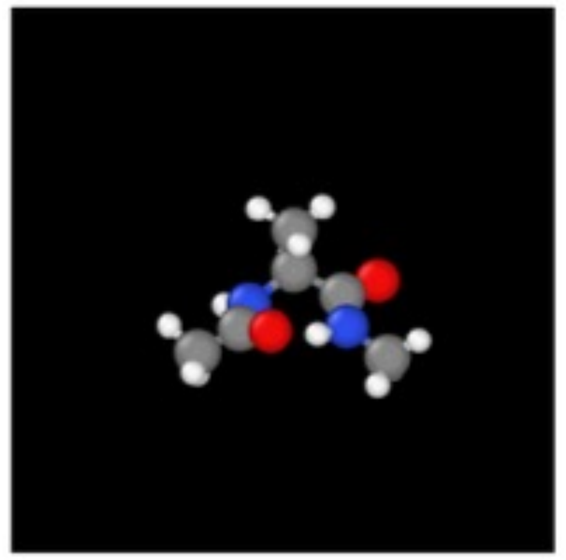
How to deal with geometric graphs?

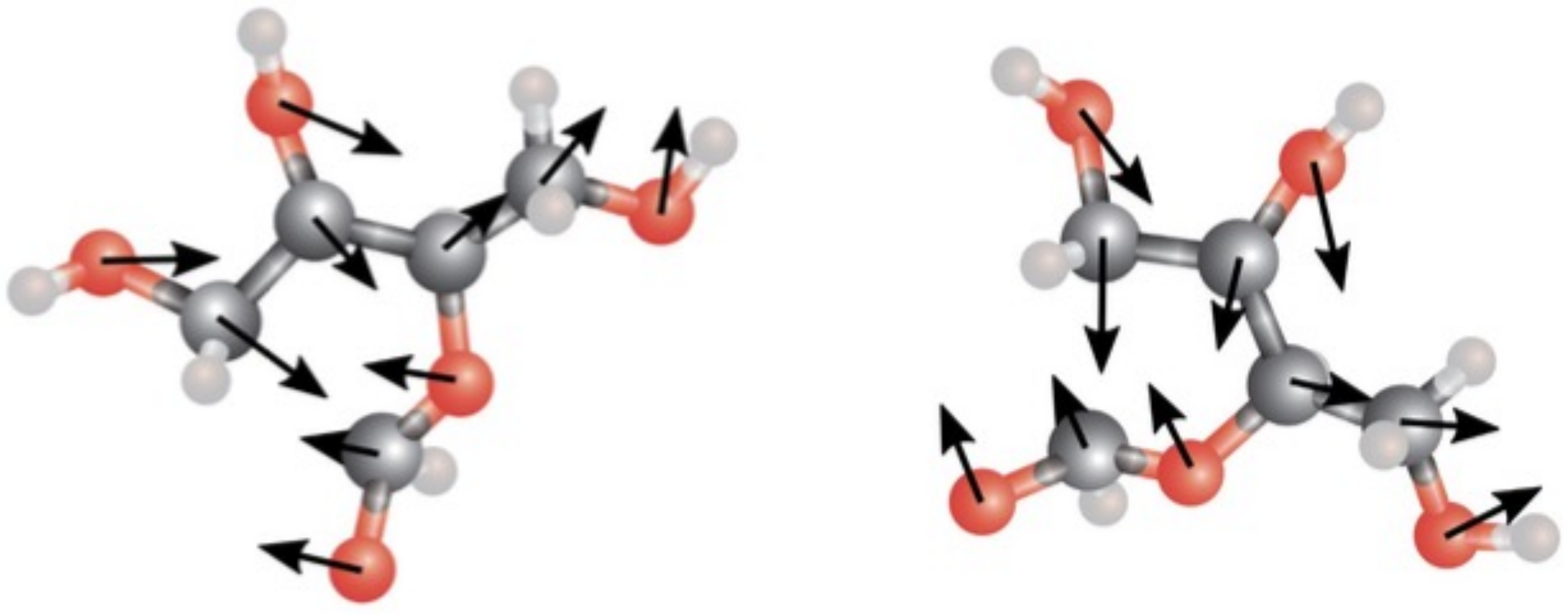
The problem of symmetry in input and output

- Beyond input space, output can also be tensors
- Example: simulation (force prediction)
  - Given a molecule and a rotated copy, predicted forces should be the same up to rotation
  - (i.e., Predicted forces are **equivariant** to rotation)



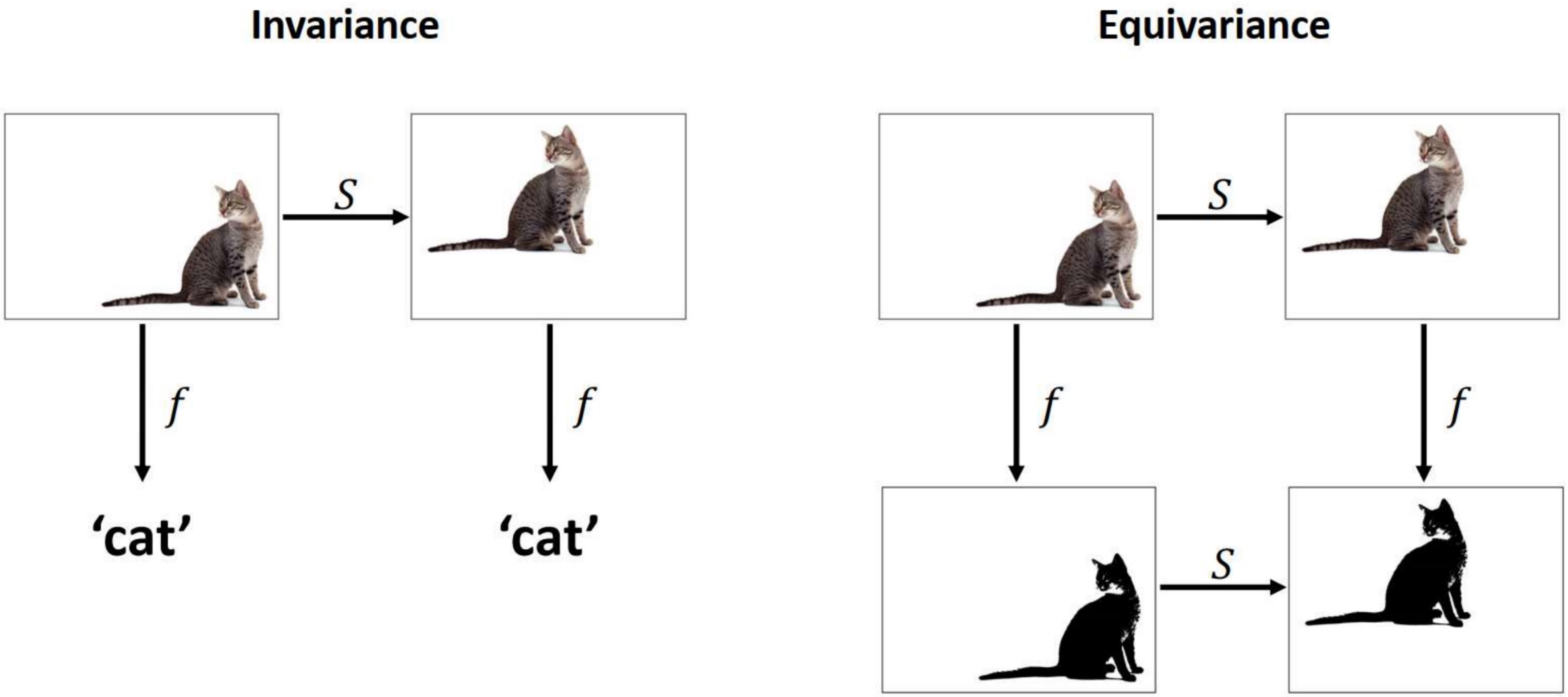






Inductive Bias: Translational In-/Equivariance

Leverage the symmetry of your data



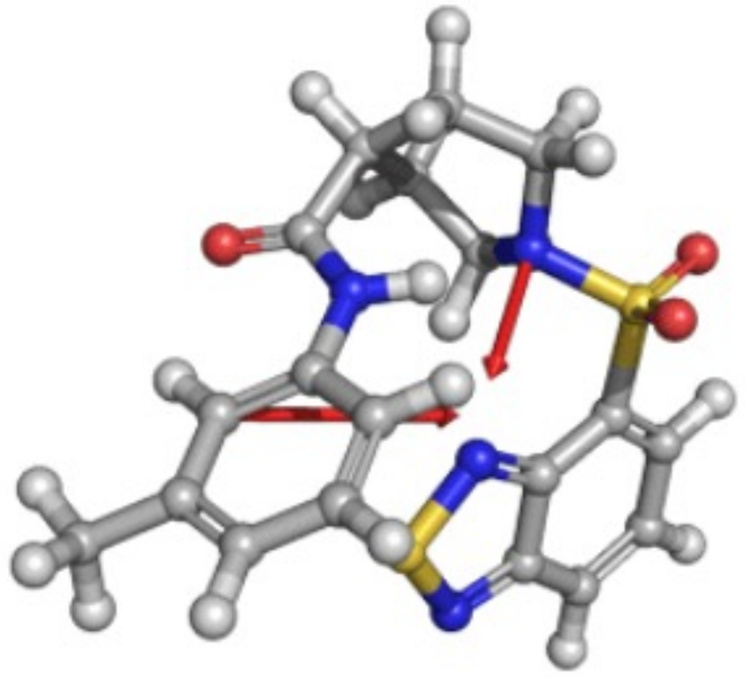
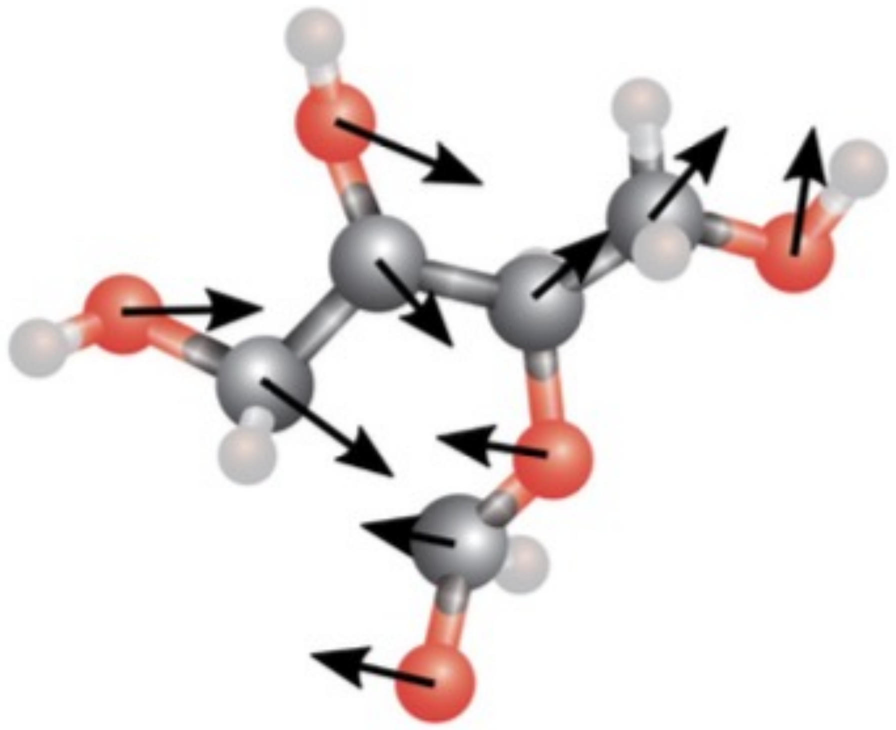
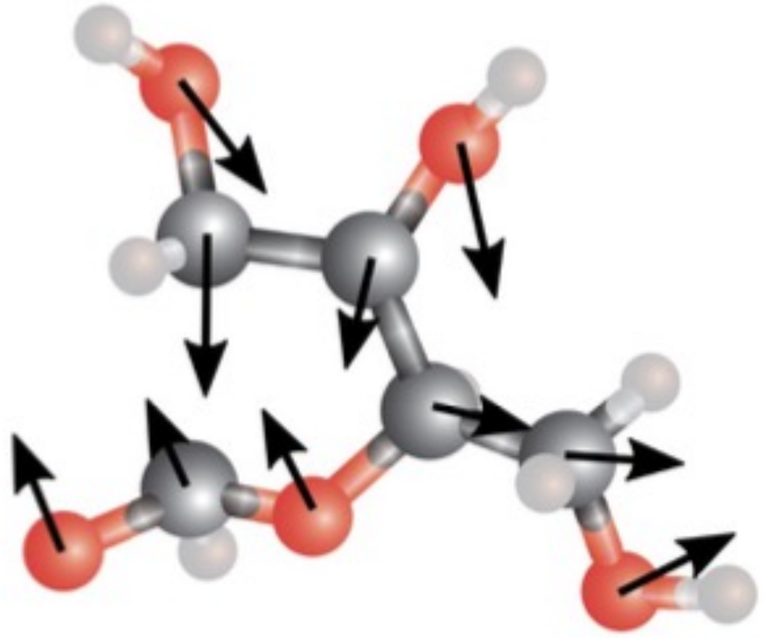
Generalising Invariance and Equivariance

Equivariance: Things change as they should

- Formal definition of Equivariance:
  a function $F: X \rightarrow Y$ is equivariant if for a transformation ρ it satisfies:

$$F \circ \rho_X(x) = \rho_Y \circ F(x)$$

  - Example: $\rho_X,\ \rho_Y$ are same rotation transformation

The classic landscape

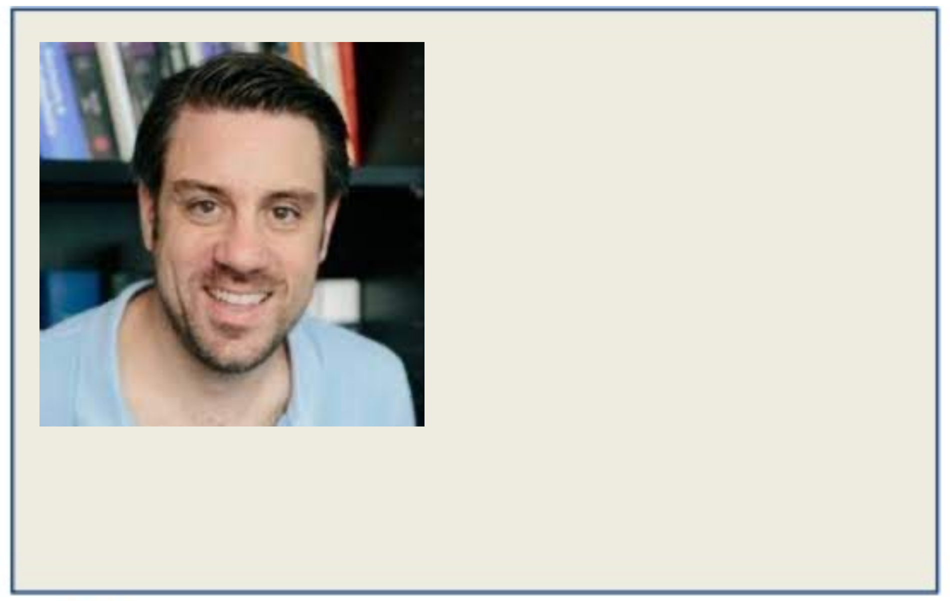
Invariance: Things do not change at all!

- Definition of Invariance: a function $F: X \rightarrow Y$ is invariant if for a transformation ρ it satisfies:

$$F \circ \rho_X(x) = F(x)$$

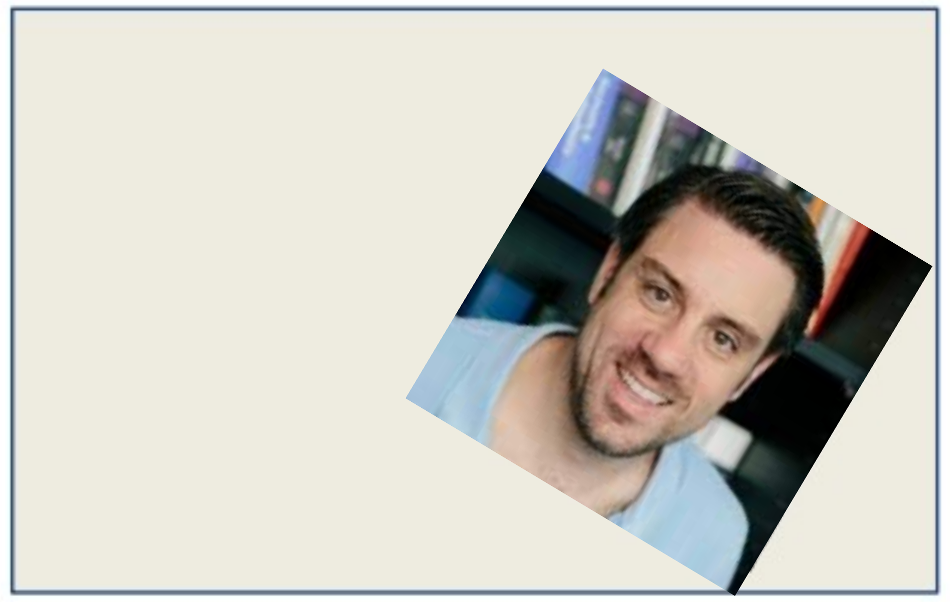
- **Note:** invariance is a special case of equivariance where ρ_Y is defined as no transformation.

$$F \circ \rho_X(x) = \rho_Y \circ F(x)$$



✓ Yes, Dr. Beck

After roto-translation…

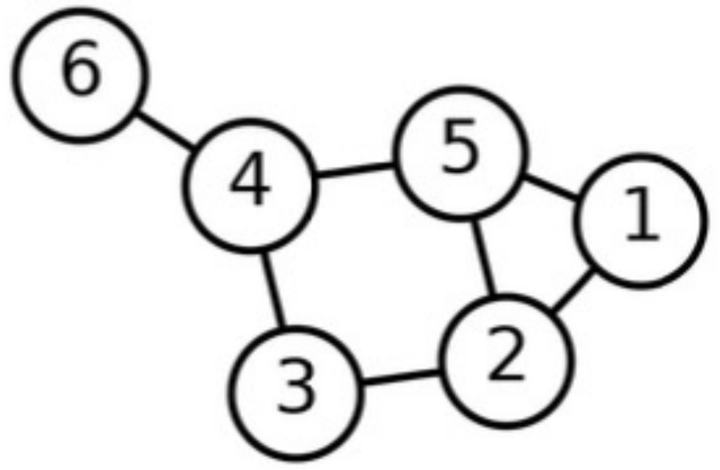
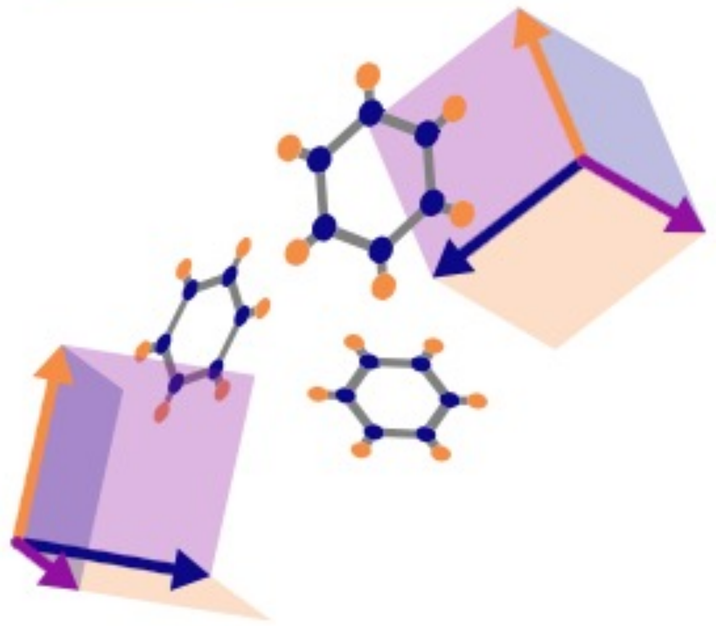


✓ Still Dr. Beck!

Inductive Biases = Respecting Symmetry

Choose your architecture based on your data type

Neural networks are specially designed for different data types in order to make use of special features (symmetries) of the data.

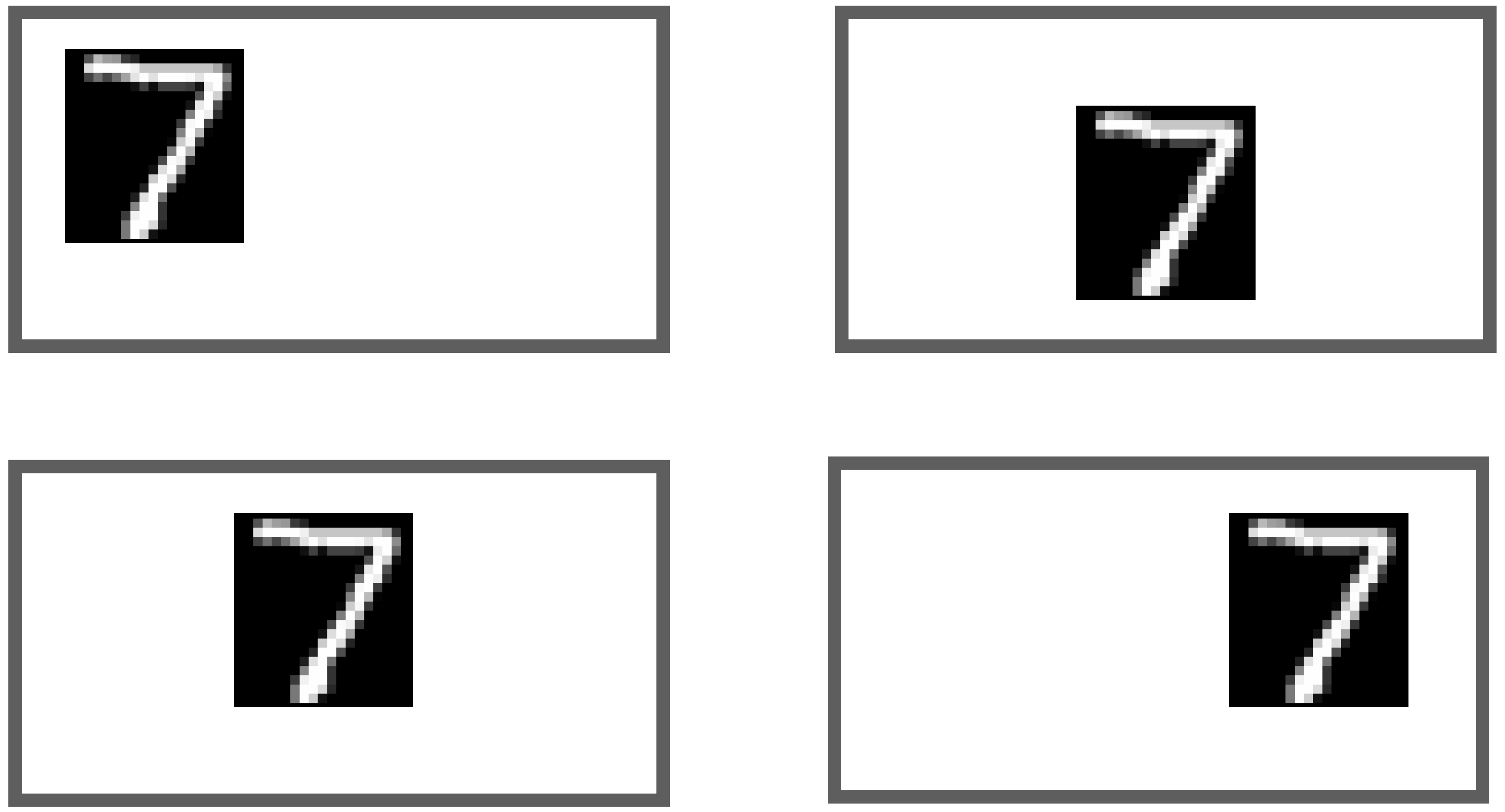
Data type	Images	Text	Graph	Geometric Graph in 3D
		Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi ultricies, justo ac viverra euismod, justo odio eleifend dolor, a imperdiet quam nibh finibus mauris. Morbi lobortis a lorem id dapibus. Interdum et malesuada fames...	 	
Type of neural network	**Convolutional** Pixels closer together are more important to each other.	**Recurrent** The meaning of a current word depends on what came before.	**Graph** Data on nodes interacts via edges	**Euclidean** Geometric data "means" the same thing even when we use different coordinate systems
	Spatial translation symmetry	***Time translation symmetry***	***Permutation symmetry***	***Euclidean symmetry***

4. Geometric GNNs

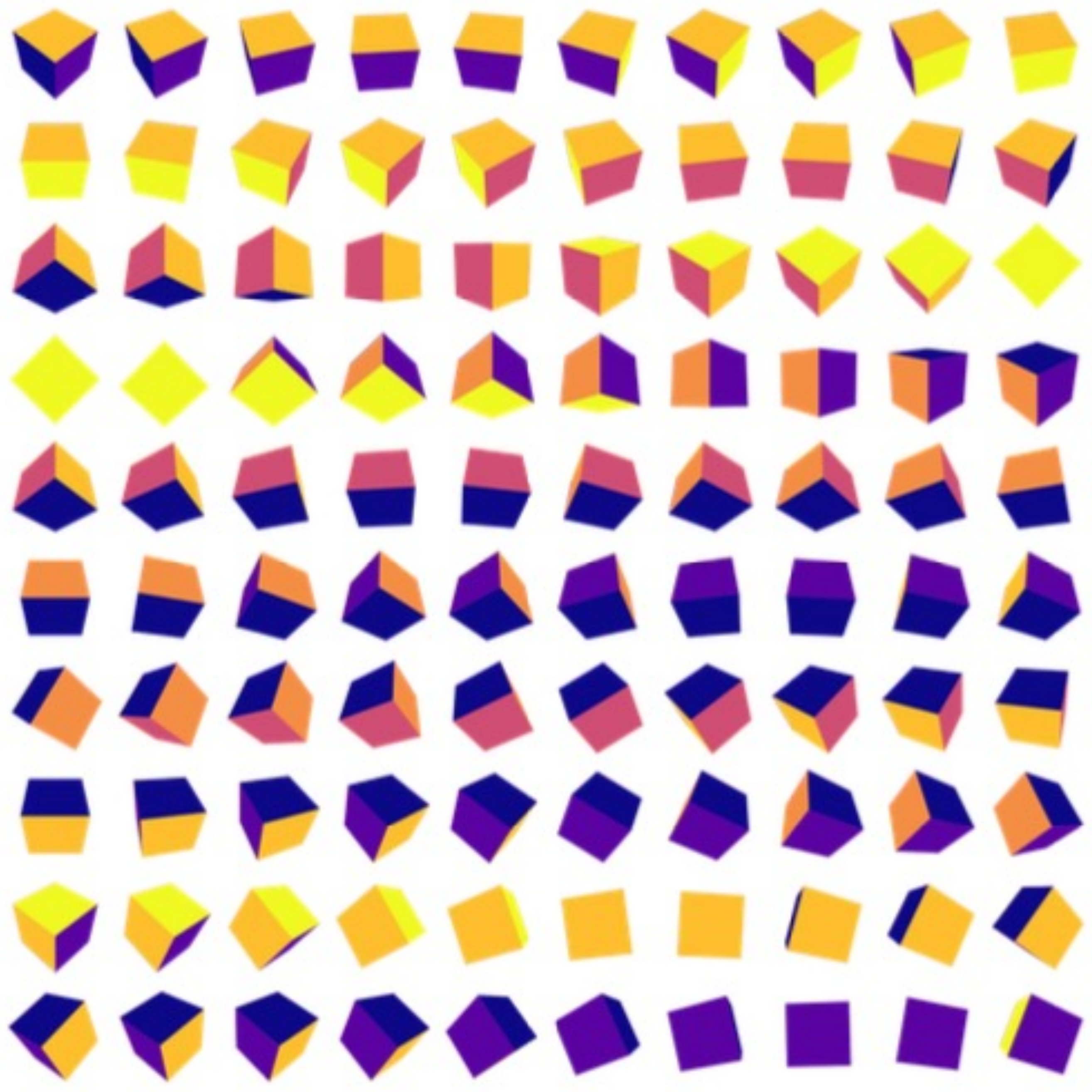
Why would we want to respect symmetry?

It makes our learning a lot more efficient!

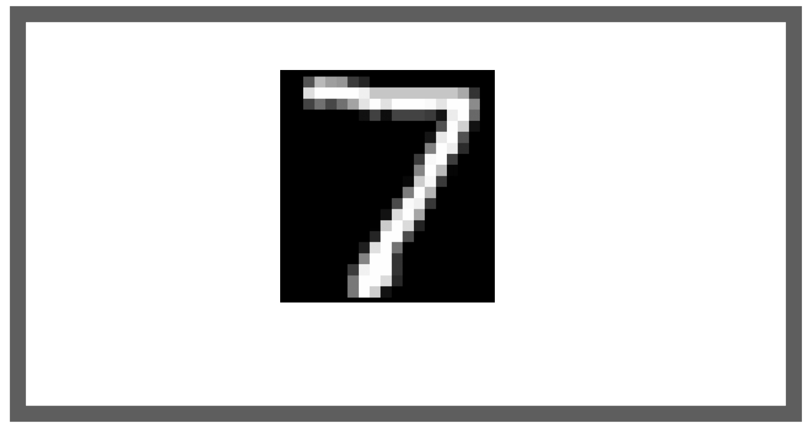
Training without translational symmetry



training without rotational symmetry



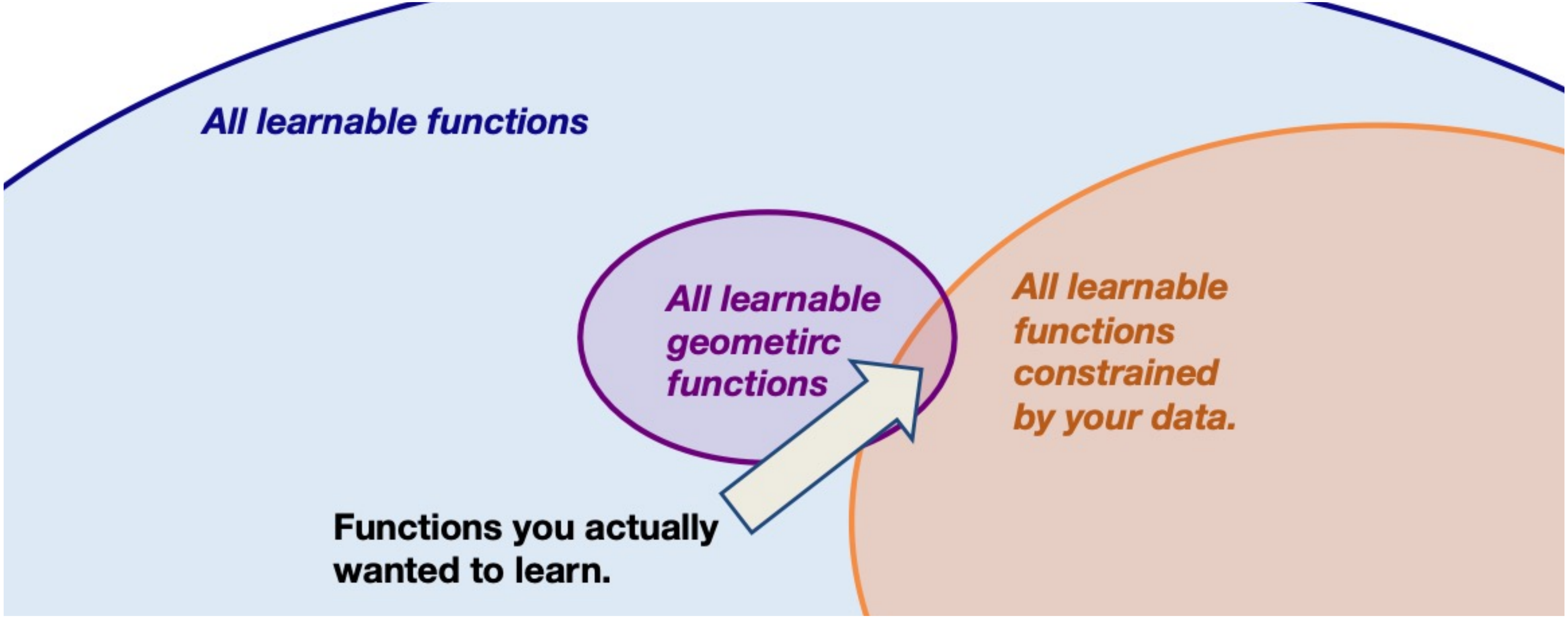
Training with translational symmetry



training with symmetry

Why would we want to respect symmetry?

Less possible functions our network has to consider!

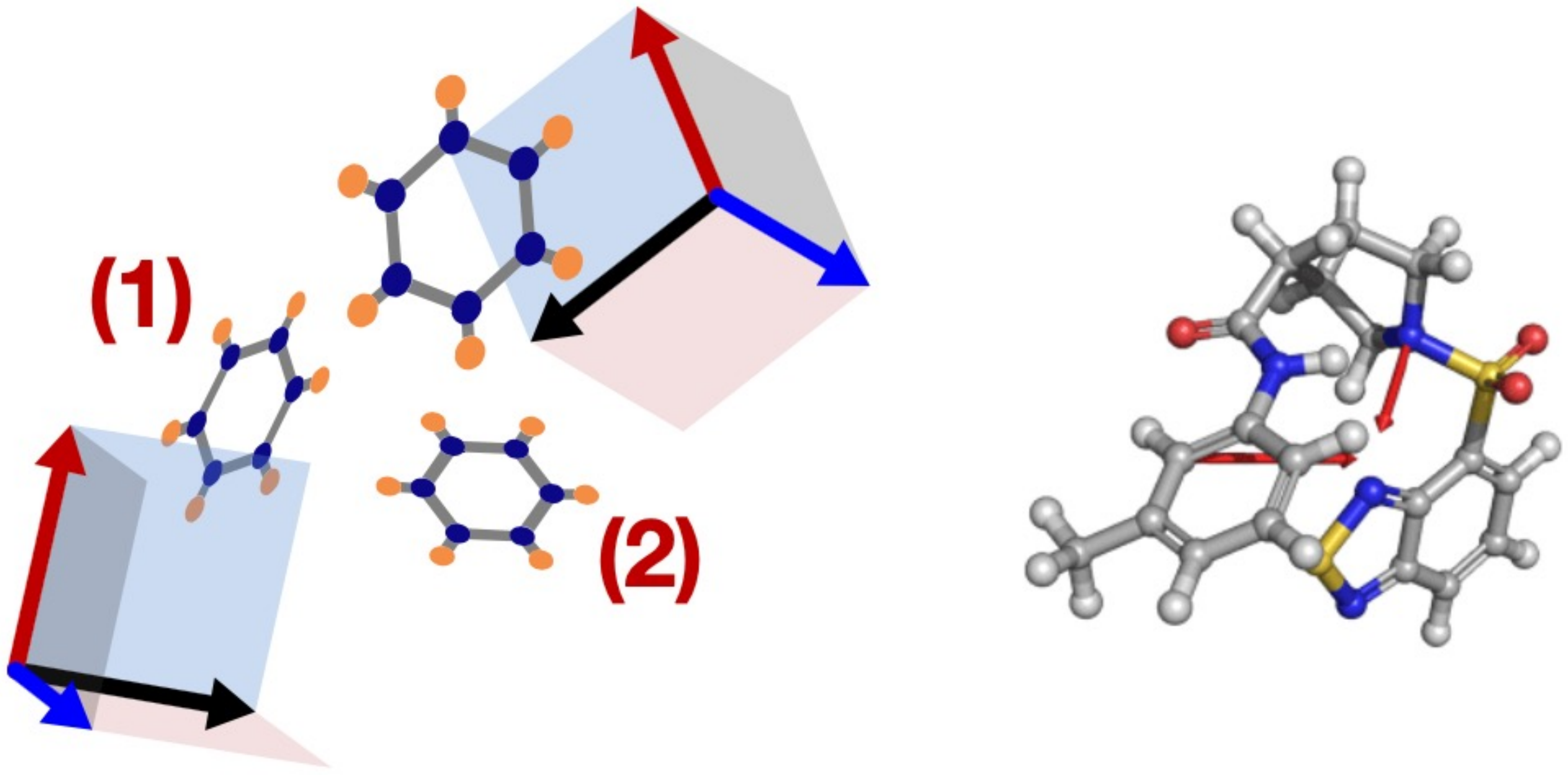


How to construct geometric GNNs

Invariance vs equivariance

Two classes of Geometric GNNs:

- Invariant GNNs for learning invariant scalar features
- Equivariant GNNs for learning equivariant tensor features.

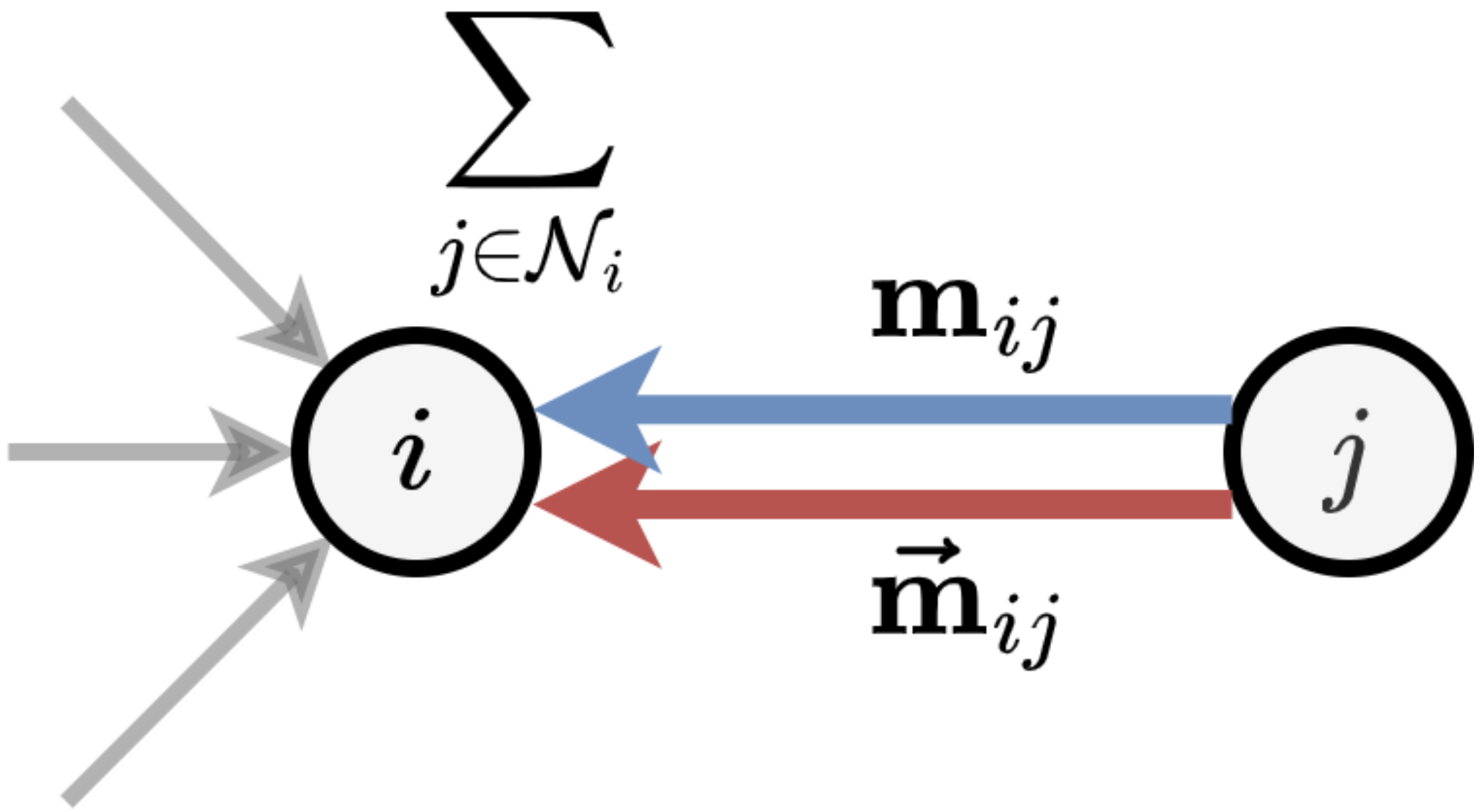


Invariant functions vs. Equivariant functions

Geometric GNN message passing

Geometric GNNs:

- **update** **scalar** **and** (optionally) **vector features**
- **aggregate and update functions which retain transformation semantics**



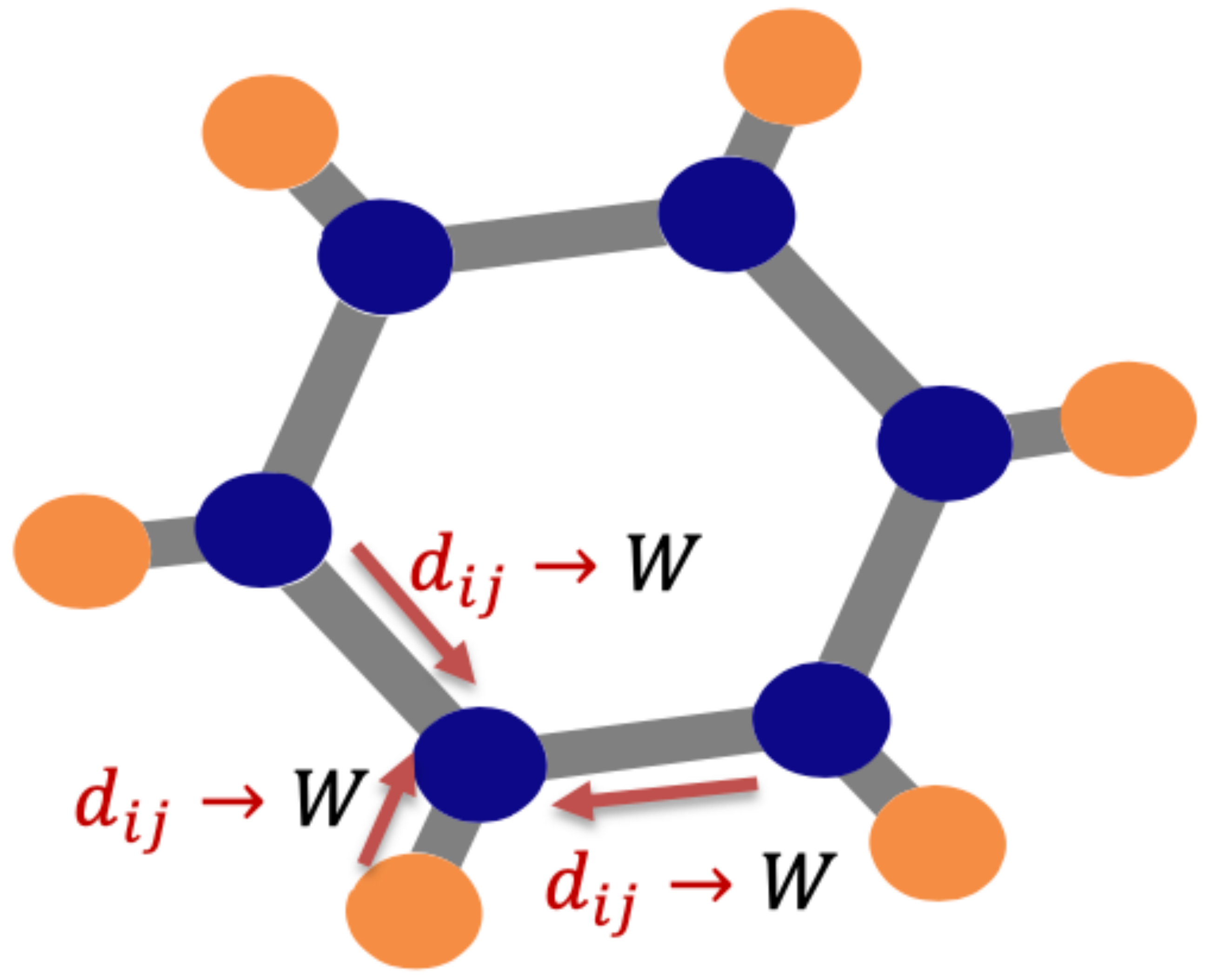
$$\boldsymbol{m}_i^{(t)}, \vec{\boldsymbol{m}}_i^{(t)} := \textsc{Agg}\left(\left\{\!\!\left\{ (\boldsymbol{s}_i^{(t)}, \boldsymbol{s}_j^{(t)}, \vec{\boldsymbol{v}}_i^{(t)}, \vec{\boldsymbol{v}}_j^{(t)}, \vec{\boldsymbol{x}}_{ij}) \mid j \in \mathcal{N}_i \right\}\!\!\right\}\right) \quad \text{(Aggregate)}$$

$$\boldsymbol{s}_i^{(t+1)}, \vec{\boldsymbol{v}}_i^{(t+1)} := \textsc{Upd}\left((\boldsymbol{s}_i^{(t)}, \vec{\boldsymbol{v}}_i^{(t)}) \ , \ (\boldsymbol{m}_i^{(t)}, \vec{\boldsymbol{m}}_i^{(t)}) \right) \quad \text{(Update)}$$

Invariant GNN: SchNet (2017)

Using relative distances as invariant weights

- **SchNet** makes W invariant by **scalarizing** relative positions $\overrightarrow{r_{ij}}$ with relative distances $d_{ij} = \|\vec{r}_{ij}\|$:
  - $\|\vec{r}_{ij}\|$ are invariant to rotations and translations
  - => each message passing layer weight W is invariant
  - => aggregated node embeddings $\sum_j x_j \cdot W$ is invariant
  - => therefore, node embeddings are invariant!



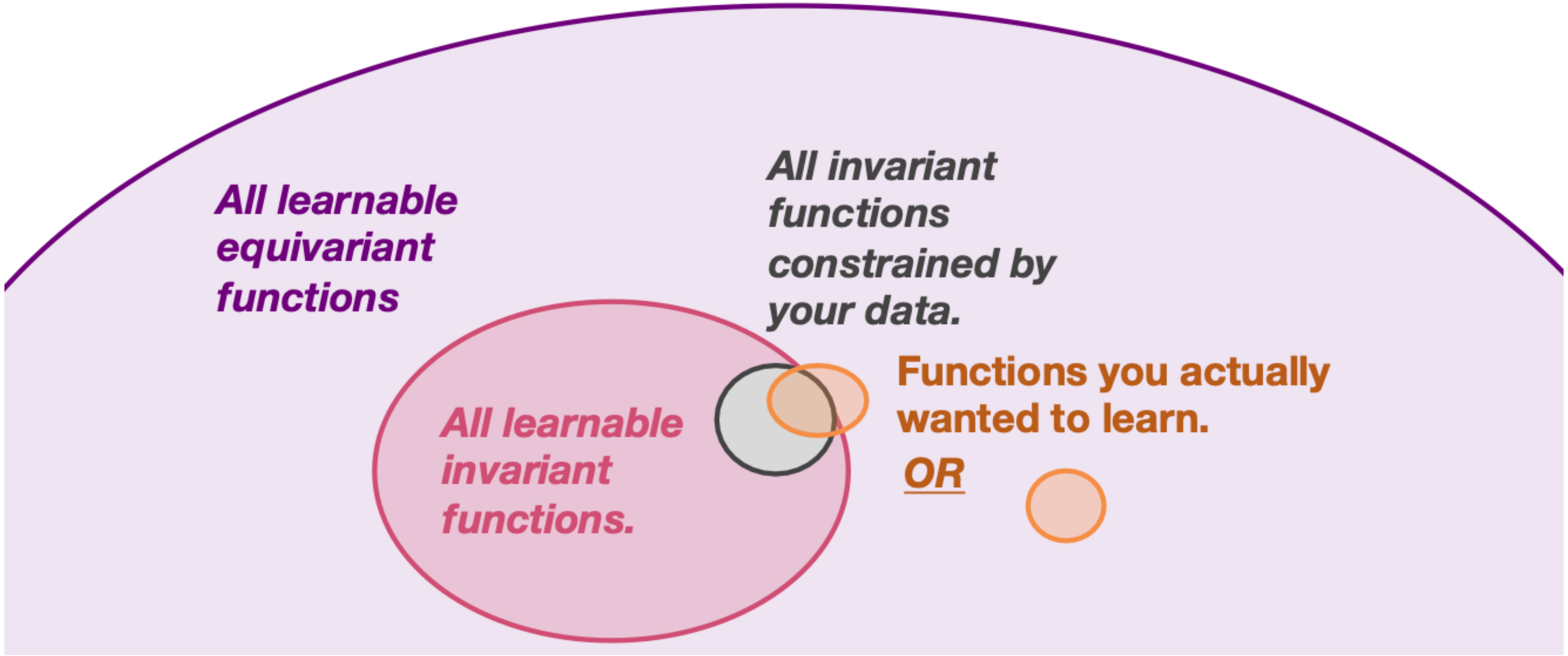
$$\mathbf{x}_i^{l+1} = (X^l * W^l)_i = \sum_j \mathbf{x}_j^l \circ W^l(\mathbf{r}_i - \mathbf{r}_j),$$

x^l: node embeddings at l layer
r: atomic coordinates

Why then equivariant GNNs?

Expanding what interactions our network can extract

- You have to guarantee that your input features already contain any necessary equivariant interactions.

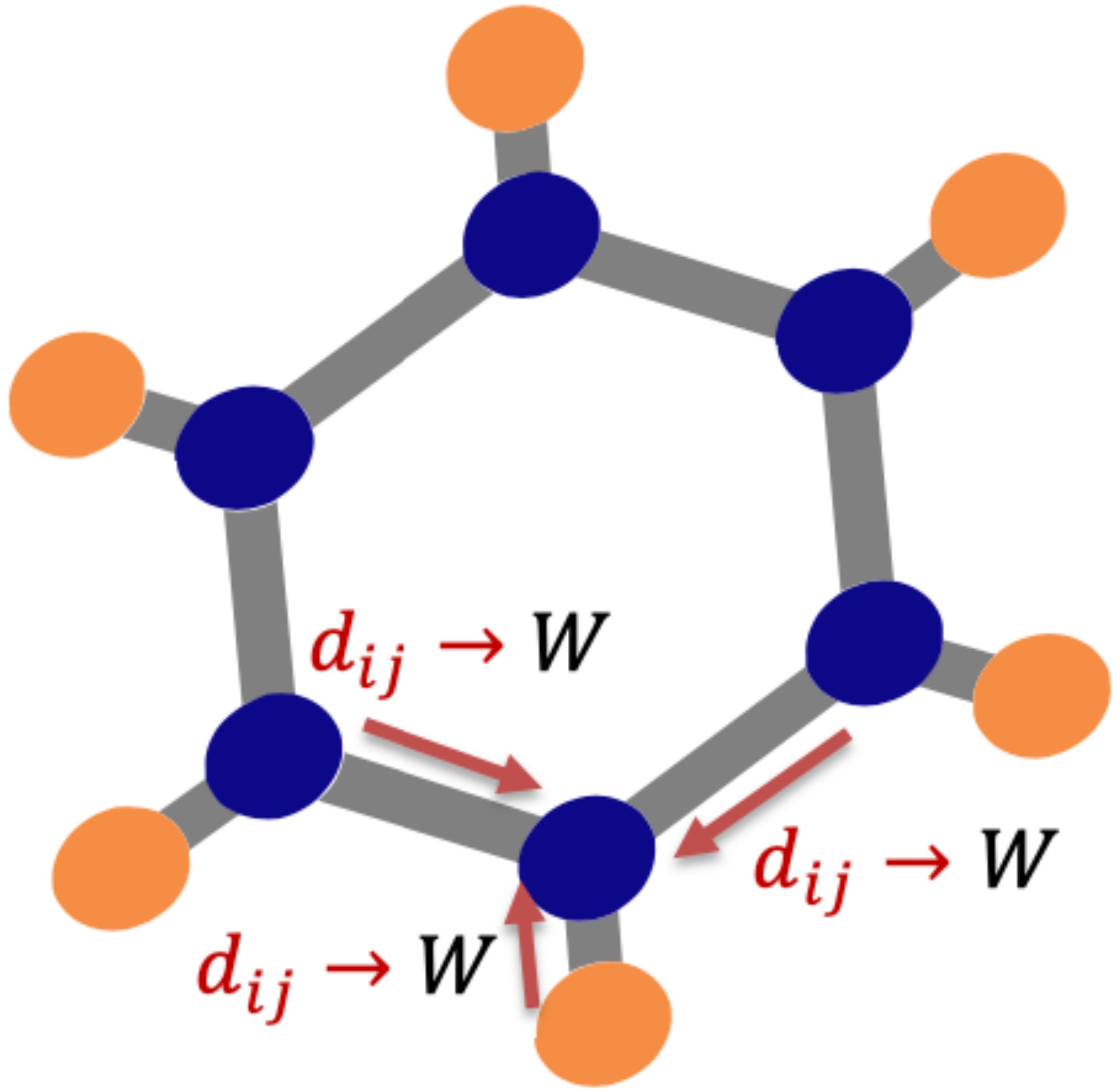


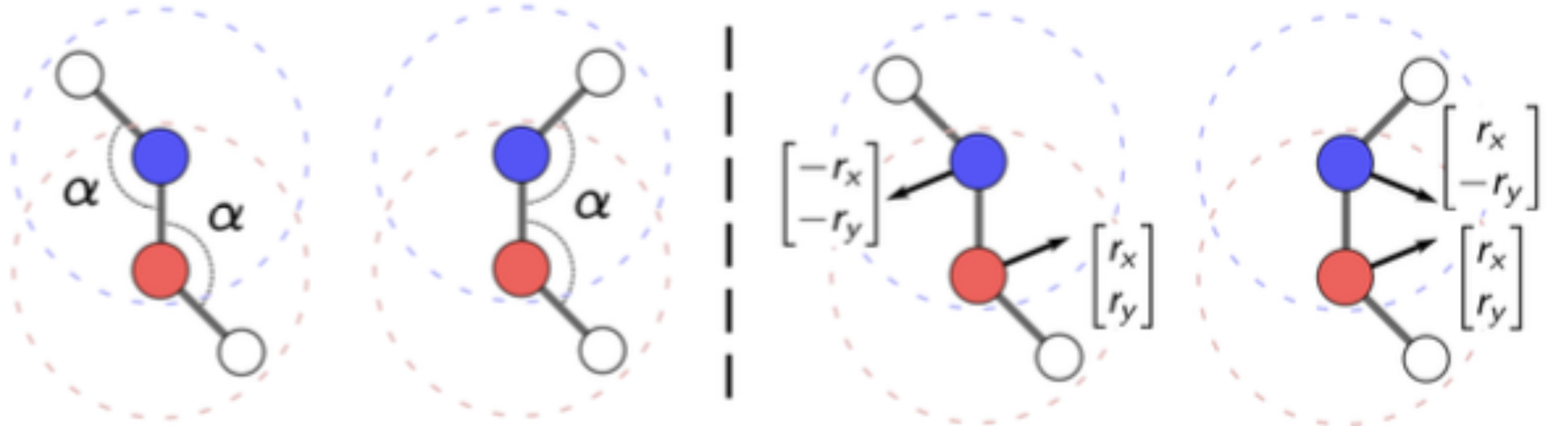
Equivariant GNN: PaiNN (2021)

One architecture per community

- **PaiNN** still take learnable weights W conditioned on the relative distance $\|\vec{r}_{ij}\|$ to control message passing

- However, differently, in PaiNN each node has two features (both scalar features s_i and vector features v_i)

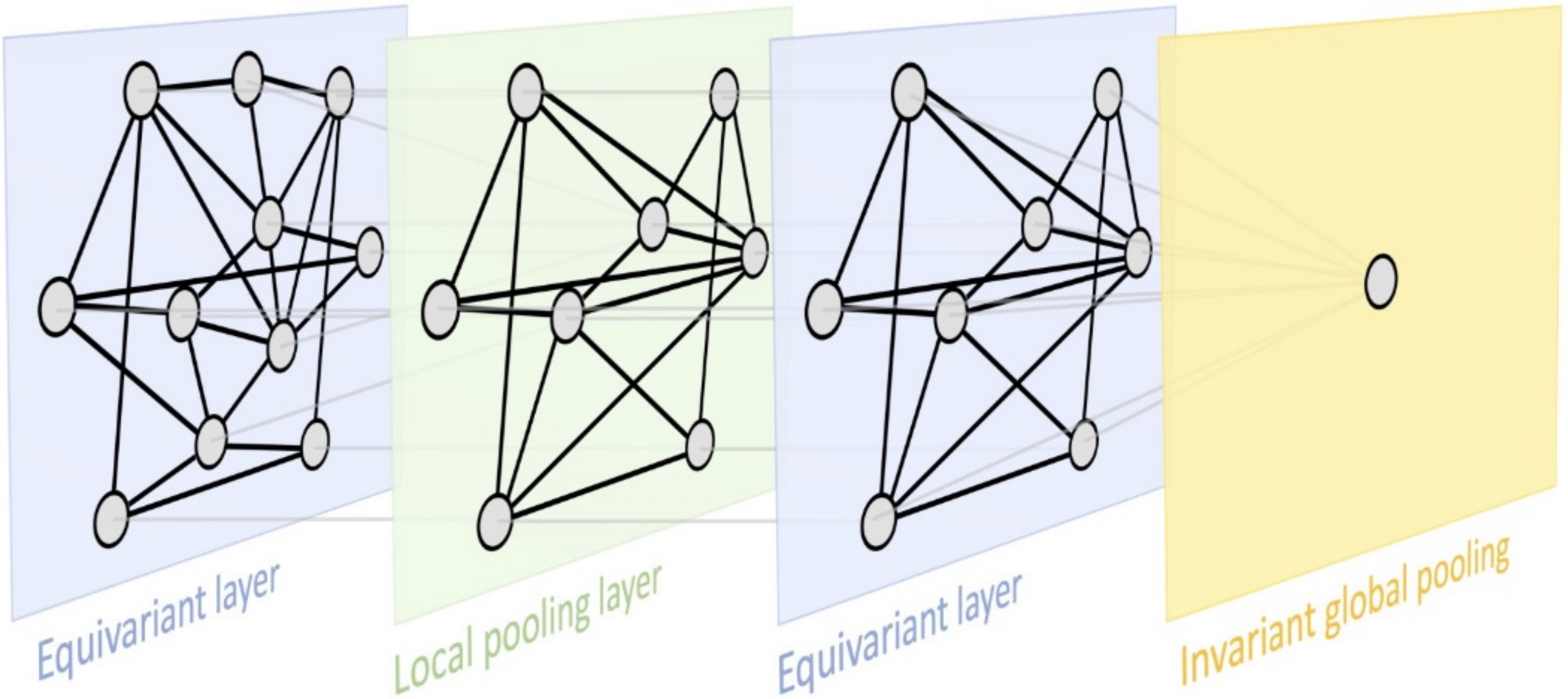




Schütt, Kristof, Oliver Unke, and Michael Gastegger. "Equivariant message passing for the prediction of tensorial properties and molecular spectra." *International Conference on Machine Learning*. PMLR, 2021.

The Geometric GNN blueprint

Stack equivariant layers with an optional invariant pooling

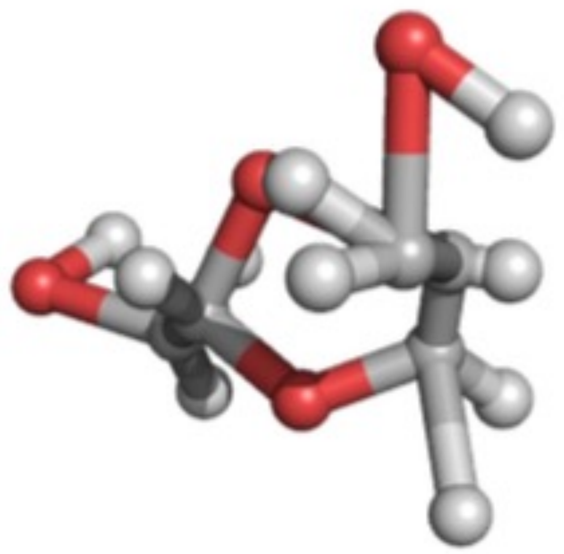


5. Outlook to Applications

Geometric GNNs for Science

Structural Bioinformatics plays a key role

- Accelerate scientific simulation
  - Molecule/Protein Design
  - Biomolecule structure prediction
  - Protein-molecule interaction
  - Molecular simulation

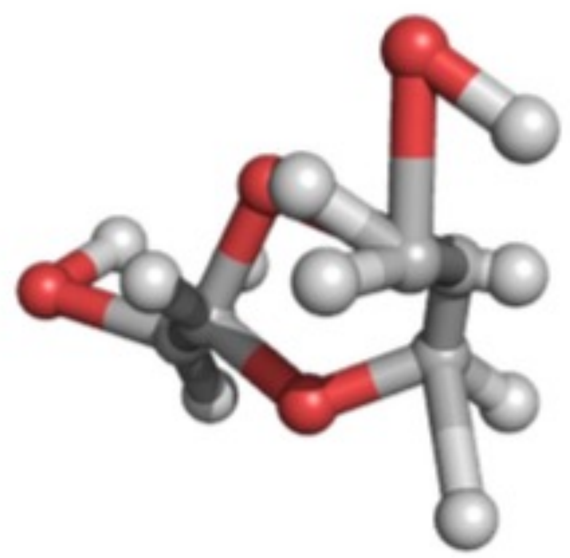


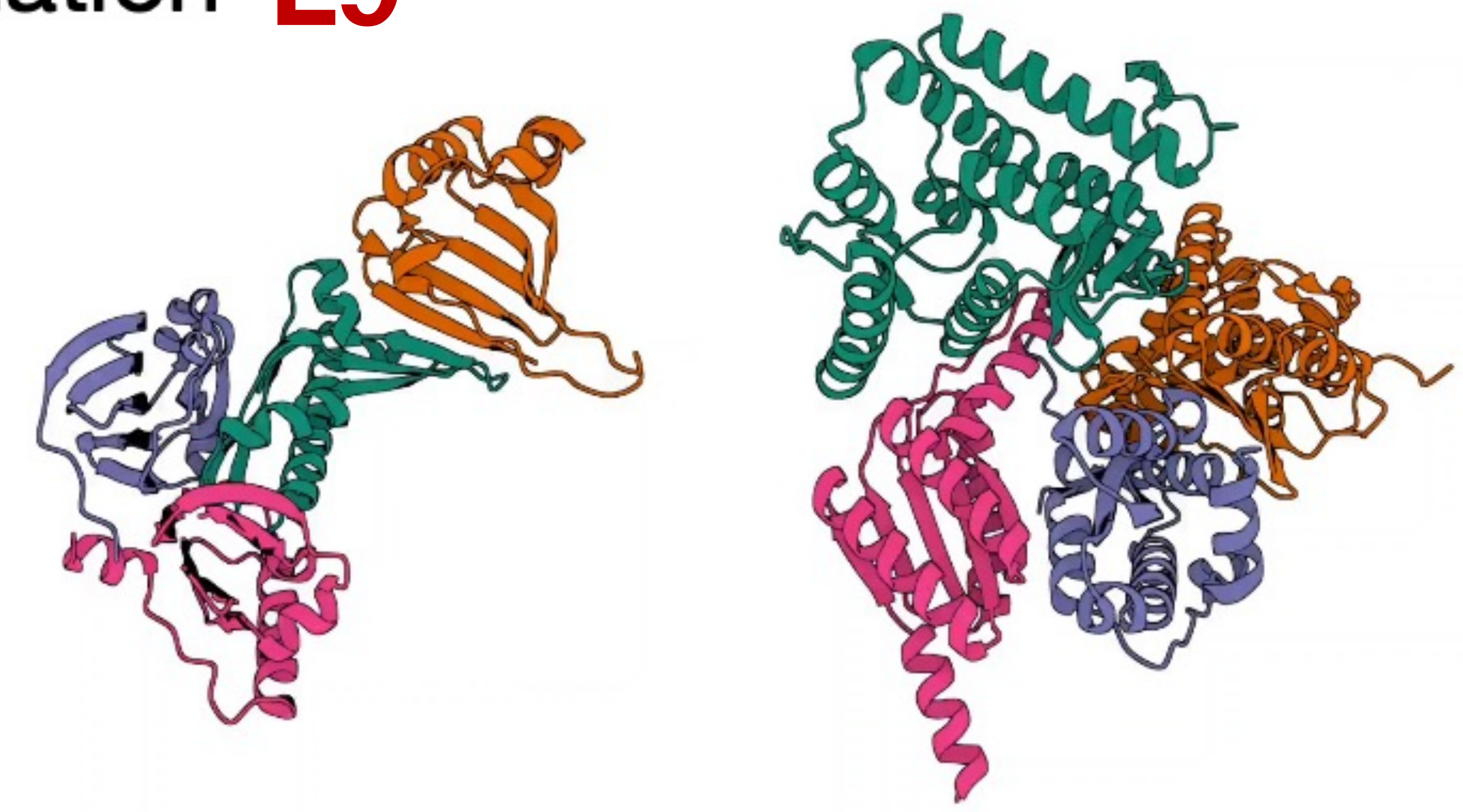
https://generatebiomedicines.com/chroma

Geometric GNNs for Science

Structural Bioinformatics plays a key role

- Accelerate scientific simulation
  - Molecule/Protein Design **L8**
  - Biomolecule structure prediction **L6**
  - Protein-molecule interaction **L10**
  - Molecular simulation **L9**





https://generatebiomedicines.com/chroma

🥡 Takeaway 🥡

We can model molecules as **graphs** and process them via **GNNs.** If we want to leverage geometric information, we can use **Geometric GNNs.**