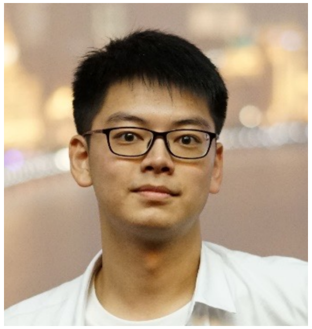
Guest lecture, Stanford CS224W Machine Learning with Graphs

Geometric Graph Learning
From Representation to Generation



Minkai Xu

Jure Leskovec

PROJECT SUPPORT

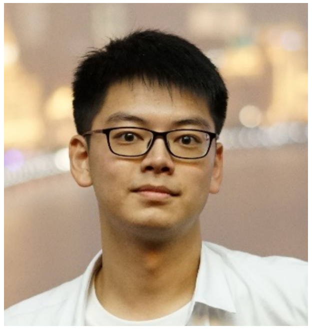
Thanks for your feedback on projects! To provide additional project support:

- TAs have been assigned as **project mentors** – we will release the list on Ed tonight
- Transparent **grading rubric** for final project – we will release by end of Friday (3/10)
- **Project OH**: Serina's OH (Thu 10am-12pm) and Hamed's OH (Fri 10am-12pm) are now dedicated to project questions
- If you can't make their OH, you can still post on Ed or contact your TA mentor!

CS224W: Machine Learning with Graphs
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

Guest lecture, Stanford CS224W Machine Learning with Graphs

Geometric Graph Learning From Representation to Generation



Minkai Xu

Jure Leskovec

Guest lecture, Stanford CS224W Machine Learning with Graphs

Geometric Graph Learning
From Representation to Generation

Minkai Xu

https://minkaixu.com

Stanford
ARTIFICIAL
INTELLIGENCE

STANFORD
INFOLAB

LELAND STANFORD JUNIOR UNIVERSITY
ORGANIZED 1891

Outline

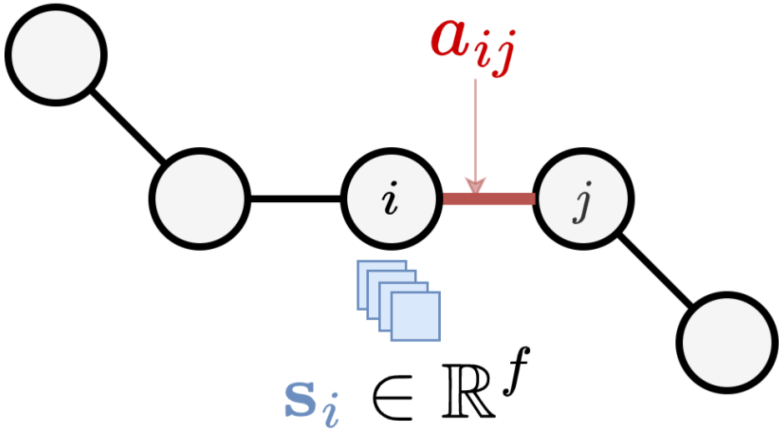
- Geometric Graphs
- Geometric Graph NNs
    - Invariant GNNs
    - Equivariant GNNs
- Geometric Generative Models
    - Geometric Diffusion Models

Outline

- **Geometric Graphs**
- Geometric Graph NNs
  - Invariant GNNs
  - Equivariant GNNs
- Geometric Generative Models
  - Geometric Diffusion Models

Graphs

A graph G = (A, S) is a set V of n nodes connected by edges. Each node has scalar attributes, e.g. atom type for molecules.



- A: an $n \times n$ adjacency matrix.
- S ∈ $R^{n \times f}$: scalar features.

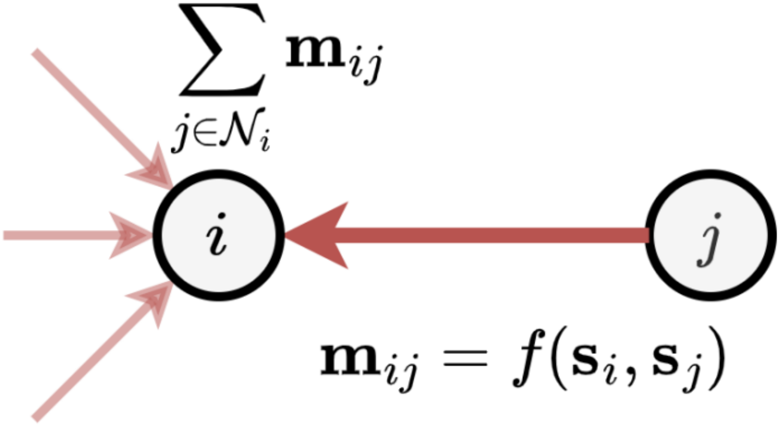
Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." *Advances in neural information processing systems* 30 (2017).
Joshi, Chaitanya K., et al. "On the expressive power of geometric graph neural networks."

Message Passing Neural Nets

- Node features are updated from iteration t to t+1 via learnable **permutation invariant** neighborhood aggregate AGG and update UPD:

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

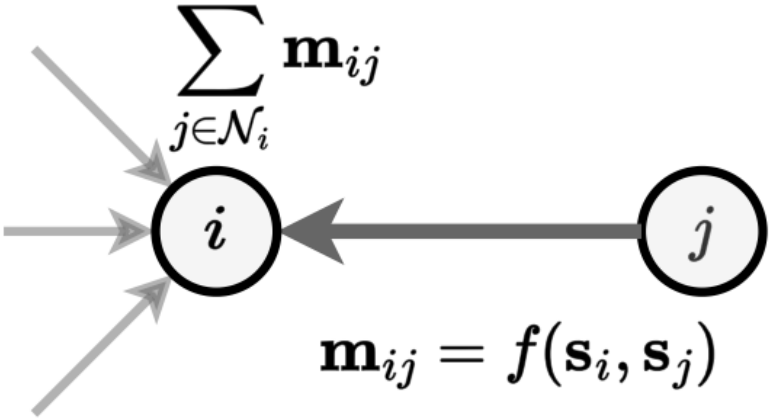
$$\boldsymbol{s}_i^{(t+1)} = \textsc{Upd}\left(\boldsymbol{s}_i^{(t)}\ ,\ \boldsymbol{m}_i^{(t)}\right)$$



$\sum_{j\in\mathcal{N}_i} \mathbf{m}_{ij}$
i
j
$\mathbf{m}_{ij} = f(\mathbf{s}_i, \mathbf{s}_j)$

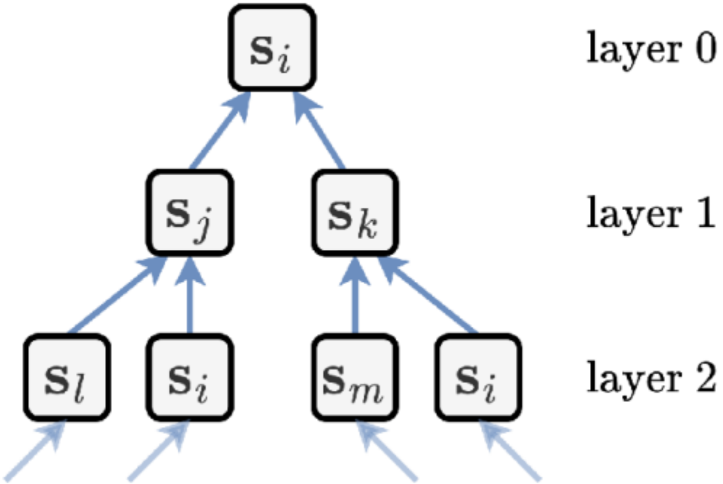
Graph Neural Networks

- Message passing updates node features using local aggregation


$\sum_{j \in \mathcal{N}_i} \mathbf{m}_{ij}$
i
j
$\mathbf{m}_{ij} = f(\mathbf{s}_i, \mathbf{s}_j)$

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

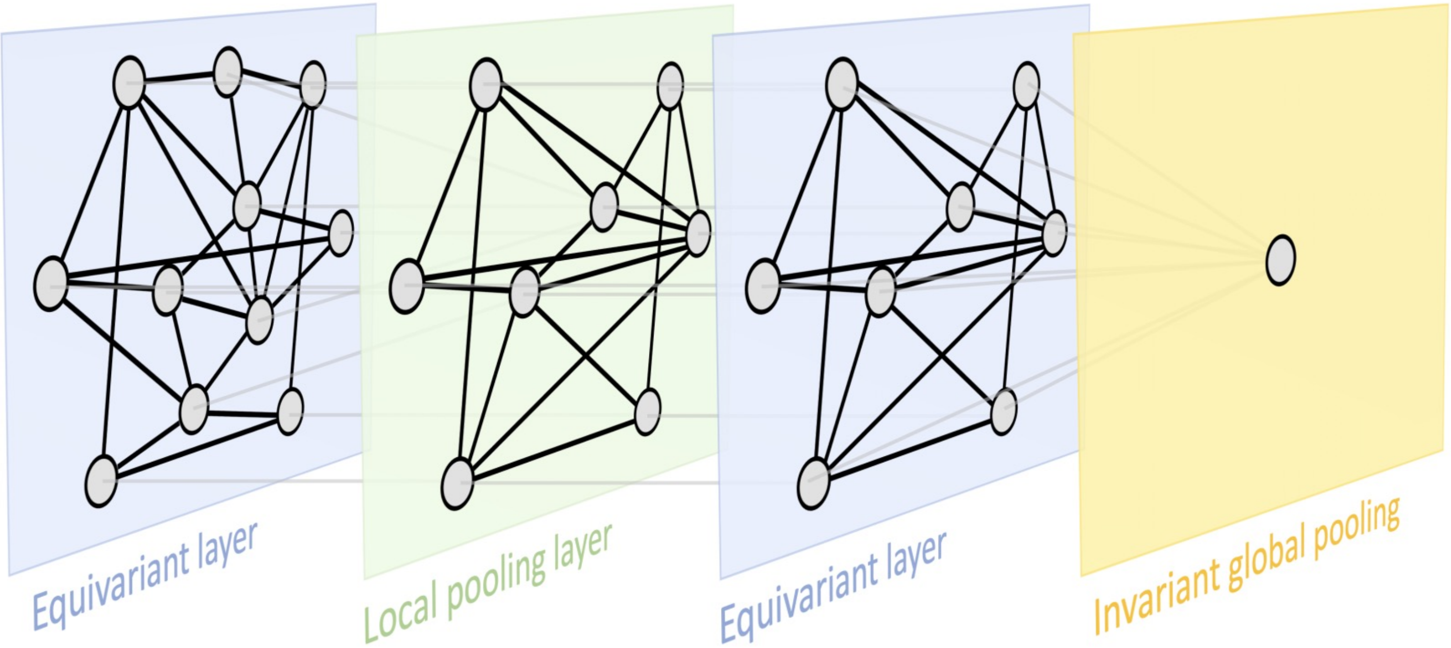
$$\boldsymbol{s}_i^{(t+1)} := \textsc{Upd}\left(\boldsymbol{s}_i^{(t)}\,,\, \boldsymbol{m}_i^{(t)}\right),$$



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

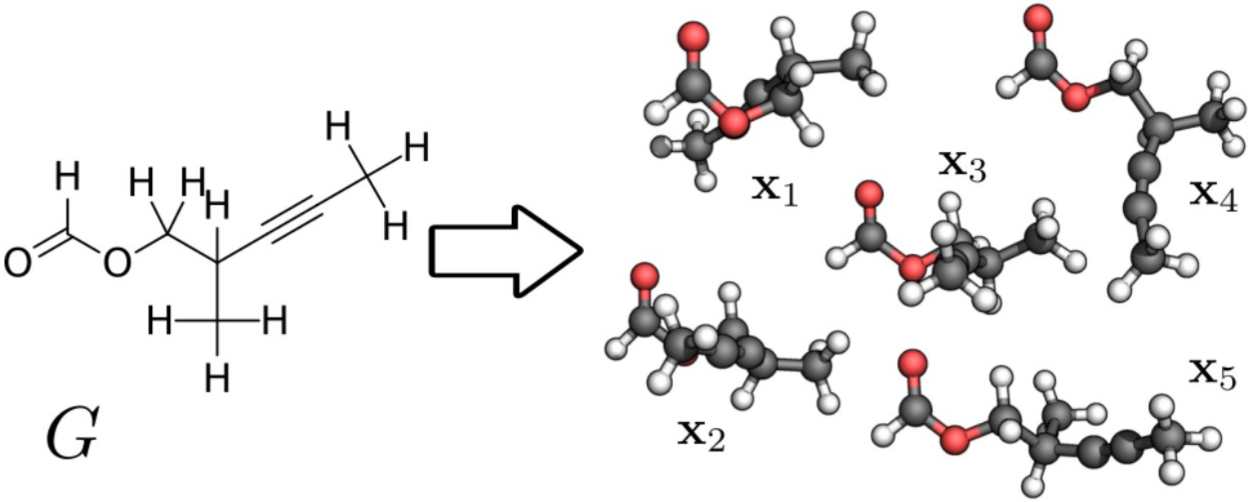
Normal Graph Neural Networks

- Advanced GNN layers make pooling over node features, which are then used to make a graph-level prediction.



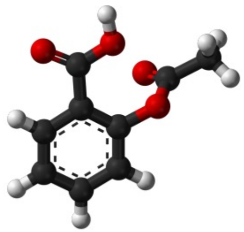
Molecular Graphs

- Molecules can be represented as a graph G with node features s_i and edge features a_{ij}.
  - Node features: atom type, atom charges…
  - Edge features: valence bond type…
  - However, sometimes, we also know the 3D positions x_i, which is actually more informative

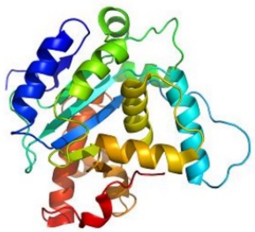


Simm, Gregor NC, and José Miguel Hernández-Lobato. "A generative model for molecular distance geometry." *ICML 2020*

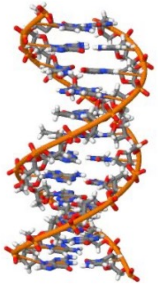
More Geometric Graphs



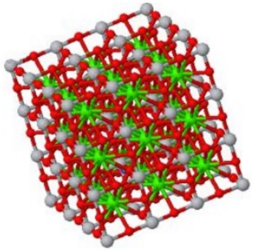
Small Molecules



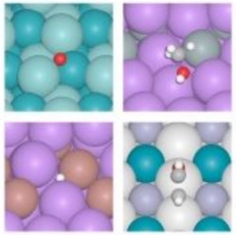
Proteins



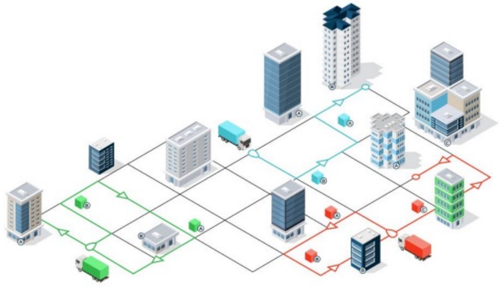
DNA/RNA



Inorganic Crystals

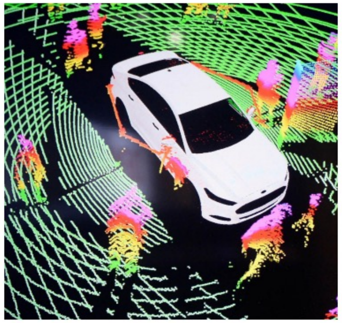


Catalysis Systems



Transportation & Logistics

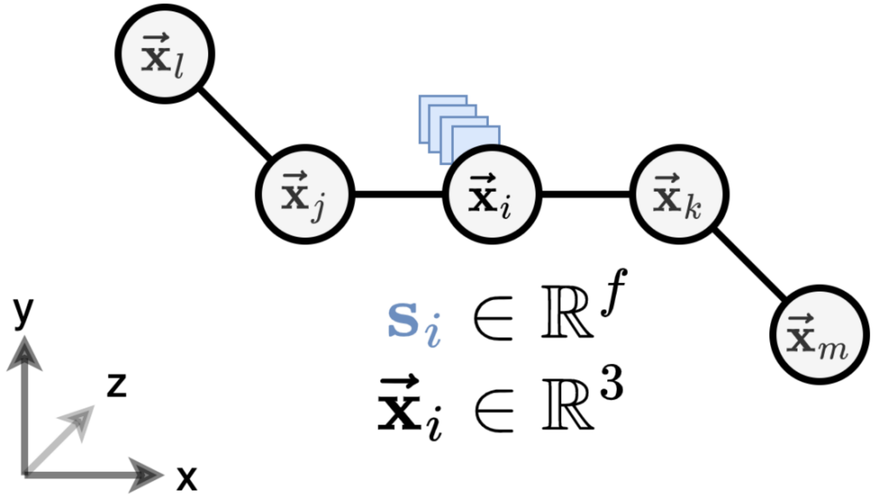
Robotic Navigation



3D Computer Vision

Geometric Graphs

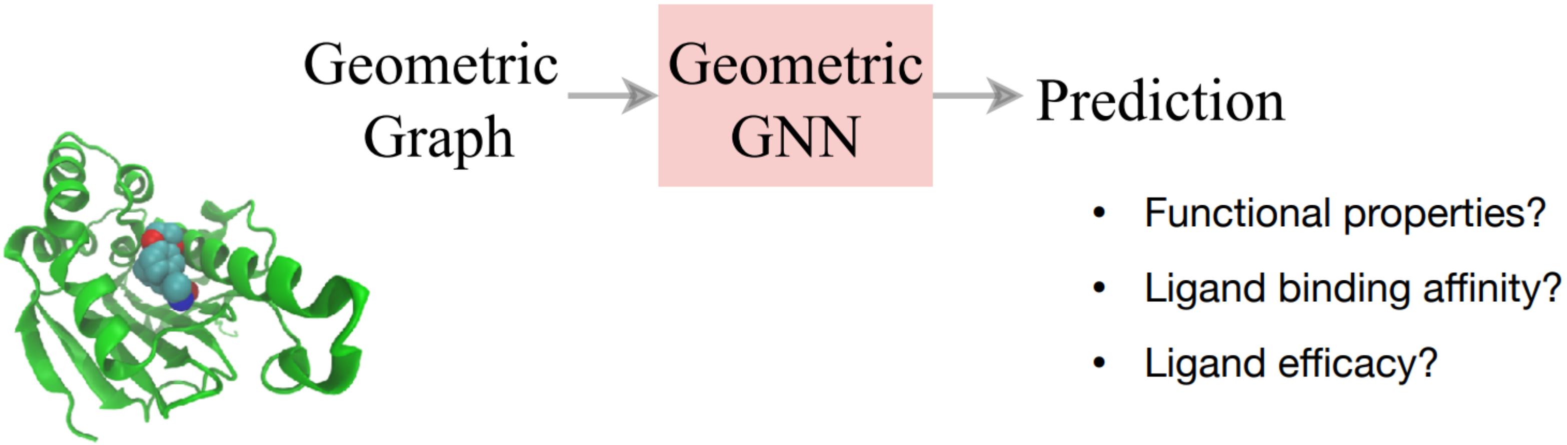
- A geometric graph G = (A, S, X) is a graph where each node is embedded in d-dimensional **Euclidean space**:



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

Broad Impact on Sciences

- Supervised Learning: Prediction
  - Properties prediction

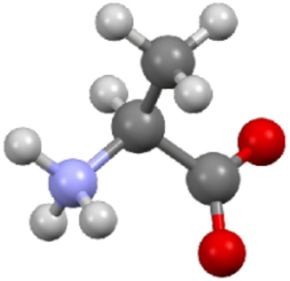


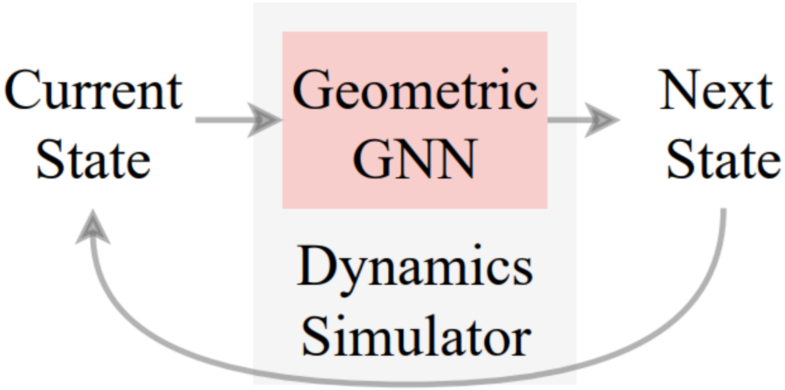
Geometric Graph → Geometric GNN → Prediction

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

Broad Impact on Sciences

- Supervised Learning: Structured Prediction
  - Molecular Simulation



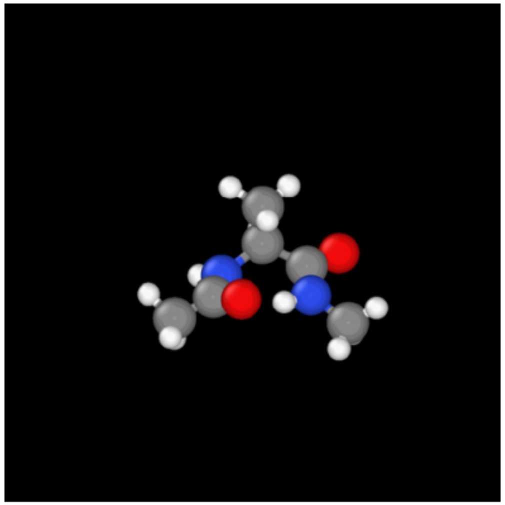


Current State

Geometric GNN

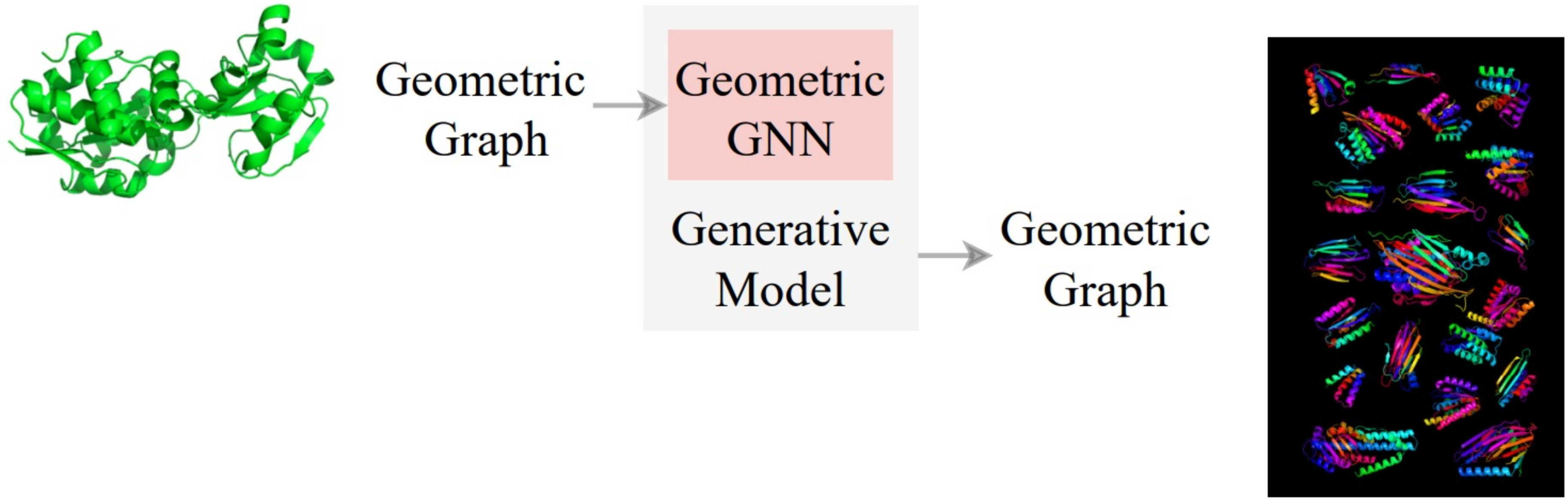
Next State

Dynamics Simulator



Broad Impact on Sciences

- Generative Models
  - Drug or material design

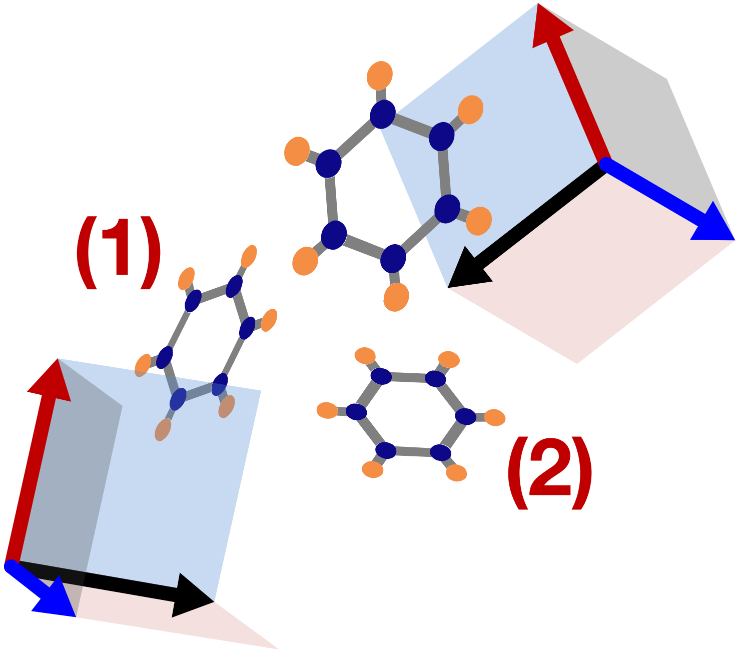


Geometric Graph → Geometric GNN

Generative Model → Geometric Graph

What’s the obstacle?

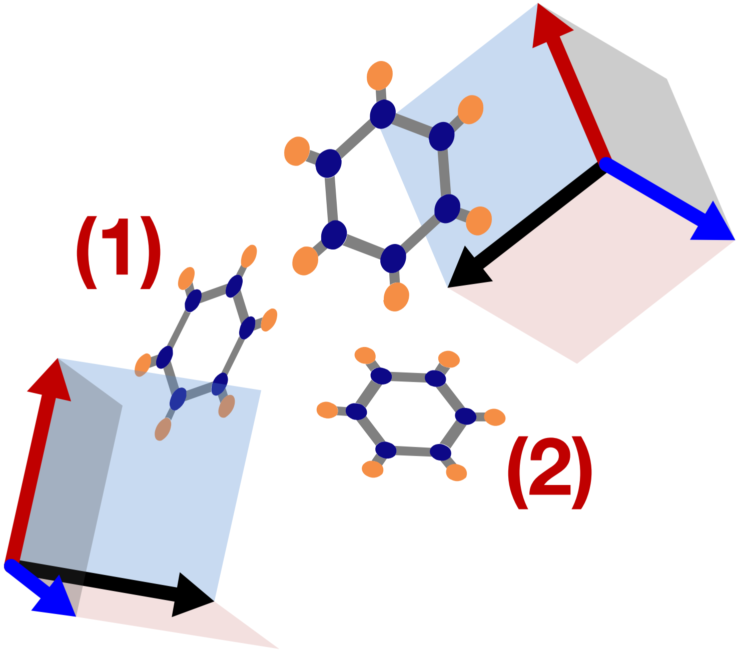
- To describe geometric graphs we use coordinate systems
    - (1) and (2) use different coordinate systems to describe the **same** molecular geometry.
- We can describe the transform between coordinate systems with symmetries of Euclidean space
    - 3D rotations, translations



Thomas, Nathaniel, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds."

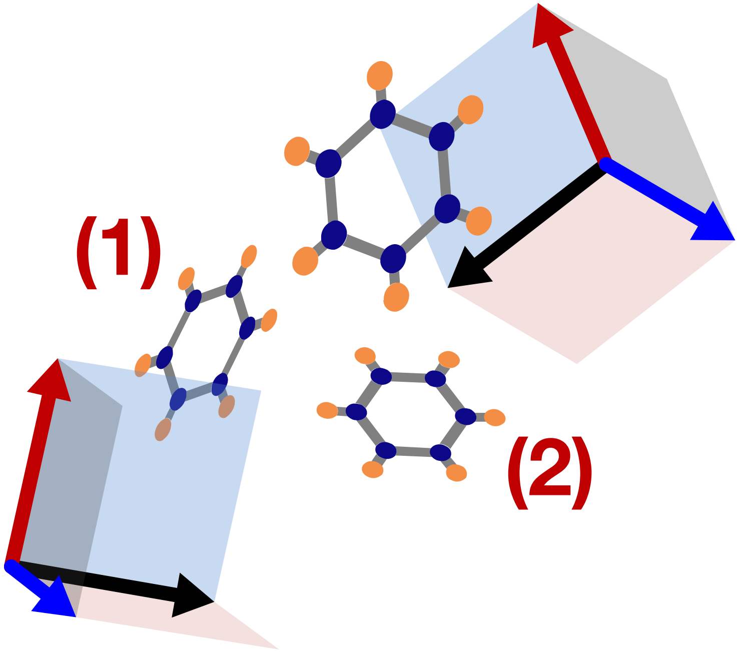
Physical Symmetry Groups

- 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!**



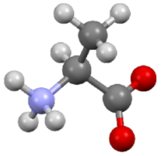
Symmetry of Inputs

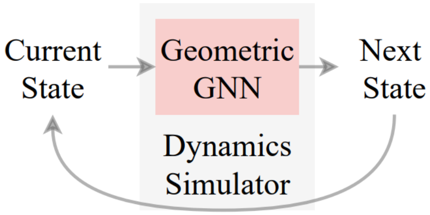
- **However, output of traditional GNNs given (1) and (2) as completely different!**
- We want our GNNs can see (1) and (2) as the same system though described differently…
- i.e., **we want design Geometric GNNs aware of symmetry!**

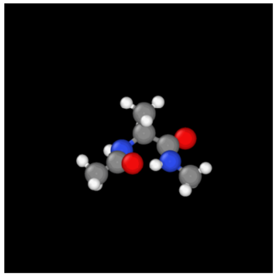


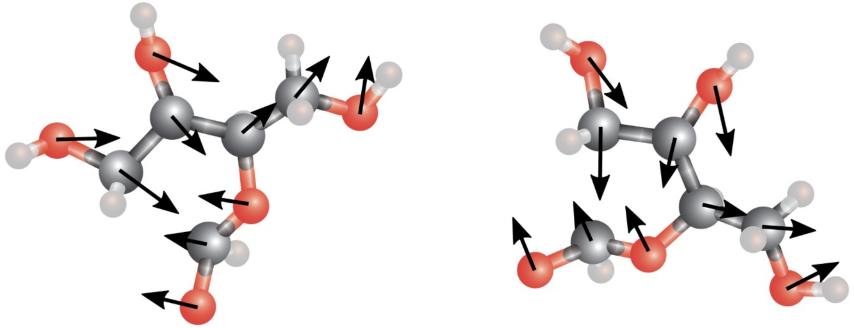
Symmetry of Outputs

- 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)









Equivariance

- Formal definition of Equivariance:
  a function $F: X \to Y$ 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

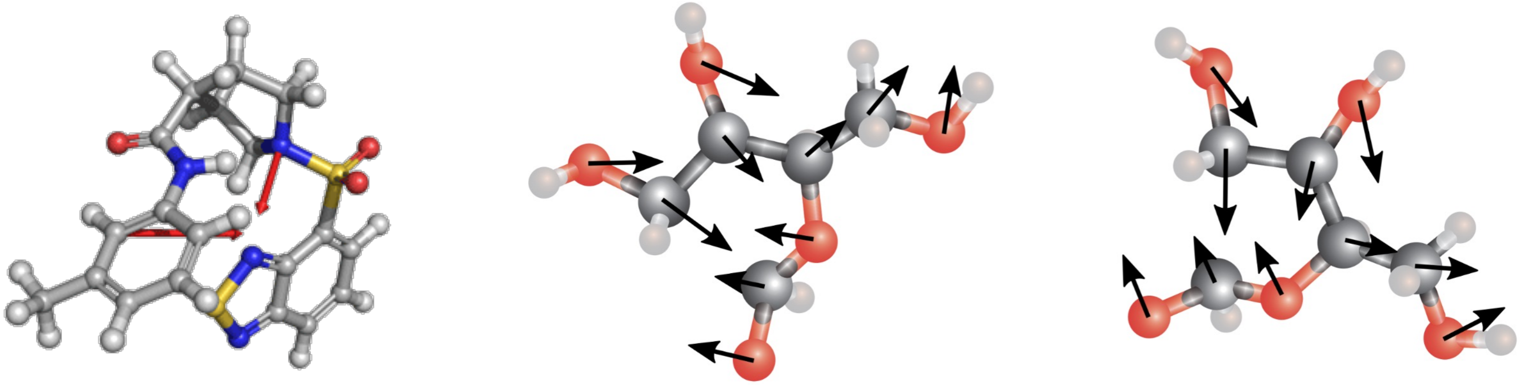
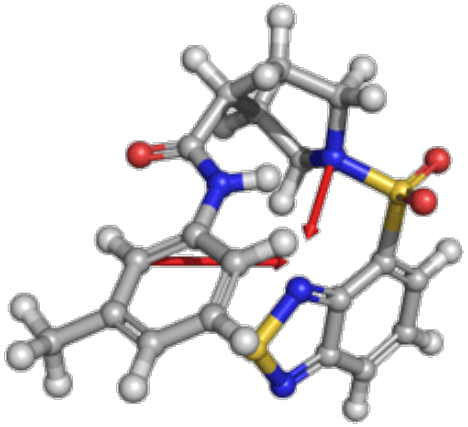
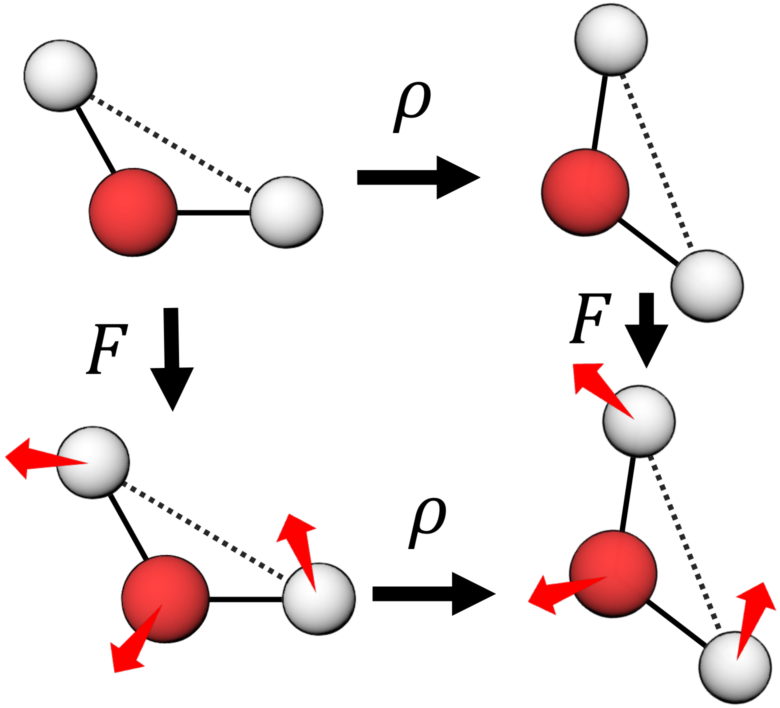


Illustration: 3D Rotation Equivariance

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

The equation says that applying the ρ on the input has the same effect as applying it to the output.





Visual explanation of the equivariance

A GIF illustrating the rotation equivariance of atomic forces.
Two red arrows stand for forces acting on atoms, which rotate together with the molecule.

Shi, Chence, et al. "Learning gradient fields for molecular conformation generation." *International Conference on Machine Learning*. PMLR, 2021.

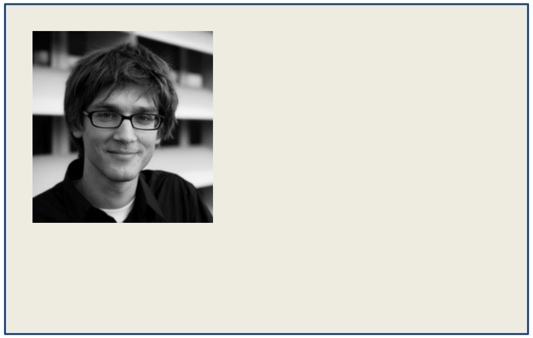
Invariance

- 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.

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



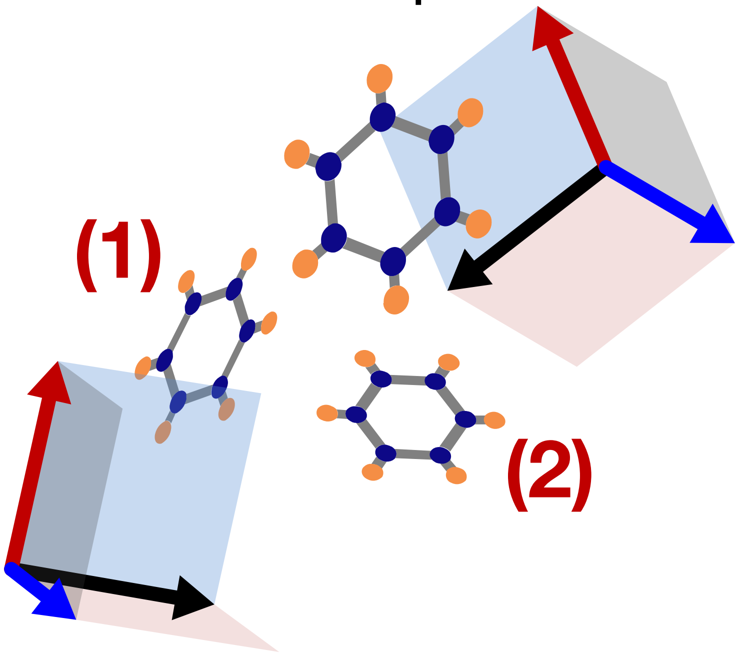
✓ Yes, Prof. Leskovec.

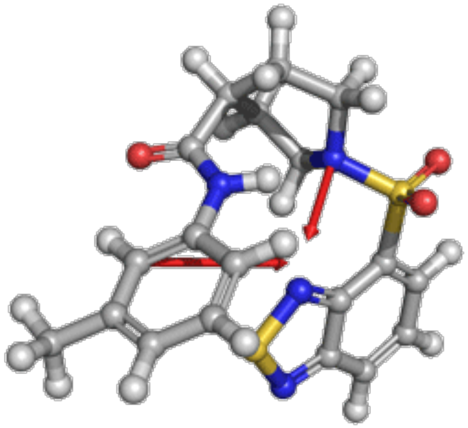
After roto-translation…

✓ Still Prof. Leskovec!

Invariance & Equivariance

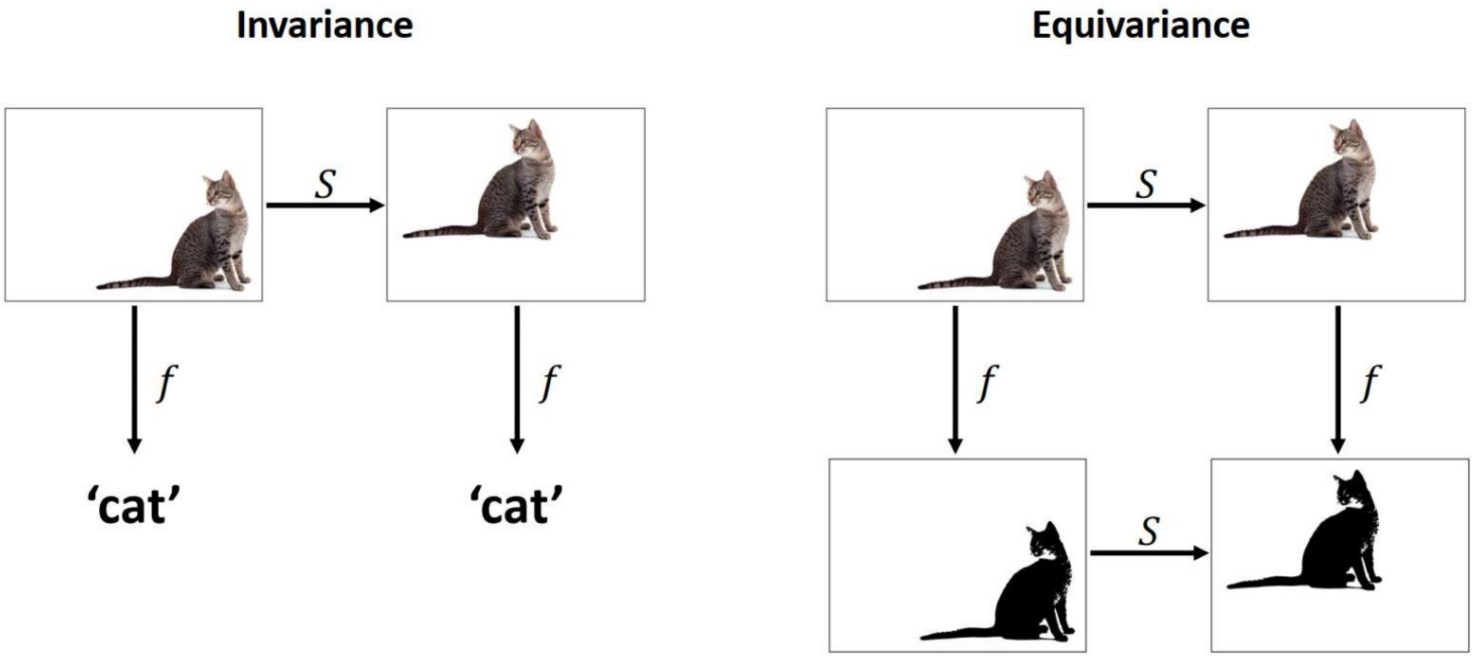
- For geometric graphs, we consider 3D Special Euclidean (SE(3)) symmetries, e.g.:
  - structure **x** -> energy E **:** invariant scalars
  - structure **x** -> force **v:** equivariant tensors rotation equivariant and translation invariant





Invariance & Equivariance

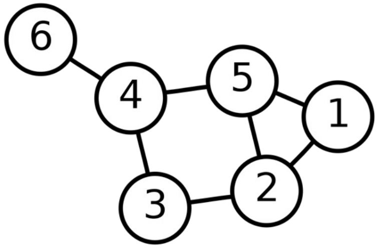
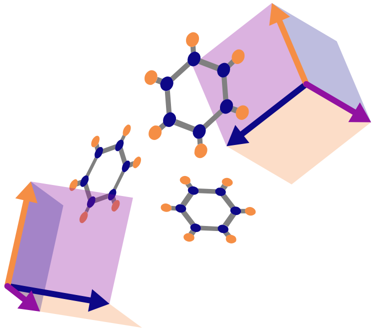
- The analogy in image domain...
  - Classification: invariant label
  - Segmentation: equivariant pixel coordinates



https://www.doc.ic.ac.uk/~bkainz/teaching/DL/notes/equivariance.pdf

Summarization

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***

Outline

- Geometric Graphs
- **Geometric Graph NNs**
  - Invariant GNNs
  - Equivariant GNNs
- Geometric Generative Models
  - Geometric Diffusion Models

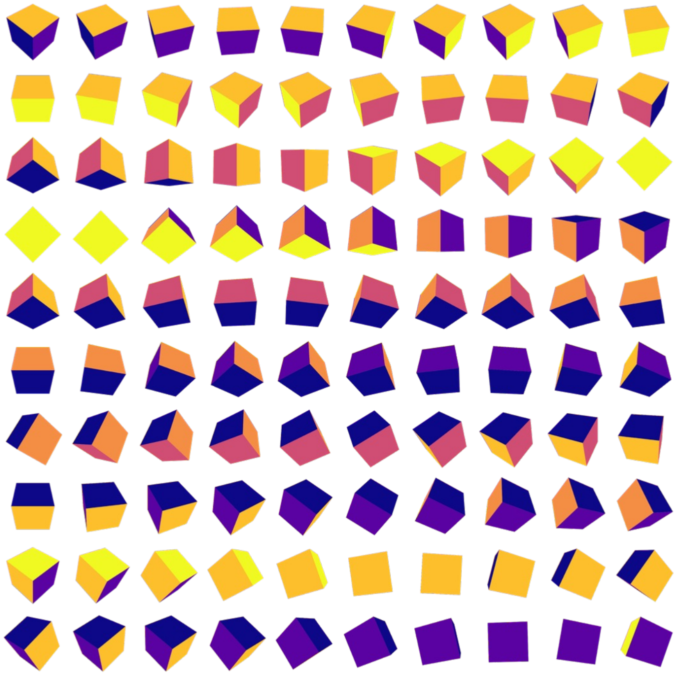
Handle Symmetry

- For ML models without handling symmetry: expensive data augmentation

  create more training data by augmenting original data to include all possible symmetries (rotations)

- Alternative: design Geometric GNNs!

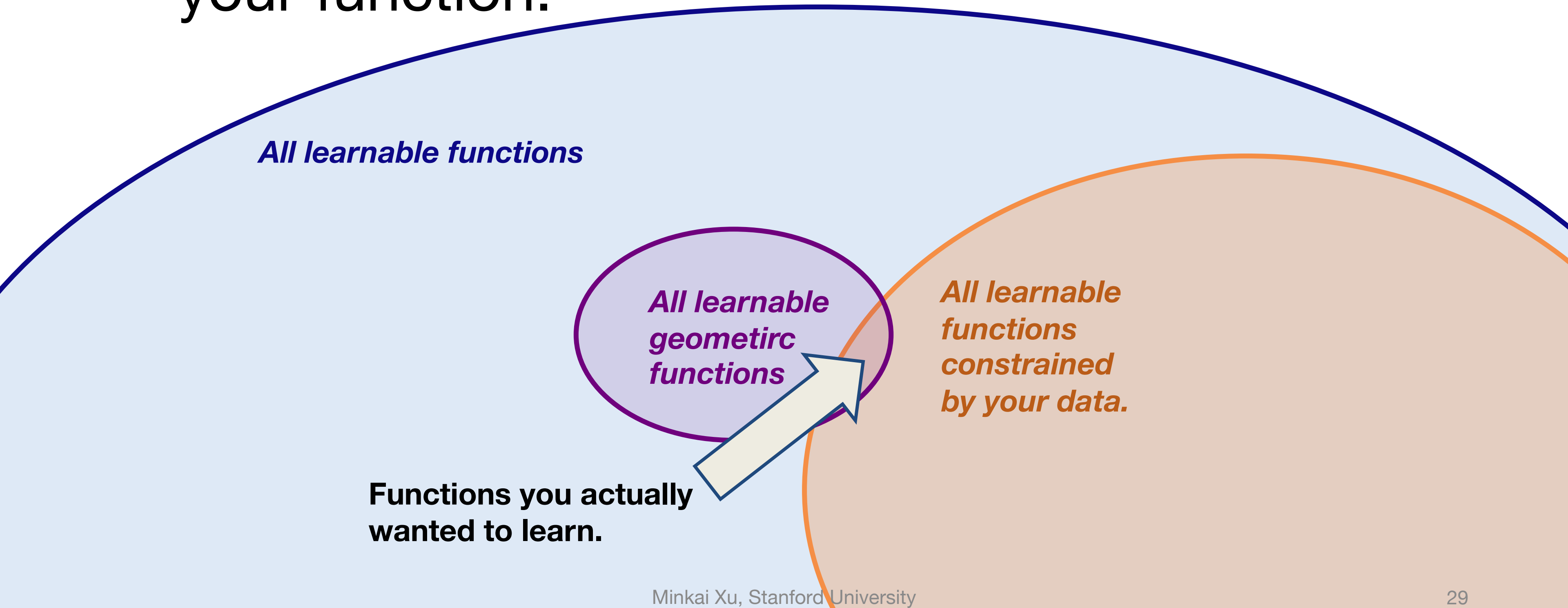
training without rotational symmetry



training with symmetry

Advantage

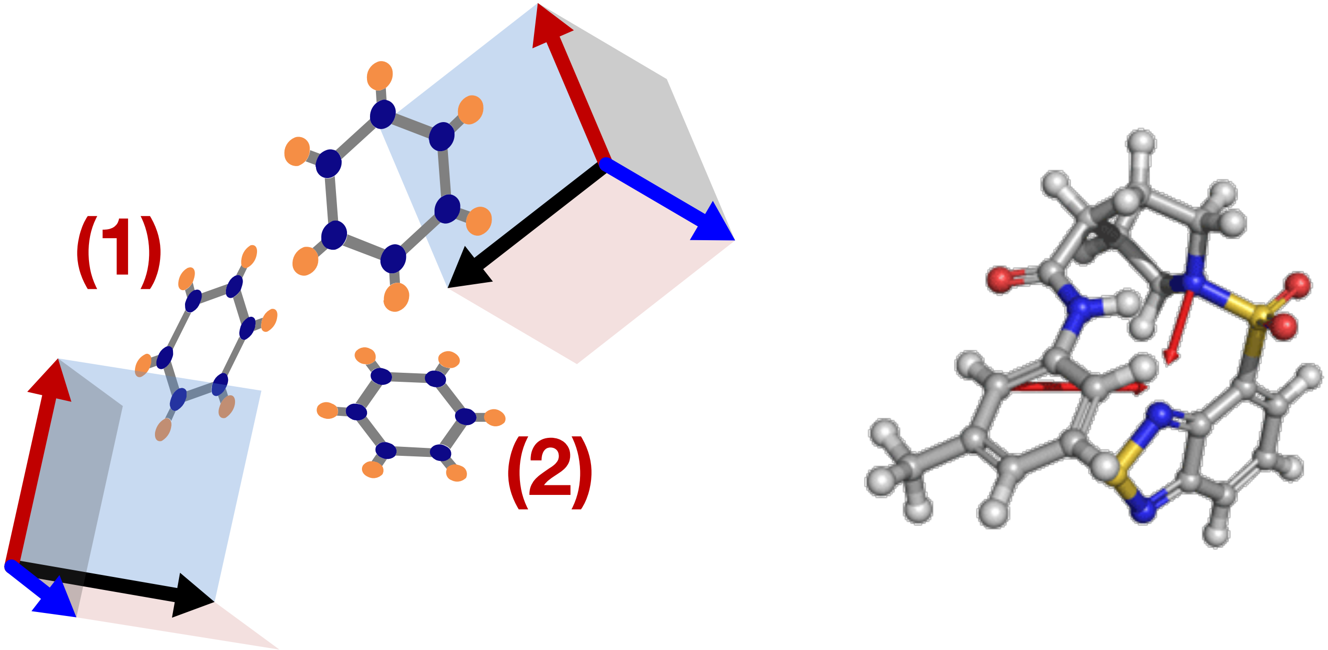
- You can substantially shrink the space of functions you need to optimize over.
- This means you need less data to constrain your function.


All learnable functions
All learnable geometirc functions
All learnable functions constrained by your data.
Functions you actually wanted to learn.

Geometric GNNs

Two classes of Geometric GNNs:

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

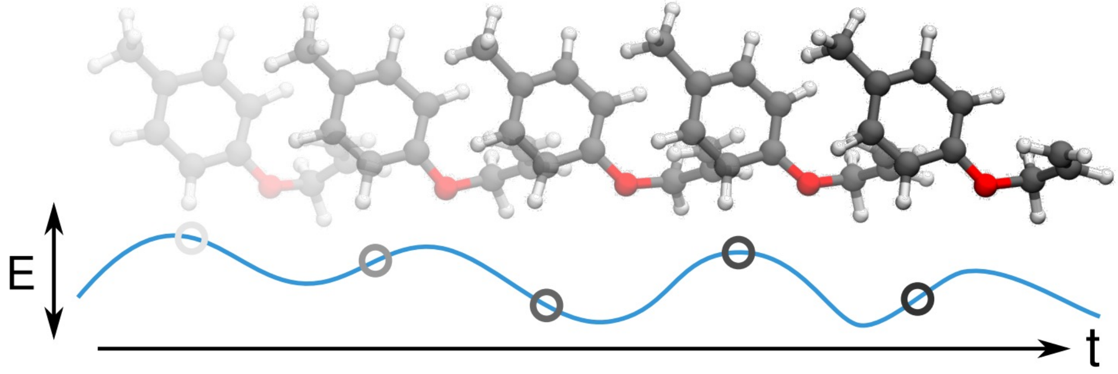


Invariant functions vs. Equivariant functions

Molecular Dynamics Simulations

- For simulating the stable structure of molecular geometries: computationally costly quantum mechanical calculations
  - Energy $E(r_1, \ldots, r_n)$
  - Forces $\mathbf{F}_i(\mathbf{r}_1, \ldots, \mathbf{r}_n) = -\frac{\partial E}{\partial \mathbf{r}_i}(\mathbf{r}_1, \ldots, \mathbf{r}_n).$

r: atomic coordinates

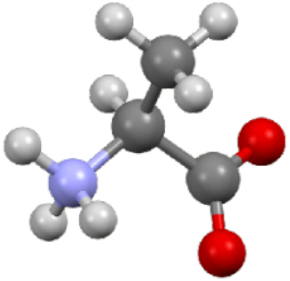


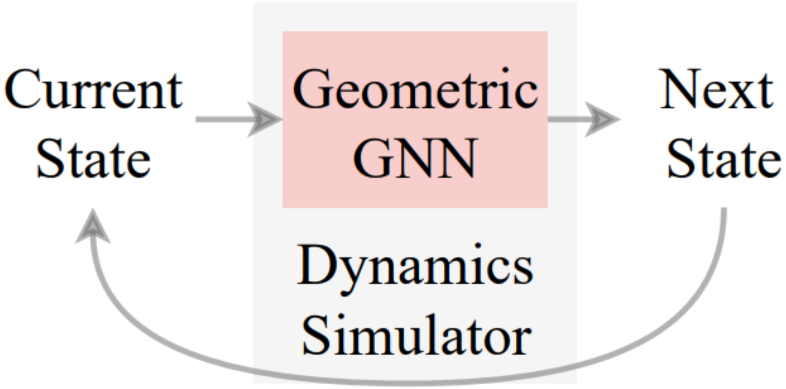
$$\hat{H}\Psi = E\Psi$$

Method	Complexity
Hartree Fock	$O(n^3) - O(n^4)$
Density Functional Theory	$O(n^3) - O(n^4)$
MP2	$O(n^5)$
CCSD	$O(n^6)$
CCSD(T)	$O(n^7)$
Full CI	$O(n!)$

Molecular Dynamics Simulations

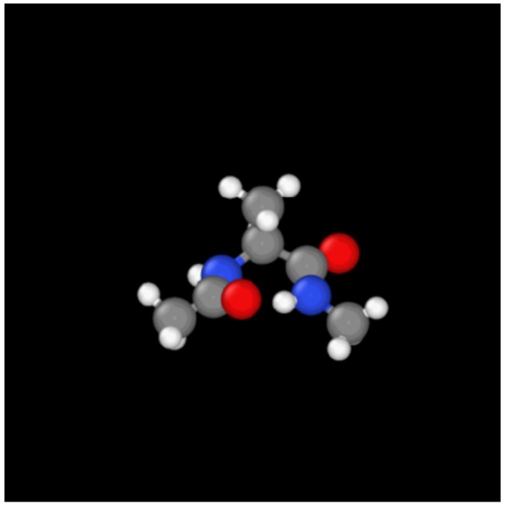
- Usage: forces can be used to optimize the structure by $X^t + \boldsymbol{F} \rightarrow X^{t+1}$ (simulation)



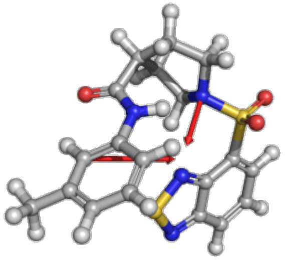


Current State → Geometric GNN → Next State

Dynamics Simulator



F: forces acting on all atoms for optimizing the structures



For ML Models…

- Problem Definition
- Inputs:
  molecular graphs with atom types $X = (x_1, \ldots, x_n) \in R^d$ and positions $R = (r_1, \ldots, r_n) \in R^3$
- Predict:
  the molecular total energy $E(r_1, \ldots, r_n)$ (invariant)
  forces $\mathrm{F} = (f_1, \ldots, f_n)$ acting on each atom (equivariant).
  - Forces are partial derivatives of energy function.

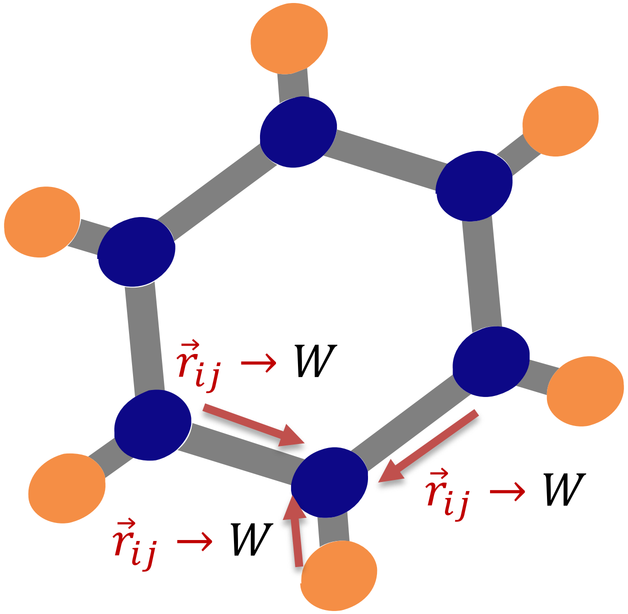
$$\mathbf{F}_i(\mathbf{r}_1, \ldots, \mathbf{r}_n) = -\frac{\partial E}{\partial \mathbf{r}_i}(\mathbf{r}_1, \ldots, \mathbf{r}_n).$$

Invariant GNNs: SchNet

- **SchNet** updates the node embeddings at the l^{th} layer by message passing layers

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

  - A weight matrix W is determined by the relative position from neighbor atoms j to i
  - This kernel matrix $W: \mathbb{R}^3 \to \mathbb{R}^{d \times d}$ then controls interaction from neighbor atoms by $x_j \cdot W$
  - All the neighbor messages are aggregated by $\sum_j x_j \cdot W$



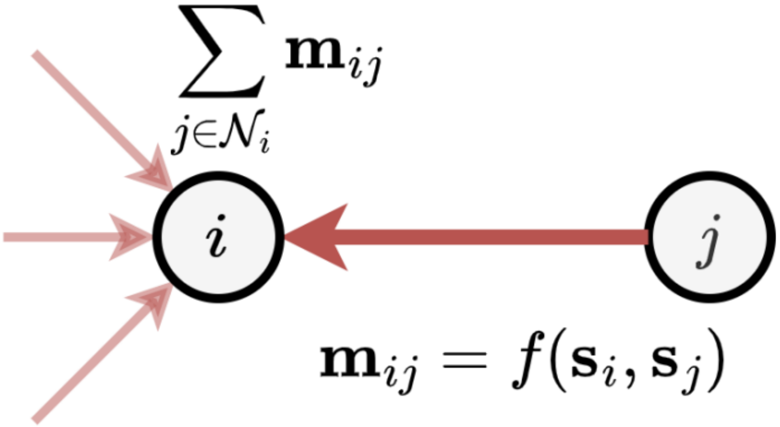
x^l: node embeddings at l layer
r: atomic coordinates

Schütt, Kristof T., et al. "Schnet–a deep learning architecture for molecules and materials." *The Journal of Chemical Physics* 148.24 (2018): 241722.

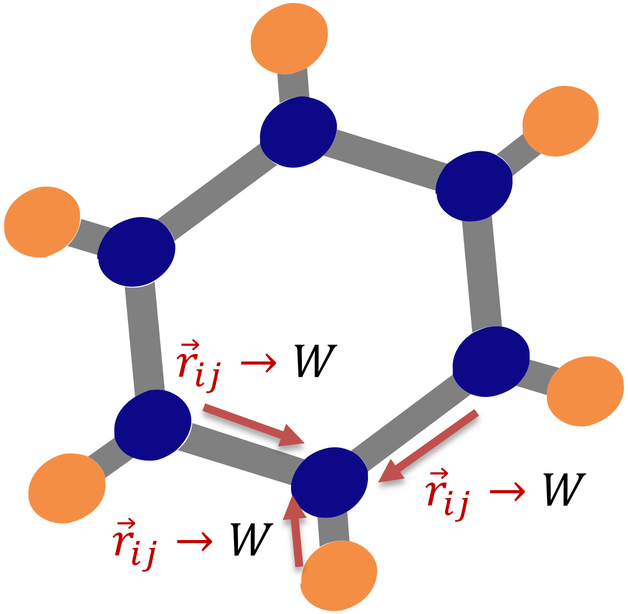
Invariant GNNs: SchNet

- **SchNet** updates the node embeddings at the l^{th} layer by message passing layers

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



$\sum_{j \in \mathcal{N}_i} \mathbf{m}_{ij}$
i
j
$\mathbf{m}_{ij} = f(\mathbf{s}_i, \mathbf{s}_j)$



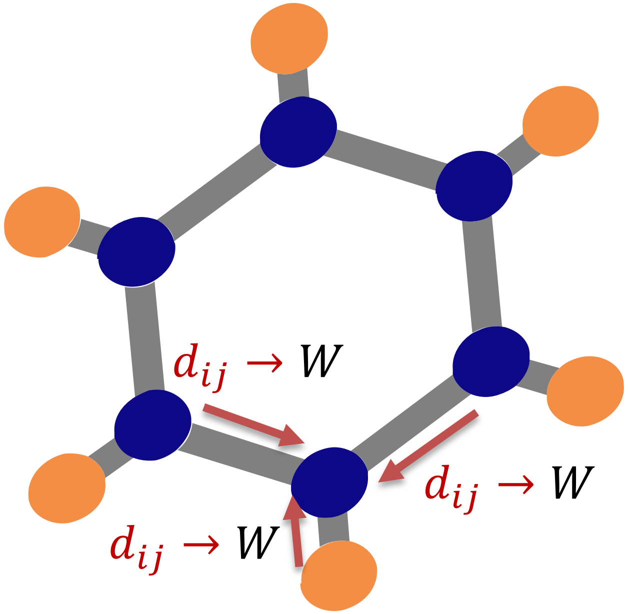
$\vec{r}_{ij} \rightarrow W$
$\vec{r}_{ij} \rightarrow W$
$\vec{r}_{ij} \rightarrow W$

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

Schütt, Kristof T., et al. "Schnet–a deep learning architecture for molecules and materials." *The Journal of Chemical Physics* 148.24 (2018): 241722.

Invariant GNNs: SchNet

- **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

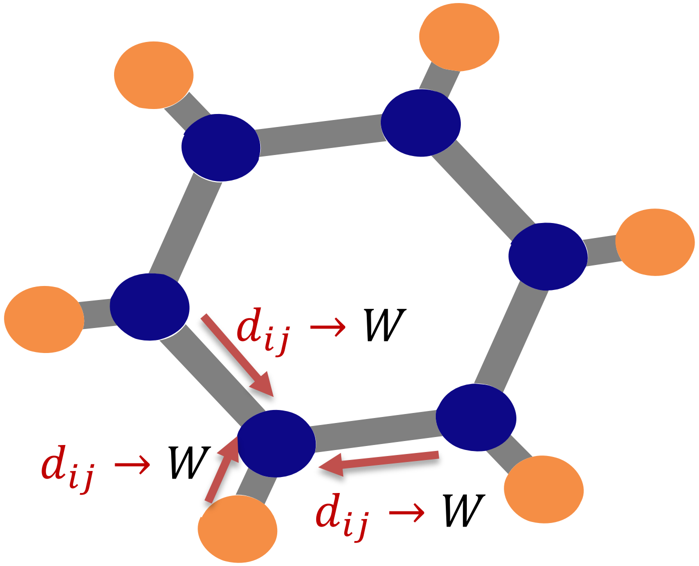


$$\mathbf{x}_i^{l+1} = (X^l * W^l)_i = \sum_j \mathbf{x}_j^l \circ W^l(\mathbf{r}_i - \mathbf{r}_j),$$

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

invariant GNNs: SchNet

- **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!



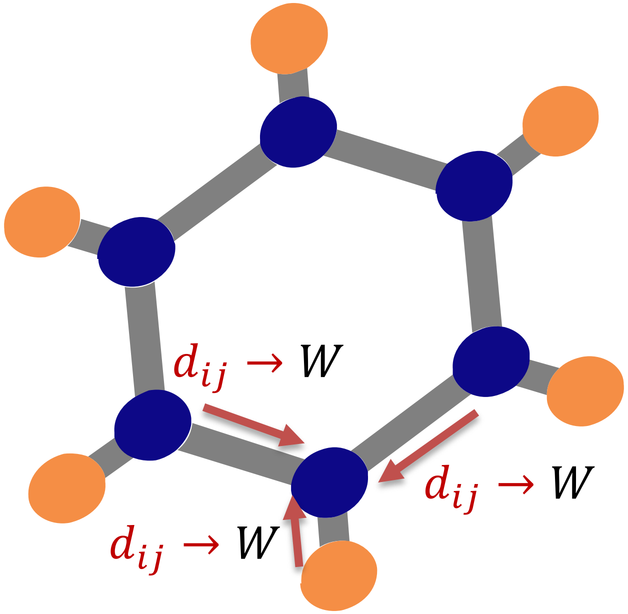
$$\mathbf{x}_i^{l+1} = (X^l * W^l)_i = \sum_j \mathbf{x}_j^l \circ W^l(\mathbf{r}_i - \mathbf{r}_j),$$

x^l: node embeddings at l layer

r: atomic coordinates

invariant GNNs: SchNet

- **SchNet** makes W invariant by **scalarizing** relative positions $\overrightarrow{r_{ij}}$ with relative distances $d_{ij} = \|\vec{r}_{ij}\|$:
  - Implementation details: Since d_{ij} is 1-dimensional, we need to expand to higher (300) dimension (better for training)
  - Radial Basis Functions (RBF): $e_k(\mathbf{r}_i - \mathbf{r}_j) = \exp(-\gamma\|d_{ij} - \mu_k\|^2)$ μ_k are chosen every $0.1A$ within $0A \leq \mu_k \leq 30A$ and $\gamma = 10A$



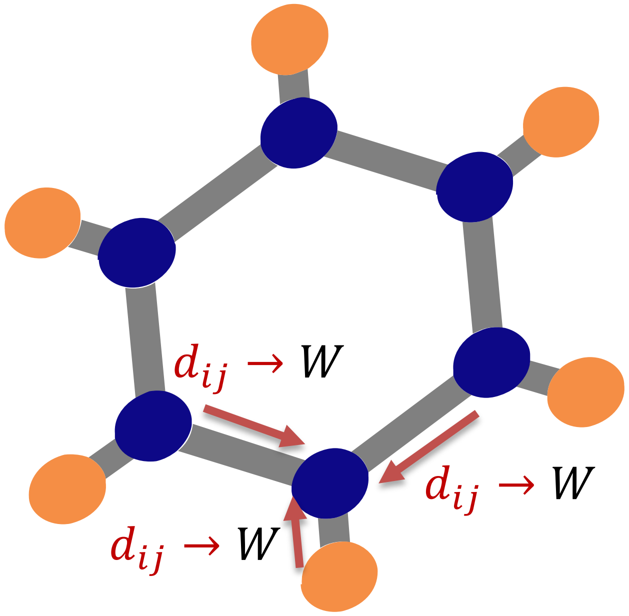
$$\mathbf{x}_i^{l+1} = (X^l * W^l)_i = \sum_j \mathbf{x}_j^l \circ W^l(\mathbf{r}_i - \mathbf{r}_j),$$

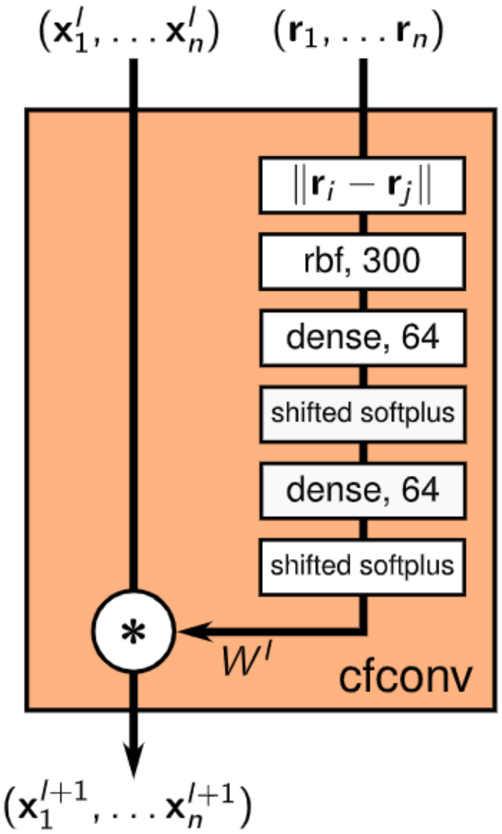
x^l: node embeddings at l layer

r: atomic coordinates

invariant GNNs: SchNet

- **RBF:** $e_k(\mathbf{r}_i - \mathbf{r}_j) = \exp(-\gamma \|d_{ij} - \mu_k\|^2)$
- **Dense:** MLPs
- **Softplus:** activation functions



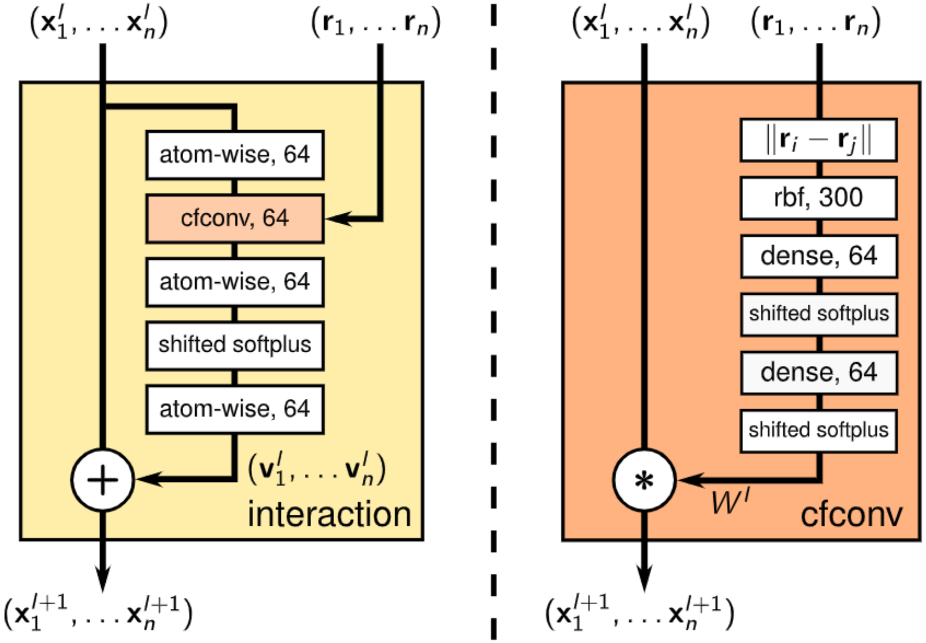

$(\mathbf{x}_1^l, \ldots \mathbf{x}_n^l)$
$(\mathbf{r}_1, \ldots \mathbf{r}_n)$
$\|\mathbf{r}_i - \mathbf{r}_j\|$
rbf, 300
dense, 64
shifted softplus
dense, 64
shifted softplus
*
W^l
cfconv
$(\mathbf{x}_1^{l+1}, \ldots \mathbf{x}_n^{l+1})$

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

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

invariant GNNs: SchNet

- cfconv module: aggregate atomic pairwise message passings
- Aggregated messages are then used for updating node embeddings through interaction module
  - Atom-wise layers
    Typical feedforward MLPs
    $$\mathbf{x}_i^{l+1} = W^l \mathbf{x}_i^l + \mathbf{b}^l$$
  - Residual connections
    $$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \mathbf{v}_i^l$$

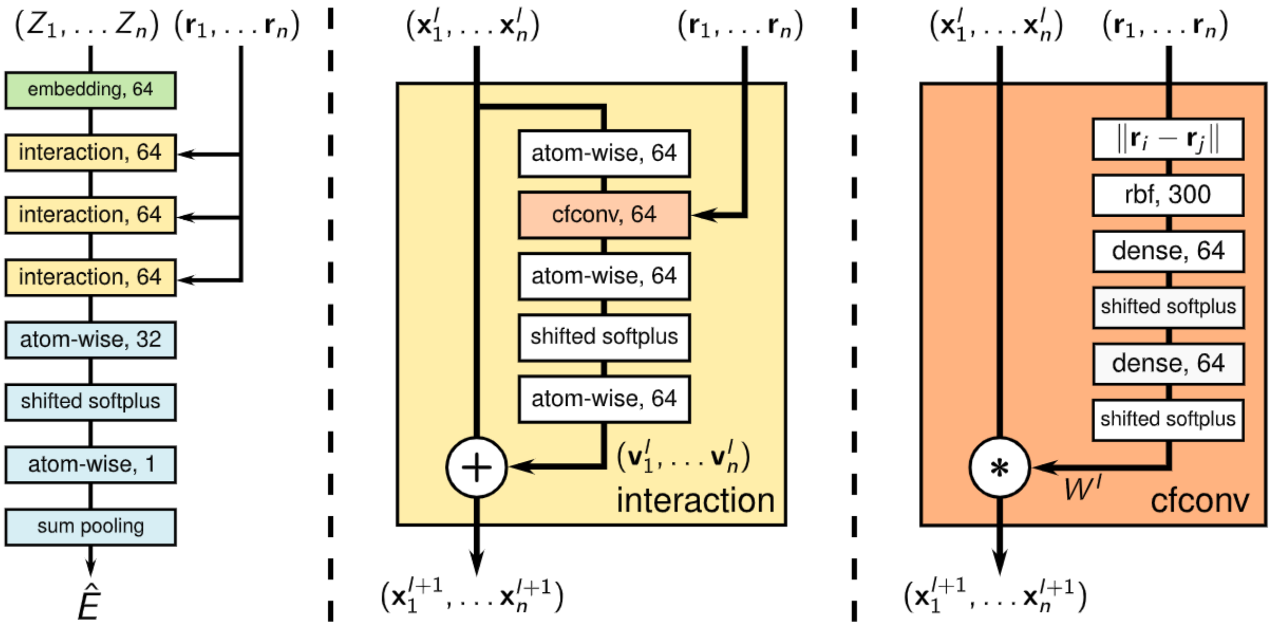


$(\mathbf{x}_1^l, \dots \mathbf{x}_n^l)$
$(\mathbf{r}_1, \dots \mathbf{r}_n)$
atom-wise, 64
cfconv, 64
atom-wise, 64
shifted softplus
atom-wise, 64
$(\mathbf{v}_1^l, \dots \mathbf{v}_n^l)$
interaction
$(\mathbf{x}_1^{l+1}, \dots \mathbf{x}_n^{l+1})$
$(\mathbf{x}_1^l, \dots \mathbf{x}_n^l)$
$(\mathbf{r}_1, \dots \mathbf{r}_n)$
$\|\mathbf{r}_i - \mathbf{r}_j\|$
rbf, 300
dense, 64
shifted softplus
dense, 64
shifted softplus
W^l
cfconv
$(\mathbf{x}_1^{l+1}, \dots \mathbf{x}_n^{l+1})$

W: weights in NNs
b: bias in NNs
v: update of x

invariant GNNs: SchNet

- Stack multiple interaction and atom-wise layers
- Predict single scalar value for each atom
- Sum all scalars together as energy prediction

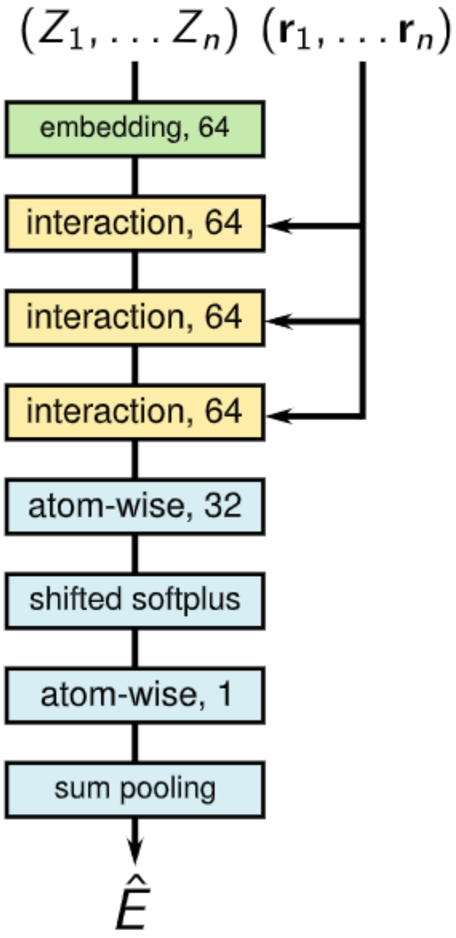

$(Z_1, \ldots Z_n)$ $(\mathbf{r}_1, \ldots \mathbf{r}_n)$
embedding, 64
interaction, 64
interaction, 64
interaction, 64
atom-wise, 32
shifted softplus
atom-wise, 1
sum pooling
$\hat{E}$
$(\mathbf{x}_1^l, \ldots \mathbf{x}_n^l)$
$(\mathbf{r}_1, \ldots \mathbf{r}_n)$
atom-wise, 64
cfconv, 64
atom-wise, 64
shifted softplus
atom-wise, 64
$(\mathbf{v}_1^l, \ldots \mathbf{v}_n^l)$
interaction
$(\mathbf{x}_1^{l+1}, \ldots \mathbf{x}_n^{l+1})$
$(\mathbf{x}_1^l, \ldots \mathbf{x}_n^l)$
$(\mathbf{r}_1, \ldots \mathbf{r}_n)$
$\|\mathbf{r}_i - \mathbf{r}_j\|$
rbf, 300
dense, 64
shifted softplus
dense, 64
shifted softplus
W^l
cfconv
$(\mathbf{x}_1^{l+1}, \ldots \mathbf{x}_n^{l+1})$

invariant GNNs: SchNet

- Forces can be computed by calculating gradient of the energy output w.r.t coordinates input

- Training objective (least square errors between predicted energy E and forces F and ground truth)

$$\ell(\hat{E}, (E, \mathbf{F}_1, \ldots, \mathbf{F}_n)) = \rho \|E - \hat{E}\|^2 + \frac{1}{n} \sum_{i=0}^{n} \left\| \mathbf{F}_i - \left(-\frac{\partial \hat{E}}{\partial \mathbf{R}_i} \right) \right\|^2$$

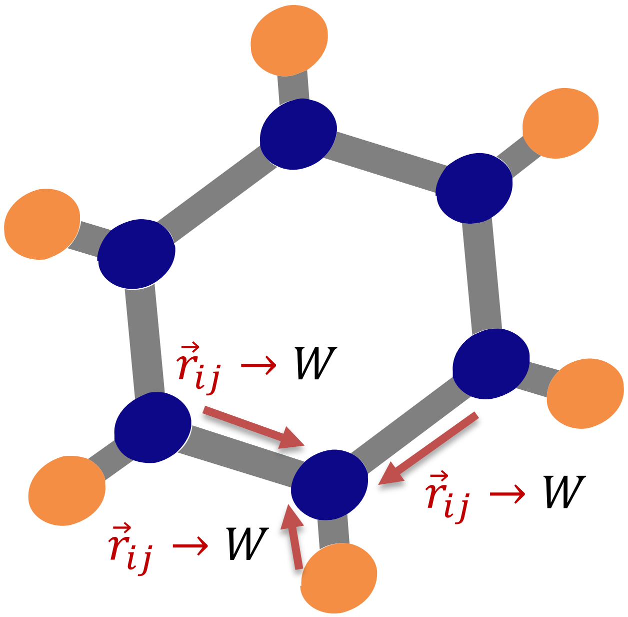


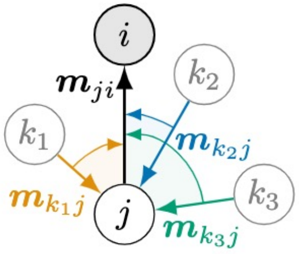
Improved SchNet: DimeNet

- Chemically, potential energy can be modeled as sum of four parts

$$E = E_{\text{bonds}} + E_{\text{angle}} + E_{\text{torsion}} + E_{\text{non-bonded}},$$

  - SchNet only depends on atom types and pairwise distance, ignore many information like angles and torsions
- **DimeNet** resolves this problem by
  - Do message interaction based on
    ① distance between atoms
    ② angle between bonds
    (both of which are invariant to translation and rotation!)

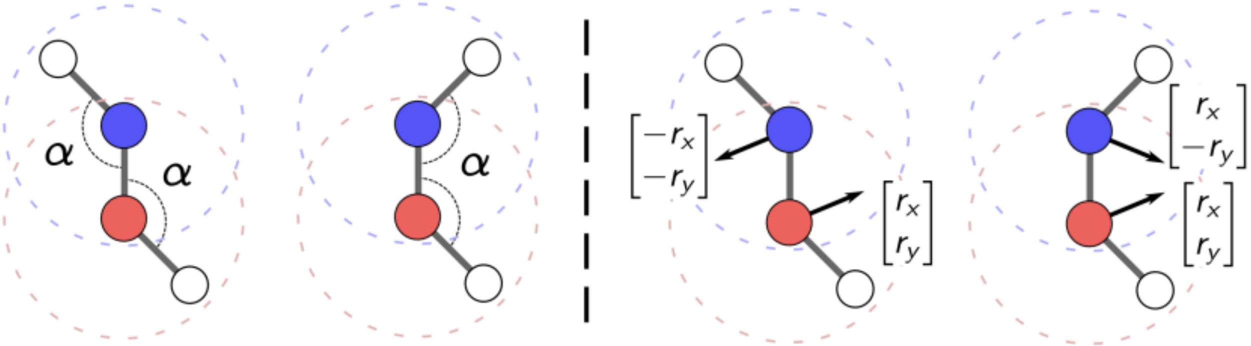




Gasteiger, Johannes, Janek Groß, and Stephan Günnemann. "Directional message passing for molecular graphs." *ICLR* (2020).

Expressiveness

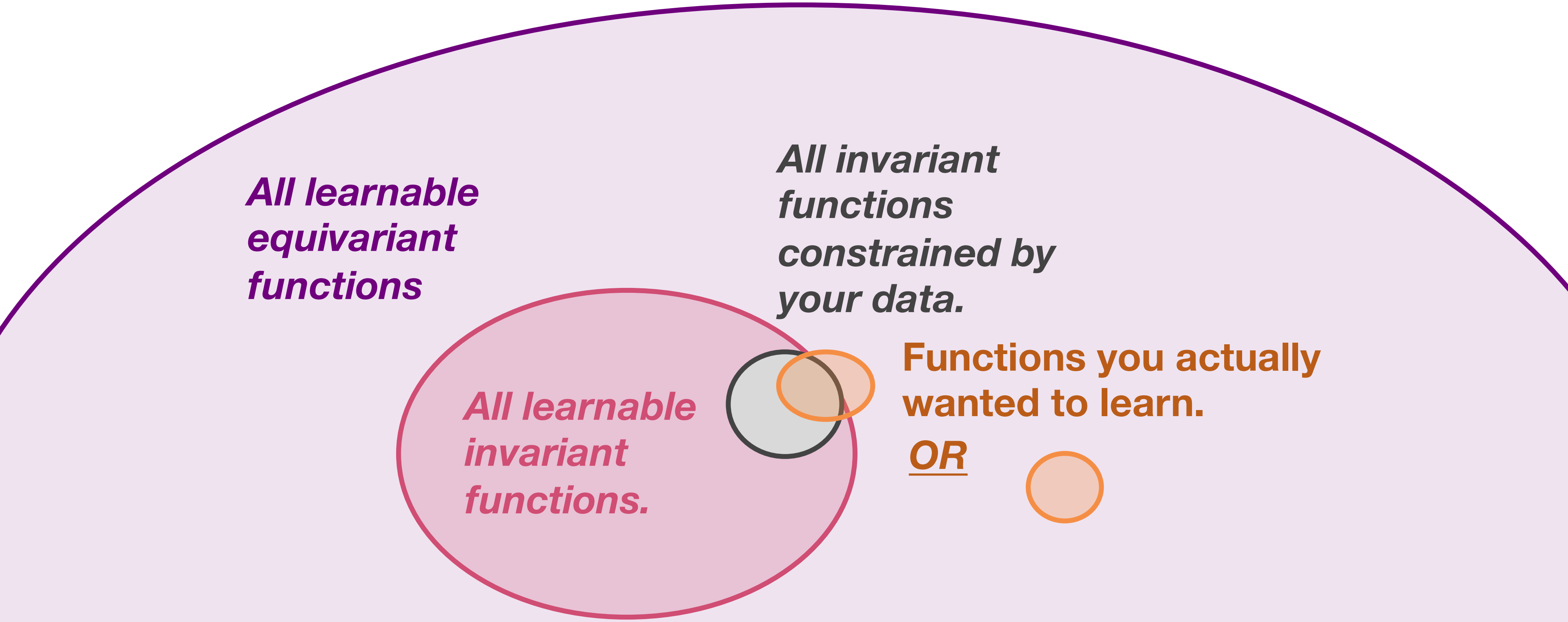
- **Distances/Angles** are incomplete descriptors for uniquely identifying geometric structure.


α
α
α
$\begin{bmatrix}-r_x\\-r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\-r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\r_y\end{bmatrix}$

- This pair of geometric graphs cannot be distinguished by **identical** scalar quantities.
- But they can be distinguished based on **directional** or **geometric** information

Limitations of invariant GNNs

- Why not limit yourself to invariant functions?
- You have to guarantee that your input features already contain any necessary equivariant interactions.



All learnable equivariant functions

All invariant functions constrained by your data.

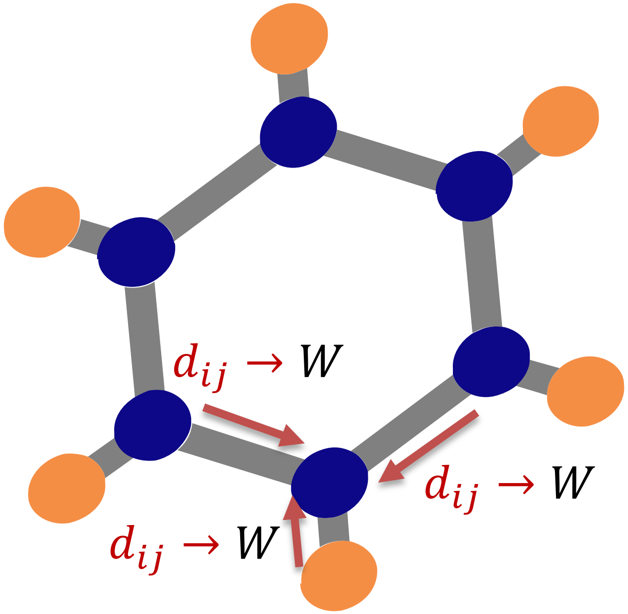
All learnable invariant functions.

Functions you actually wanted to learn.

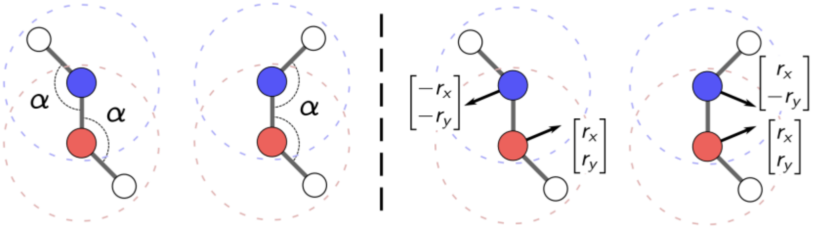
OR

Equivariant GNNs: PaiNN

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



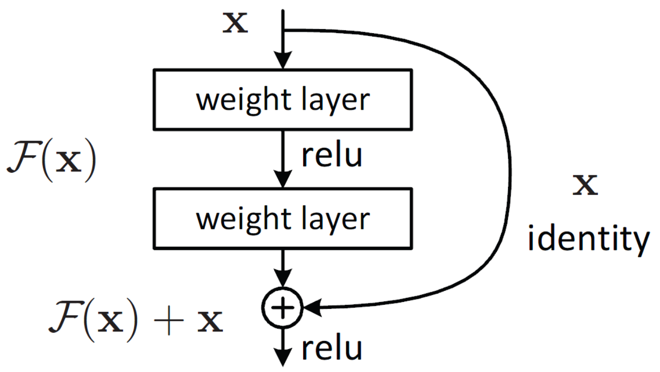
- However, differently, in PaiNN each node has two features (both scalar features s_i and vector features v_i)


α
α
α
$\begin{bmatrix}-r_x\\-r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\-r_y\end{bmatrix}$
$\begin{bmatrix}r_x\\r_y\end{bmatrix}$

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.

Equivariant GNNs: PaiNN

- The two features (scalar features s_i and vector features v_i) are
  - initialed by: atom embeddings and 0 tensors
  - updated by: residual updates



$s, \Delta s$: scalar features and its updates

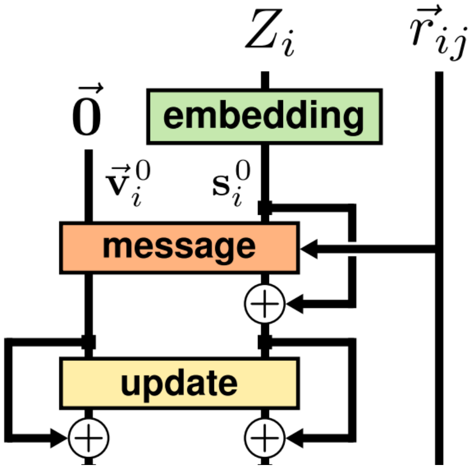
$v, \Delta v$: tensor features and its updates

ϕ, W: networks

$$s_i = s_i + \Delta s_i$$
$$v_i = v_i + \Delta v_i$$

$$\begin{aligned}\Delta \mathbf{s}_i^m &= (\phi_s(\mathbf{s}) * \mathcal{W}_s)_i \\ &= \sum_j \phi_s(\mathbf{s}_j) \circ \mathcal{W}_s(\|\vec{r}_{ij}\|)\end{aligned}$$

$$\begin{aligned}\Delta \vec{\mathbf{v}}_i^m = &\sum_j \vec{\mathbf{v}}_j \circ \phi_{vv}(\mathbf{s}_j) \circ \mathcal{W}_{vv}(\|\vec{r}_{ij}\|) \\ &+ \sum_j \phi_{vs}(\mathbf{s}_j) \circ \mathcal{W}'_{vs}(\|\vec{r}_{ij}\|) \frac{\vec{r}_{ij}}{\|\vec{r}_{ij}\|}\end{aligned}$$

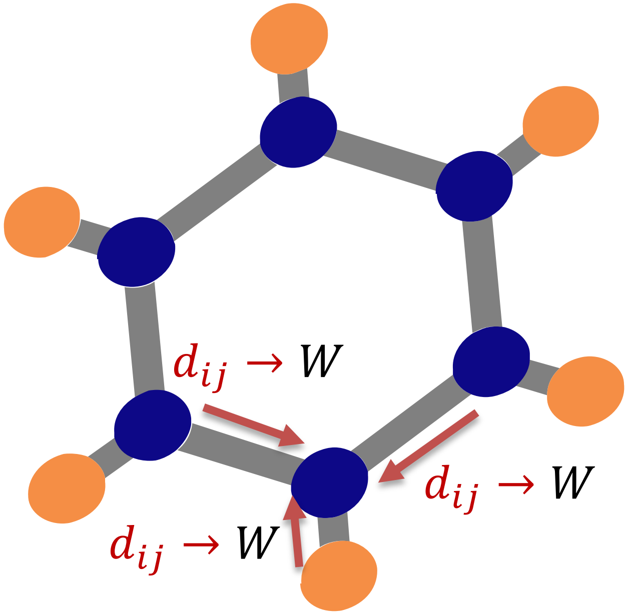


Equivariant GNNs: PaiNN

- scalar features s_i update for atom i:

$$\begin{aligned}\Delta \mathbf{s}_i^m &= (\boldsymbol{\phi}_s(\mathbf{s}) * \mathcal{W}_s)_i \\ &= \sum_j \boldsymbol{\phi}_s(\mathbf{s}_j) \circ \mathcal{W}_s(\|\vec{r}_{ij}\|)\end{aligned}$$

  - ϕ_s, W_s are neural networks



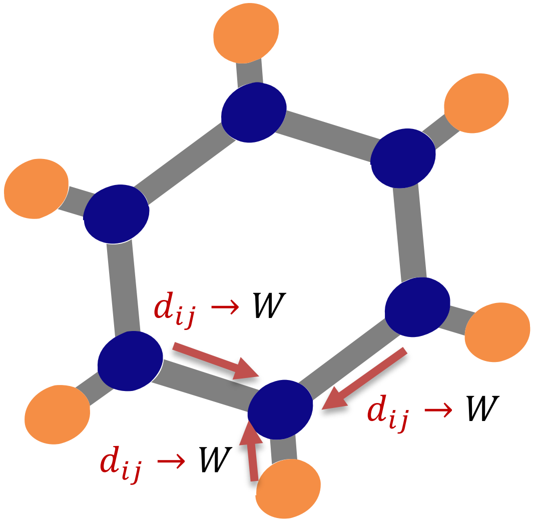
- Similar to SchNet
  - invariant weights W_s by $\|\vec{r}_{ij}\|$
  - => invariant messages ϕ_s
  - => passing invariant messages $\phi_s \cdot W_s$
  - => invariant sum Agg over messages

Equivariant GNNs: PaiNN

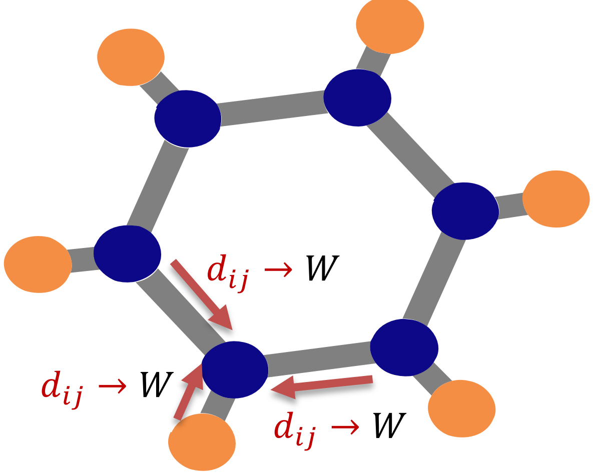
- vector features v_i update:

$$\Delta\vec{\mathbf{v}}_i^m = \sum_j \vec{\mathbf{v}}_j \circ \phi_{vv}(\mathbf{s}_j) \circ \mathcal{W}_{vv}(\|\vec{r}_{ij}\|) + \sum_j \phi_{vs}(\mathbf{s}_j) \circ \mathcal{W}'_{vs}(\|\vec{r}_{ij}\|) \frac{\vec{r}_{ij}}{\|\vec{r}_{ij}\|}$$

  - ϕ, W are all neural networks

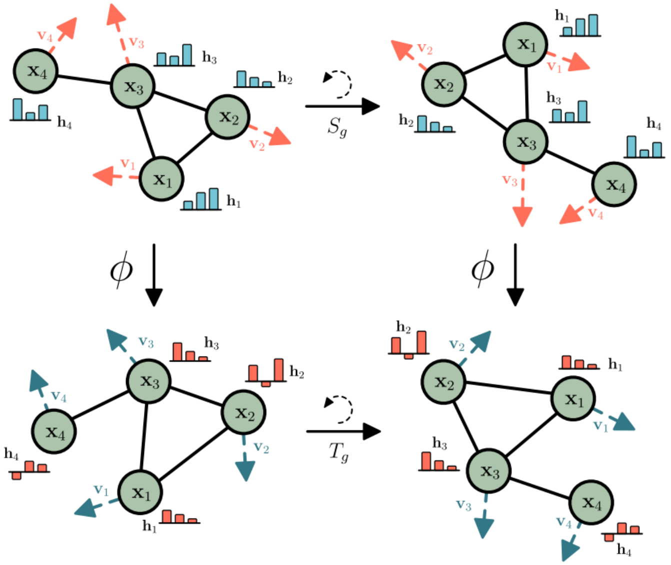


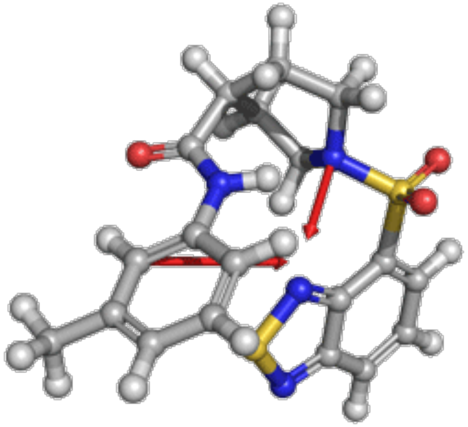
- Different to SchNet
  - invariant weights W_s by $\|\vec{r}_{ij}\|$
  - => invariant messages ϕ_s
  - => passing tensor messages $\phi_s \cdot W_s \cdot \vec{r}_{ij}$
  - => weighted sum of relative directions $\vec{r}_{ij}$, thus keeping the equivariant properties!



Equivariant GNNs: PaiNN

- By stacking multiple PaiNN layers…
- vector features v_i after final layer are tensor features equivariant w.r.t to input coordinates
- are therefore can be directly used as force prediction





Satorras, Vıctor Garcia, Emiel Hoogeboom, and Max Welling. "E(n) equivariant graph neural networks." *International conference on machine learning*. PMLR, 2021.

Summary of Geometric GNNs

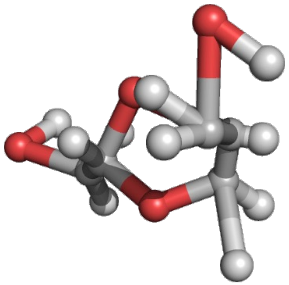
- Geometric GNNs need to capture sufficient information of geometries
- SchNet (and DimeNet) achieve invariance by only learning over scalarized invariant features (distances, angles, …)
- PaiNN designs both scalar and tensor features, where the tensor features are equivariant with input coordinates
- Applications: equivariant output can be used as force prediction for molecular simulation

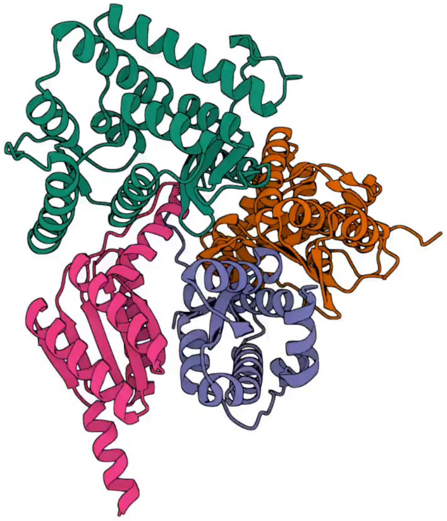
Outline

- Geometric Graphs
- Geometric Graph NNs
    - Invariant GNNs
    - Equivariant GNNs
- **Geometric Generative Models**
    - Geometric Diffusion Models

Broad Applications

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

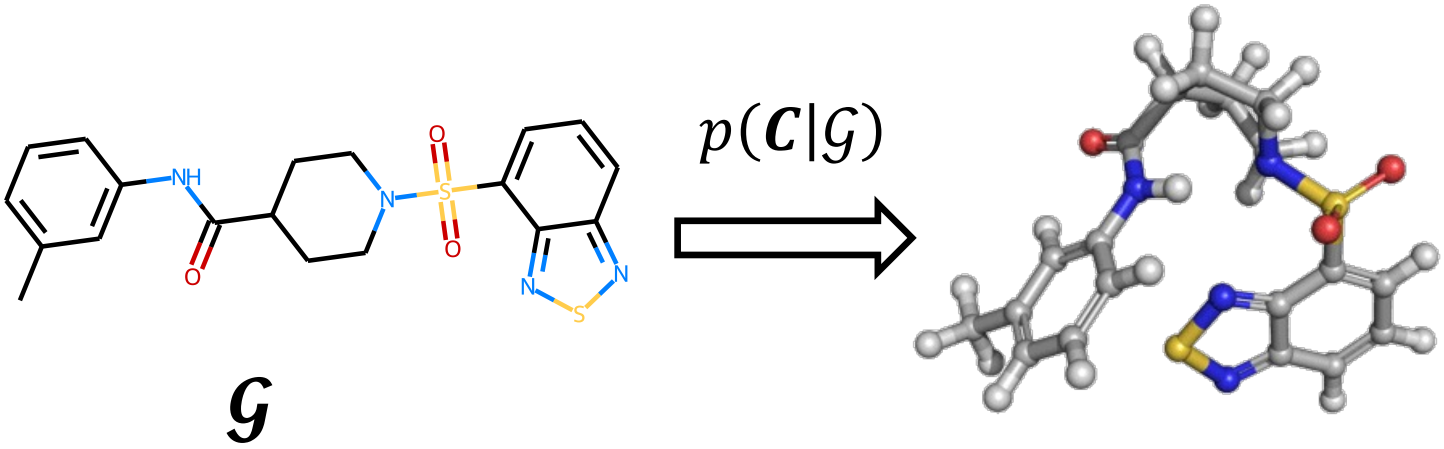




https://generatebiomedicines.com/chroma

Molecular Conformation Generation

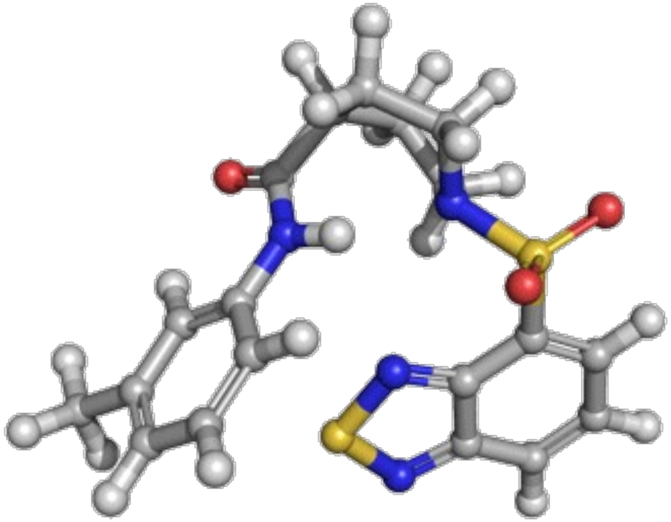
- Generate stable conformations from molecular graph
  - Molecular graph $\mathcal{G}$: 2D atom-bond graph
  - Conformation $\boldsymbol{C}$: atomic 3D coordinates
  - One molecule can have multiple possible conformations, which follows a distribution conditioned on temperature T



$$\boldsymbol{C} \propto \underbrace{\exp(-E(\boldsymbol{C})/T)}_{\textbf{Boltzmann distribution}}$$

Challenges

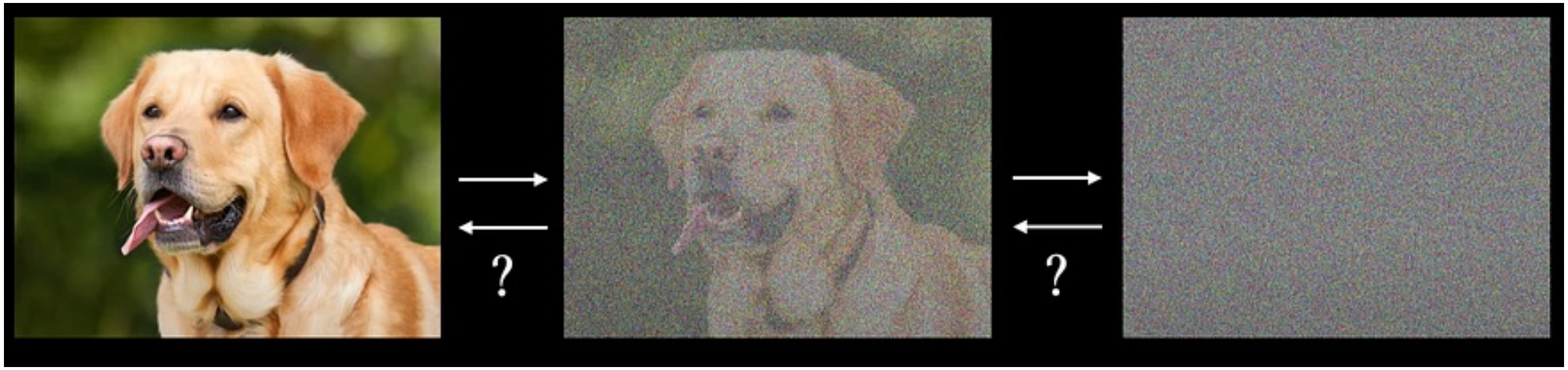
- Generative models learns the data distribution
- Similar to the learning algorithm, generation process should also capture the physical symmetry groups, i.e., **equivariant to roto-translation**

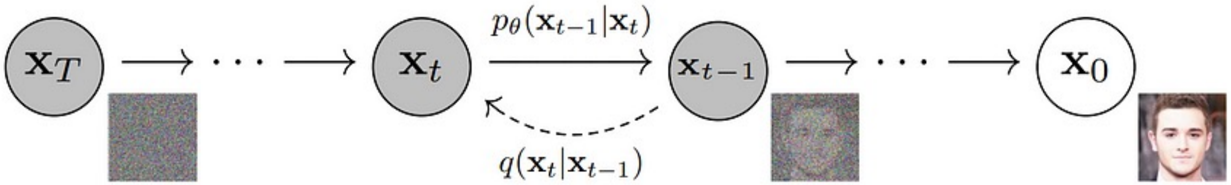


$$\boldsymbol{C} \propto \underbrace{\exp(-E(\boldsymbol{C})/T)}_{\textbf{Boltzmann distribution}}$$

Background: Diffusion Models

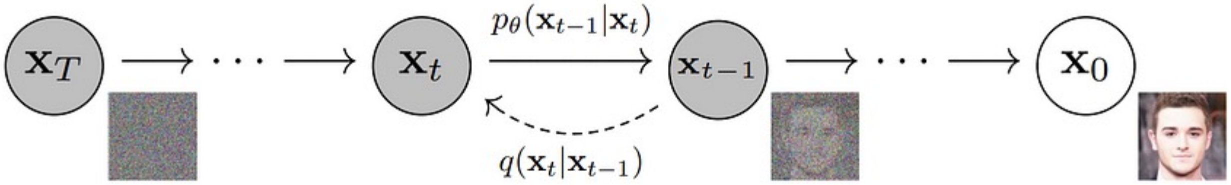
- Define forward diffusion process to destroy data into different noisy-level samples
- Learn reverse models to generate by denoising




$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$

Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models. " *Advances in Neural Information Processing Systems* 33 (2020): 6840-6851.

Background: Diffusion Models


$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$

- Training:
  - Sample random noise ϵ
  - Destroy the data by $x_t = \mu_t x + \sigma_t \epsilon$ at every t
    - μ_t, σ_t , t are pre defined
  - Learn models $f_\theta(x_t, t)$ to predict the noise ϵ
- Sampling:
  - Sample $x_T \sim N(0, I)$ from Gaussian random noise
  - Generate x by repeatedly predicting and subtracting the noise
  - Recover clean data

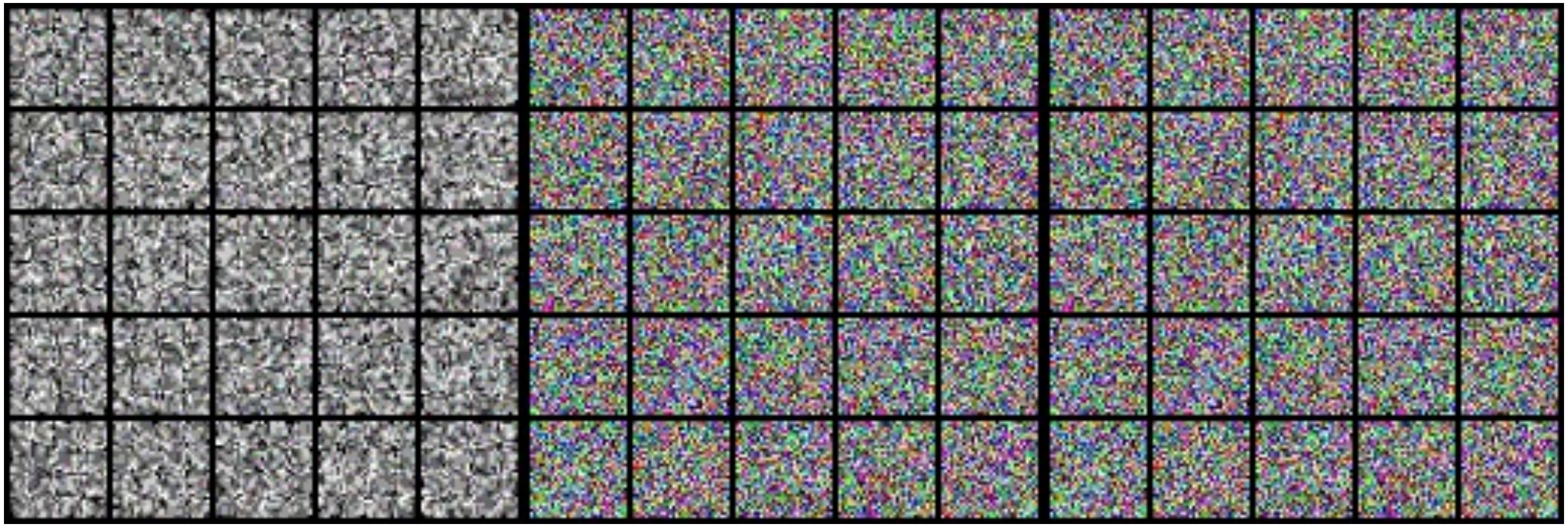
t: timestep
μ: means to shrink data
σ: variance as noise level

higher t -> smaller μ and larger σ

$\mu_0 = 1, \sigma_0 = 0$
$\mu_1 = 0, \sigma_1 = 1$

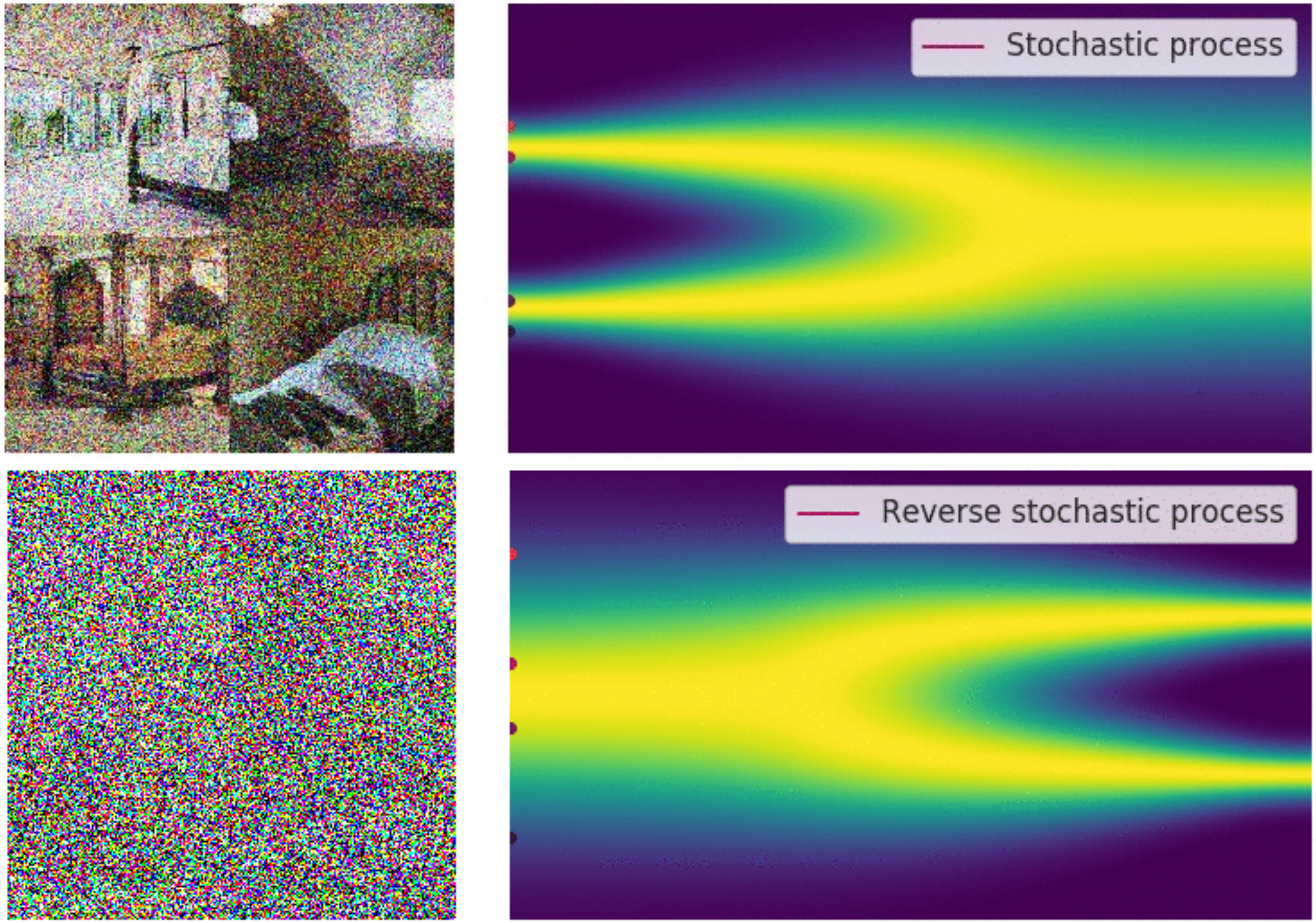
Background: Diffusion Models

- Then the learned reverse model can be used to generate data by progressively denoising



Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." *Advances in neural information processing systems* 32 (2019).

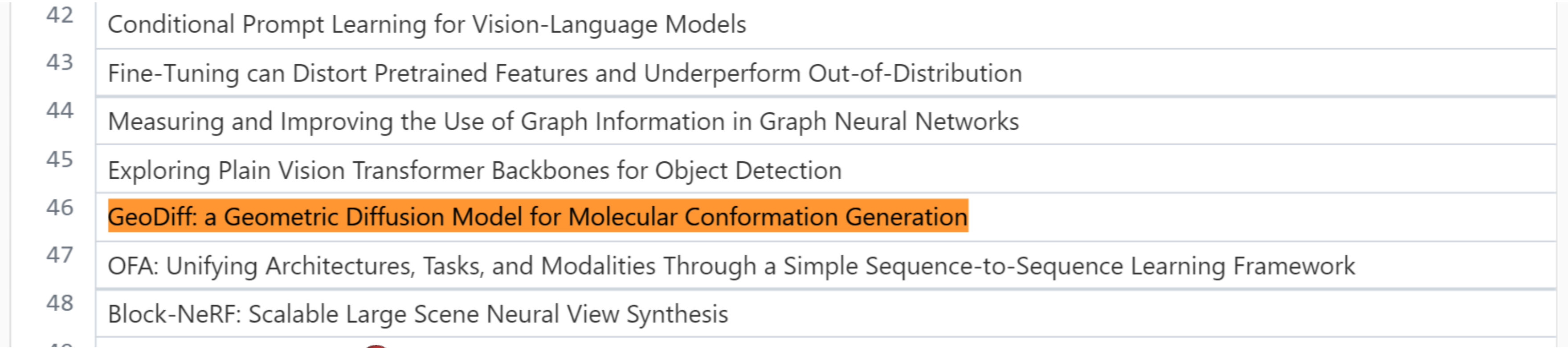
Background: Diffusion Models



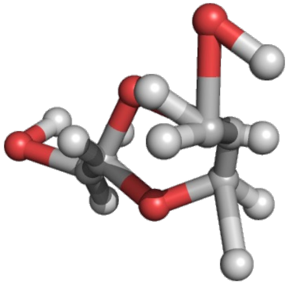
Song, Yang, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. "Score-based generative modeling through stochastic differential equations." *ICLR* (2021) Best Paper

Geometric Diffusion

- We bring the idea into molecule generation!
- *Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. "GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation." In International Conference on Learning Representations. 2021.*
- Top 50 most cited AI papers in 2022



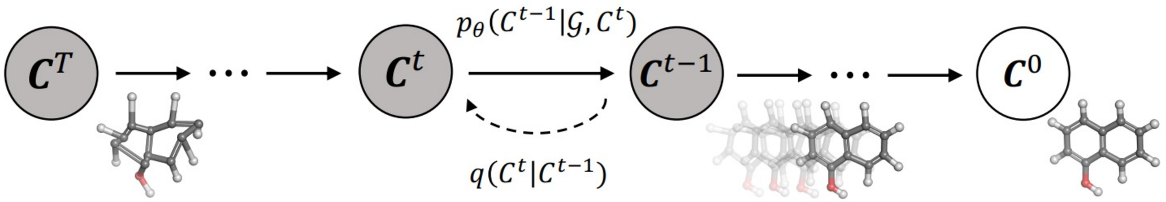
42	Conditional Prompt Learning for Vision-Language Models
43	Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution
44	Measuring and Improving the Use of Graph Information in Graph Neural Networks
45	Exploring Plain Vision Transformer Backbones for Object Detection
46	GeoDiff: a Geometric Diffusion Model for Molecular Conformation Generation
47	OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework
48	Block-NeRF: Scalable Large Scene Neural View Synthesis



https://www.zeta-alpha.com/post/must-read-the-100-most-cited-ai-papers-in-2022

Geometric Diffusion

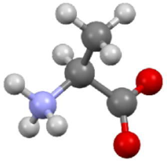
- GeoDiff (Geometric Diffusion)

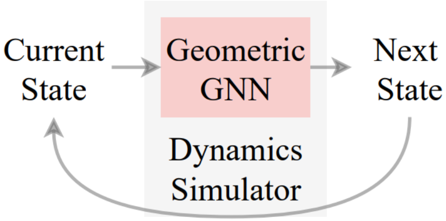


- Diffusion process gradually perturb the molecular geometry until the conformation is destroyed.
- Symmetrically, we want to learn the reverse generative process to progressively refined a random noisy geometry

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. "GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation." In *International Conference on Learning Representations*. 2021.

Geometric Diffusion



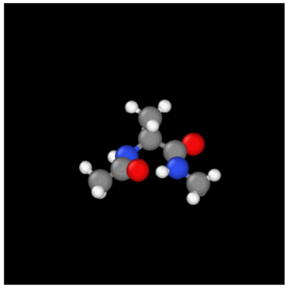


Current State

Geometric GNN

Next State

Dynamics Simulator



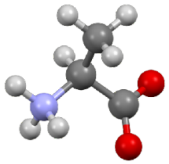
- Diffusion process gradually perturb the molecular geometry until the conformation is destroyed.
- Symmetrically, we want to learn the reverse generative process to progressively refined a random noisy geometry

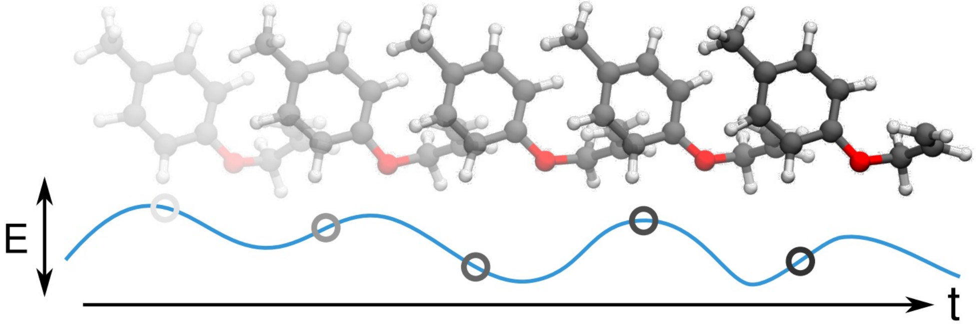
- Wait! This is kind of similar to molecular simulation!

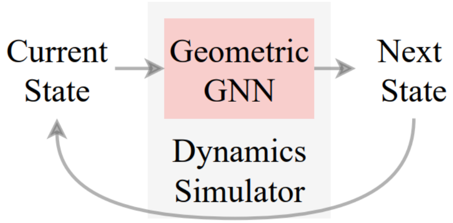
Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. "GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation." In *International Conference on Learning Representations*. 2021.

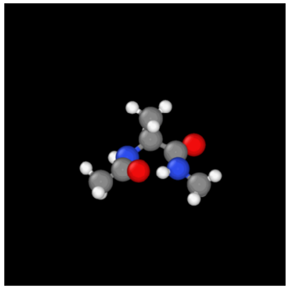
Geometric Diffusion

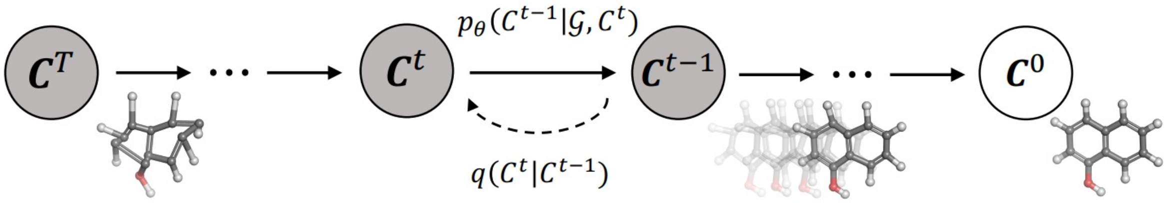
- To optimize structures to better states:
  - Simulation: learns to predict force
  - Diffusion models: learns to predict noise







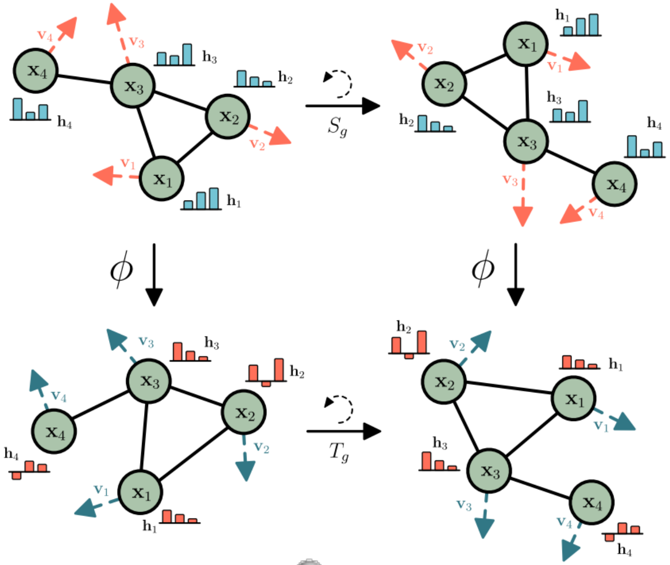




Geometric Diffusion

- Insight!
  Similar to force, denoising direction should be equivariant with the molecular coordinates!

- Solution:
  Parameterizing the denoising network with equivariant GNN ☺



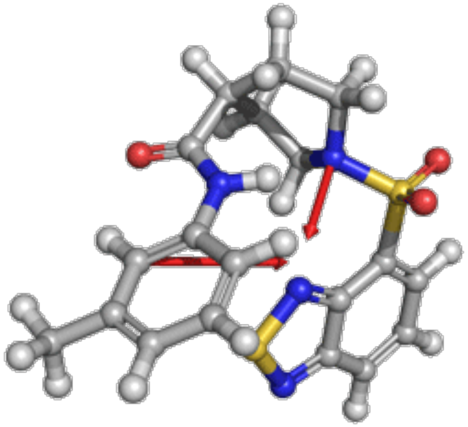
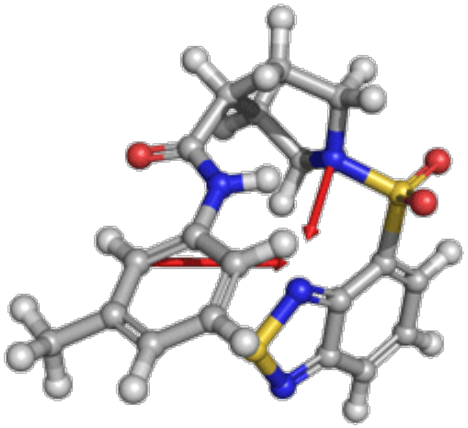


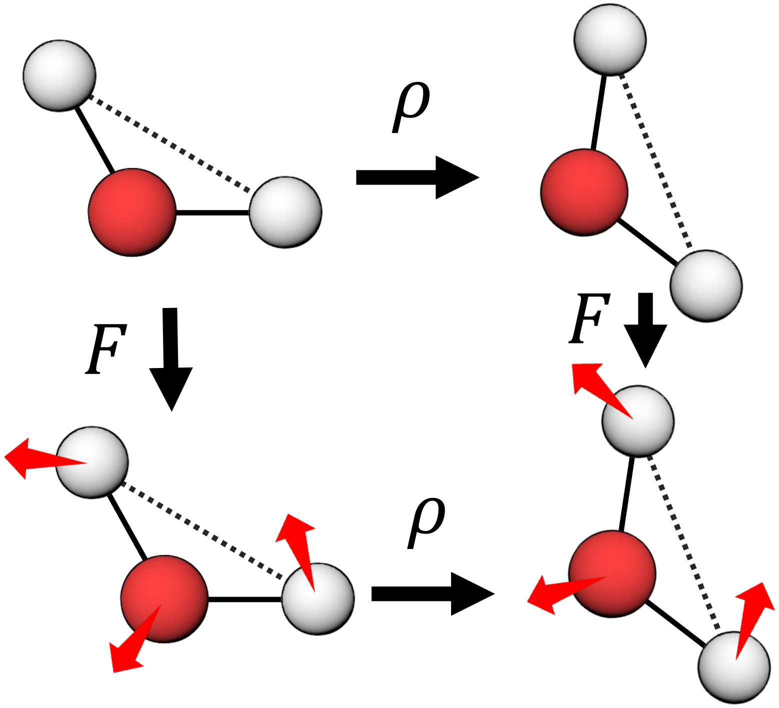
Illustration: 3D Rotation Equivariance

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

The equation says that applying the ρ on the input has the same effect as applying it to the output.



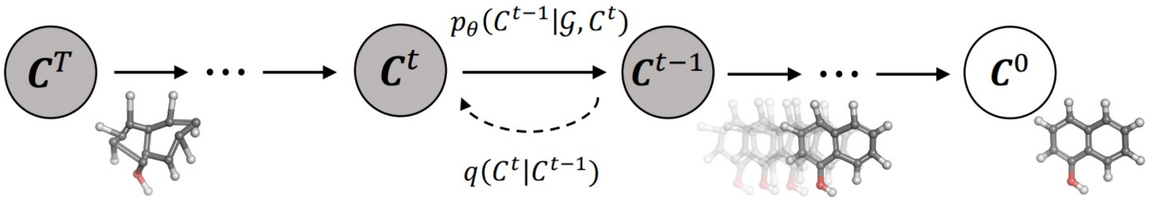
A GIF illustrating the rotation equivariance of atomic forces.
Two red arrows stand for forces acting on atoms, which rotate together with the molecule.

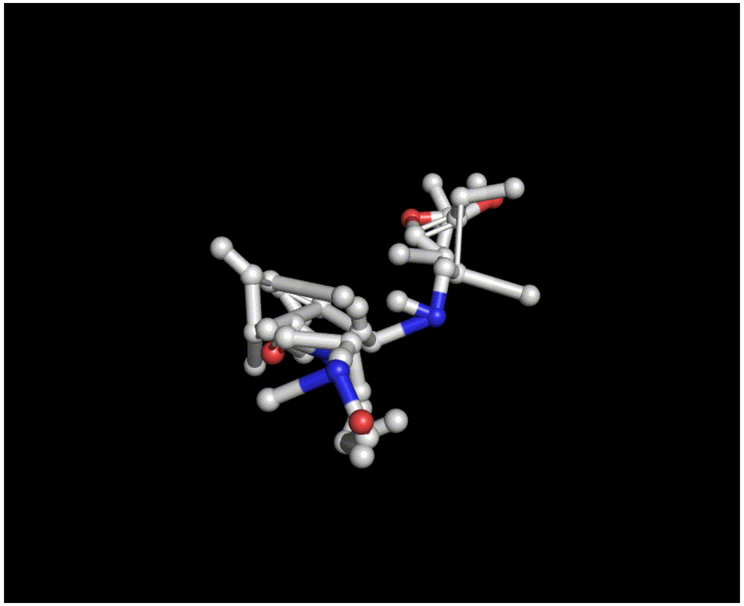


Visual explanation of the equivariance

GeoDiff: Sampling

- Sampling by an equivariant denoising procedure:

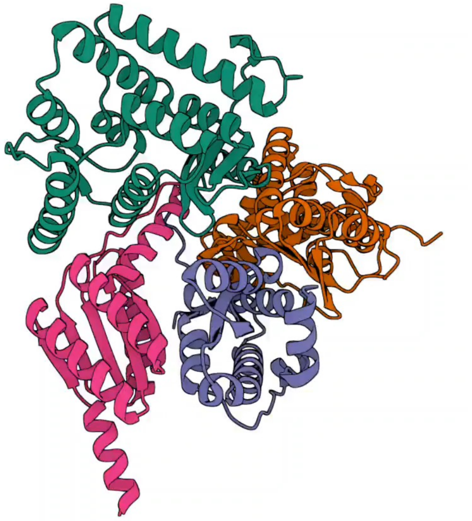


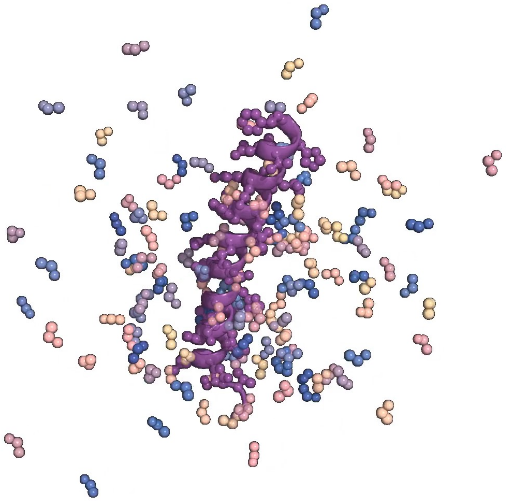


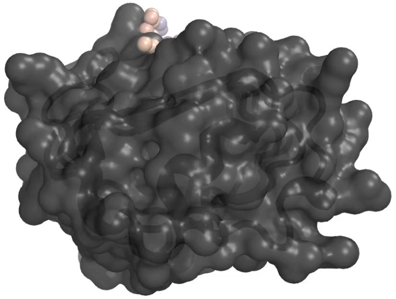
Most Recent Progress

- *Illuminating protein space with a programmable generative model*
  John Ingraham, Max Baranov, Zak Costello, Vincent Frappier, et al

- *Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models*
  Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, et al







Summary of Geometric Generative Models

- Geometric generative models should also capture the roto-translational symmetries
- GeoDiff build the connection between molecular simulations and molecular diffusion generations
- GeoDiff learns equivariant GNNs to conduct an equivariant denoising diffusion process
- Broad impact for real world drug discovery challenges (drug design, protein design, drug-protein interaction)

Summary

- Geometric Graphs
- Geometric Graph NNs
  - Invariant GNNs
  - Equivariant GNNs
- Geometric Generative Models
  - Geometric Diffusion Models

Future Directions

- More expressive geometric GNNs
- More principled geometric generative models
- More impactful downstream applications

Acknowledgement

- Jure Leskovec (Stanford) https://cs.stanford.edu/~jure/
- Tess Smidt (MIT) https://blondegeek.github.io/
- Jian Tang (MILA) https://jian-tang.com/
- Yang Song (Caltech) http://yang-song.github.io/
- Stefano Ermon (Stanford) https://cs.stanford.edu/~ermon/
- Ron Dror (Stanford) https://cs.stanford.edu/people/rondror/
- Chaitanya Joshi (Cambridge) https://www.chaitjo.com/

Thank you!

And any question?