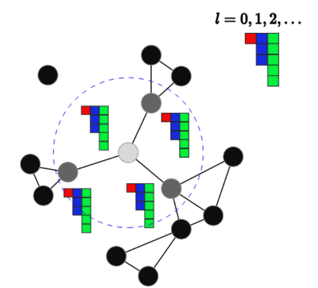
Equivariant neural networks - Tutorial Introduction

today

Graph Neural Networks

- Nodes - atoms
- Edges - bonds between atoms in cutoff
- Layer - In this presentation layer is number of interactions



$$\mathbf{m}_i^{t+1} = \sum_{j \in \mathcal{N}(i)} M_t\left(\mathbf{h}_i^t, \mathbf{h}_j^t, \mathbf{e}_{ij}\right)$$

$$\mathbf{h}_i^{t+1} = U_t\left(\mathbf{h}_i^t, \mathbf{m}_i^{t+1}\right)$$

S. Batzner, E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials, Nature Communications 13(2453), 2022

A. Musaelian, Learning local equivariant representations for large-scale atomistic dynamics, Nat Commun 14, 579 (2023).

Invariant vs Equivariant

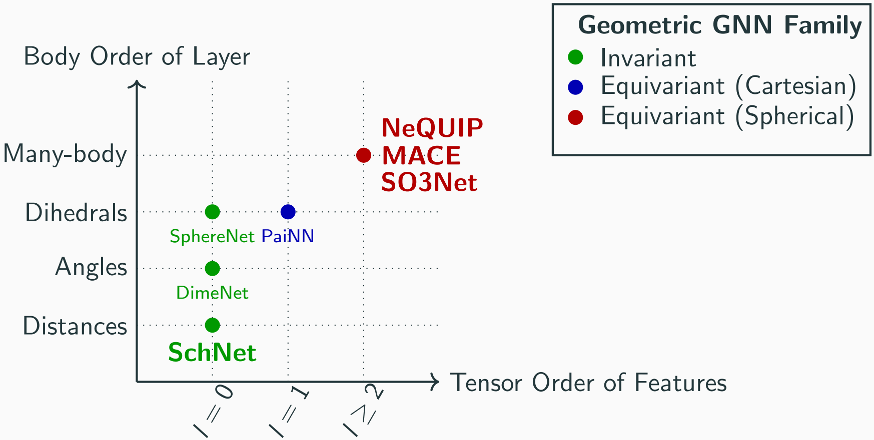
Invariant

$$f(D_X[g]x) = f(x)$$

Equivariant

$$f(D_X[g]x) = D_Y[g]f(x)$$

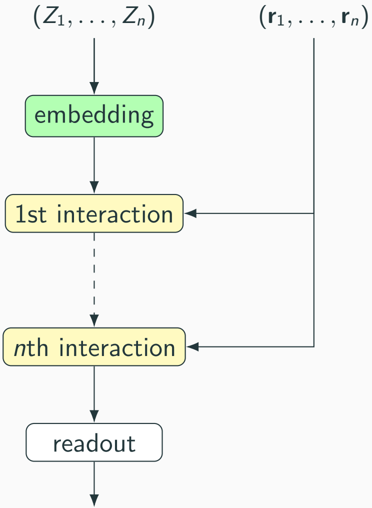
Invariant vs Equivariant

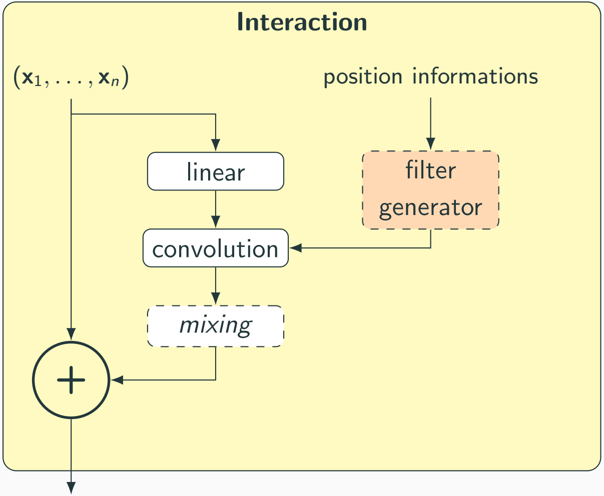


A. Duval, A Hitchhiker's Guide to Geometric GNNs for 3D Atomic Systems, arxive 2023

Model schemes

Do not search to one-to-one scheme in the literature





Convolution filter

Invariant

$$S(\mathbf{r}) = R(r_{ij})$$

Equivariant

$$S^l_m(\mathbf{r}) = R(r_{ij})Y^l_m(\hat{\mathbf{r}}_{ij})$$

Radial function $R(r_{ij})$

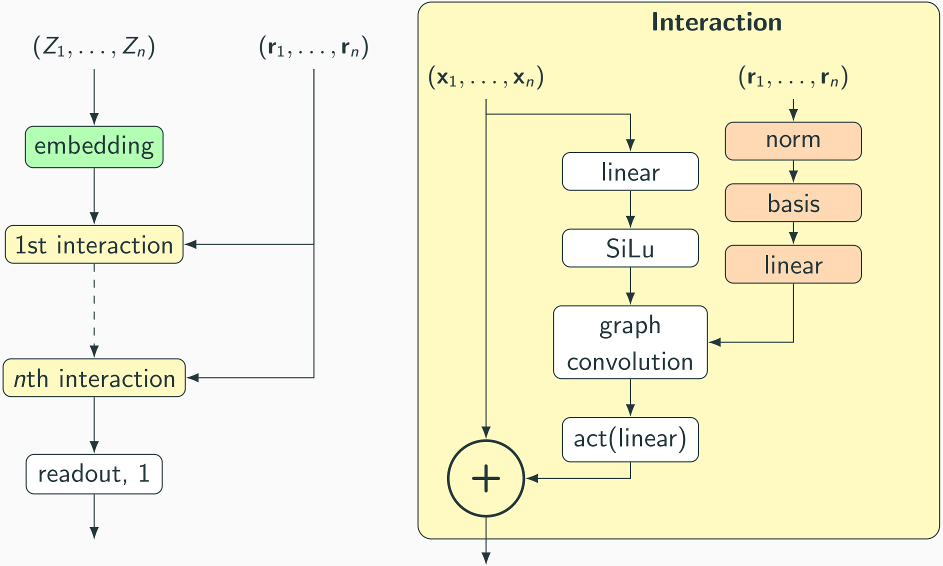
- MLP of radial functions j

$$R(r_{ij}) = \mathrm{MLP}\left(\{j\}\right)$$

E.g. MACE-MP0 has 10 Bessel functions and MLP `[10, 64, 64, 64, ···]`

From invariant...

SchNet scheme



SchNet

Atom embeddings

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

SchNet

Atom embeddings

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

Atom-wise layers (a.k.a. linear or self-interaction)

$$\mathbf{y}_i^l = \mathbf{W}^l \mathbf{x}_i^l + \mathbf{b}^l$$

SchNet

Atom embeddings

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

Atom-wise layers (a.k.a. linear or self-interaction)

$$\mathbf{y}_i^l = \mathbf{W}^l \mathbf{x}_i^l + \mathbf{b}^l$$

Interaction blocks

- atom i get informations from j atoms in the cutoff radius
- Propagation of a scalar information
- 2-body, only dependes on radial functions
- cfconv = continuous-filter convolutional

SchNet

Atom embeddings

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

Atom-wise layers (a.k.a. linear or self-interaction)

$$\mathbf{y}_i^l = \mathbf{W}^l \mathbf{x}_i^l + \mathbf{b}^l$$

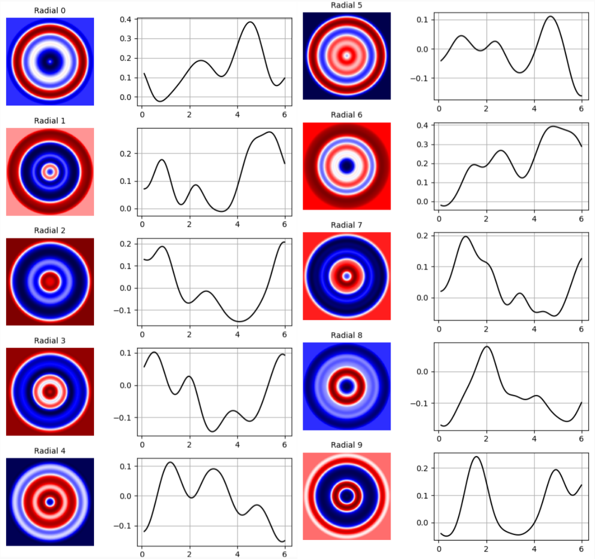
Interaction blocks

- Element-wise multiplication between scalar features and a filter $\mathbf{R}$

$$\mathbf{xc}_i^t = \sum_{j=0}^{n_{\text{atoms}}} \mathbf{x}_j^t \circ \mathbf{R}^t(\mathbf{r}_j - \mathbf{r}_i)$$

Readout for the atom-site energy is atom-wise mixing nn.linear to get 1

SchNet - filter generation



...to equivariant

Equivariant feature vectors

"The feature vectors are geometric objects that comprise a direct sum of irreducible representations of the O(3) symmetry group."

$$\underbrace{128\times0e}_{\text{scalar}} + \overbrace{128\times1o}^{\text{vector}} + \underbrace{128\times2e}_{\text{tensor}} + \cdots$$

- e (even) and o (odd) - parity
- e1 - pseudovectors (e.g. normal of the plane)

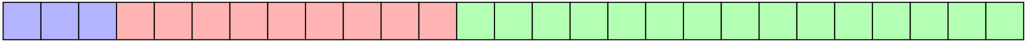
Symbols

- ℓ - order of features (max num after parity symbol) (in MACE L_{MAX})
- l - order of spherical harmonics in the filter (in MACE ℓ)

implemented via python `e3nn` library

Equivariant feature vectors

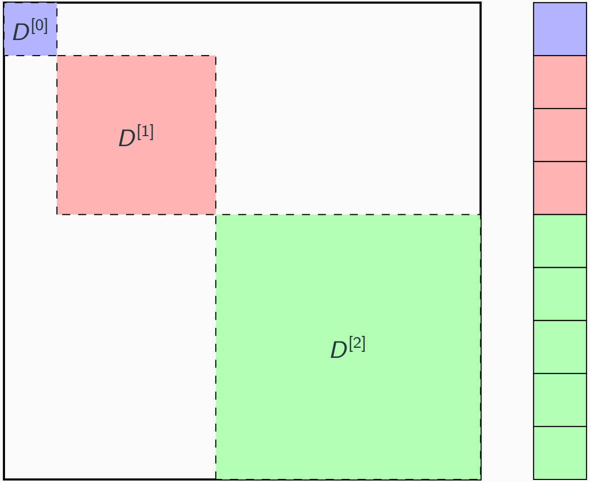
3×0e + 3×1o + 3×2e



$\ell = 0$ $\ell = 1$ $\ell = 2$

Equivariant feature vectors

Transformation of 1×0e +1×1o + 1×2e

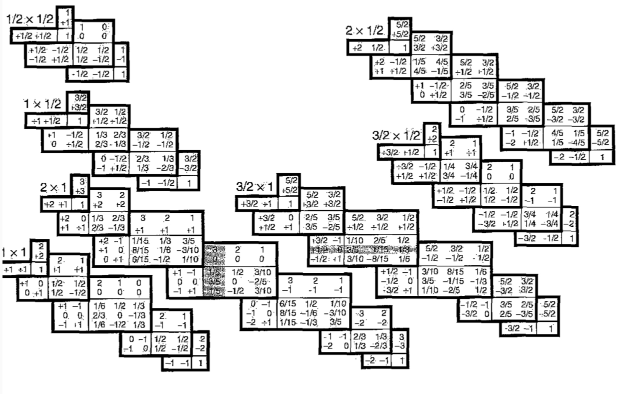


Tensor product

$$(\mathbf{x} \otimes \mathbf{y})_{\ell_{\text{out}}, m_{\text{out}}} = \sum_{m_1, m_2} \begin{pmatrix} \ell_1 & \ell_2 & \ell_{\text{out}} \\ m_1 & m_2 & m_{\text{out}} \end{pmatrix} \mathbf{x}_{\ell_1, m_1} \mathbf{y}_{\ell_2, m_2}$$

Tensor product

$$(\mathbf{x} \otimes \mathbf{y})_{\ell_{\text{out}}, m_{\text{out}}} = \sum_{m_1, m_2} \begin{pmatrix} \ell_1 & \ell_2 & \ell_{\text{out}} \\ m_1 & m_2 & m_{\text{out}} \end{pmatrix} \mathbf{x}_{\ell_1, m_1} \mathbf{y}_{\ell_2, m_2}$$



Tensor product

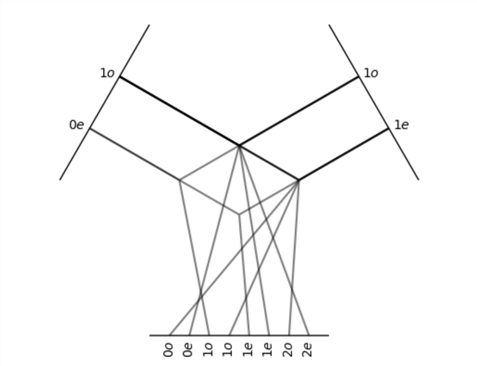
$$(\mathbf{x} \otimes \mathbf{y})_{\ell_{\mathrm{out}}, m_{\mathrm{out}}} = \sum_{m_1, m_2} \begin{pmatrix} \ell_1 & \ell_2 & \ell_{\mathrm{out}} \\ m_1 & m_2 & m_{\mathrm{out}} \end{pmatrix} \mathbf{x}_{\ell_1, m_1} \mathbf{y}_{\ell_2, m_2}$$

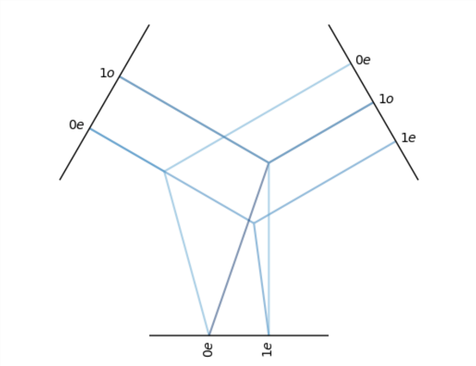
Output features: $|\ell_1 - \ell_2| \leq \ell_{\text{out}} \leq \ell_1 + \ell_2$

Tensor product

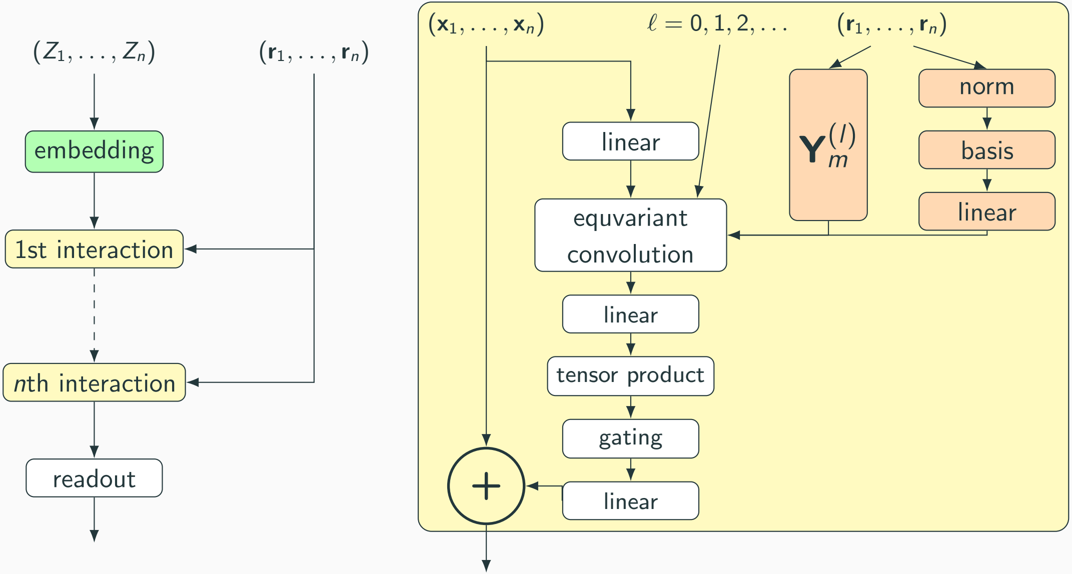
$$(\mathbf{x} \otimes \mathbf{y})_{\ell_{\text{out}}, m_{\text{out}}} = \sum_{m_1, m_2} \begin{pmatrix} \ell_1 & \ell_2 & \ell_{\text{out}} \\ m_1 & m_2 & m_{\text{out}} \end{pmatrix} \mathbf{x}_{\ell_1, m_1} \mathbf{y}_{\ell_2, m_2}$$

Output features: $|\ell_1 - \ell_2| \leq \ell_{\text{out}} \leq \ell_1 + \ell_2$

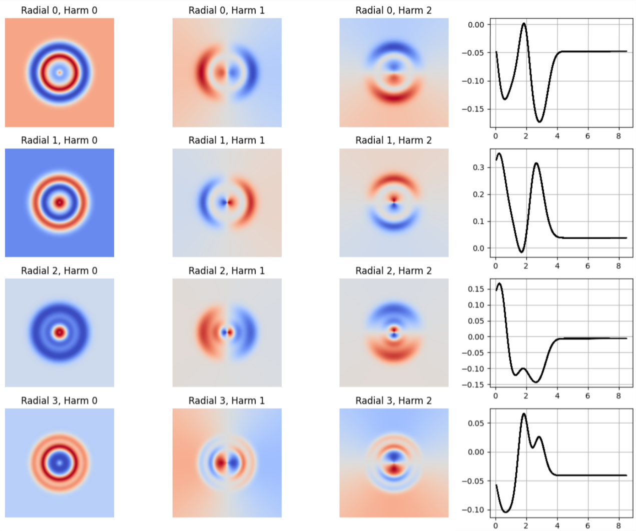




SO3Net scheme



SO3Net - filter generation



SO3Net

Atom embeddings

- Embedding is via scalars

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

- Has to be expanded with 0s non-scalars

SO3Net

Atom embeddings

- Embedding is via scalars

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

- Has to be expanded with 0s non-scalars

Atom-wise layers

$$\mathbf{y}^t_{i,(lm),f} = \sum_{ff'} \mathbf{W}_{ff'} \mathbf{h}^t_{i(lm)f'}$$

SO3 convolution (msg creation)

$$\mathbf{h}^t_{i(\ell m)f} = \sum_{j \in \mathcal{N}_{\mathrm{at}}[i]} \sum_{\ell_1 m_1} \sum_{l_2 m_2} \mathbf{x}^t_{j(\ell_1 m_1)f} \mathbf{R}^t_{l_2 f}(r_{ij}) \mathbf{Y}_{l_2, m_2}(\hat{\mathbf{r}}_{ij}) \mathbf{C}^{\ell m}_{\ell_1 m_1 l_2 m_2}$$

SO3Net

Atom embeddings

- Embedding is via scalars

$$\mathbf{x}_i^0 = \mathbf{a}_{Z_i}$$

- Has to be expanded with 0s non-scalars

Atom-wise layers

$$\mathbf{y}_{i,(lm),f}^t = \sum_{ff'} \mathbf{W}_{ff'} \mathbf{h}_{i(lm)f'}^t$$

SO3 convolution (msg creation)

$$\mathbf{h}_{i(\ell m)f}^t = \sum_{j \in \mathcal{N}_{\mathrm{at}}[i]} \sum_{\ell_1 m_1} \sum_{l_2 m_2} \mathbf{x}_{j(\ell_1 m_1)f}^t \mathbf{R}_{l_2 f}^t(r_{ij}) \mathbf{Y}_{l_2, m_2}(\hat{\mathbf{r}}_{ij}) \mathbf{C}_{\ell_1 m_1 l_2 m_2}^{\ell m}$$

SO3Net

Atom-wise layers

$$\mathbf{y}^t_{i,(lm),f} = \sum_{ff'} \mathbf{W}_{ff'} \mathbf{h}^t_{i(lm)f'}$$

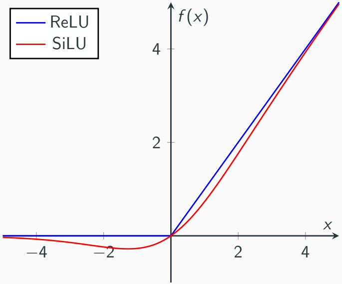
Update

$$\mathbf{x}^{t+1}_{i(\ell m)f} = \mathbf{x}^t_{i(\ell m)f} + \sum_{\ell_1 m_1} \sum_{\ell_2 m_2} \mathbf{x}^t_{i(\ell_1 m_1)f} \mathbf{y}^t_{i(\ell_2 m_2)f} C^{\ell m}_{\ell_1 m_1 \ell_2 m_2}$$

Gating

Conventional NN: Activation Functions

$$\mathbf{y} = \sigma(\mathbf{x})$$



Equivariant NN: Gating Mechanisms

$$\mathbf{y} = \underbrace{\left(\bigoplus_i \sigma_1(x_{l=0,i})\right)}_{\text{Activation of scalar features}} \oplus \underbrace{\left(\bigoplus_j \sigma_2(g_{l=0,j})\, x_{l\neq 0,j}\right)}_{\text{Gating of non-scalar features}}$$

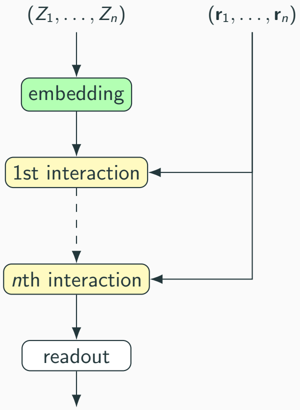
$g_{l=0,j} \longrightarrow$ Additional scalar gating features that activate all features of the same *irrep*, simultaneously and equally!

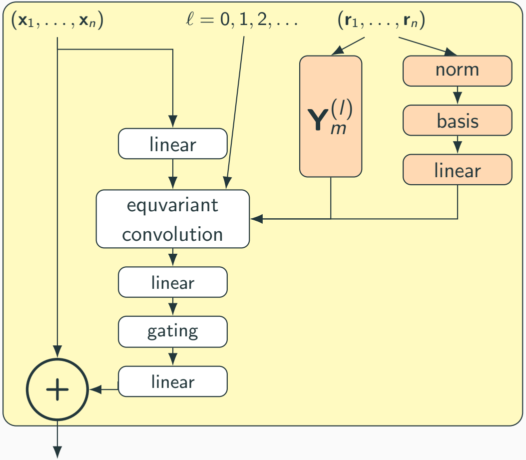
Activations can only be applied to scalar features!

Non-scalar features can be activated using additional scalar features!

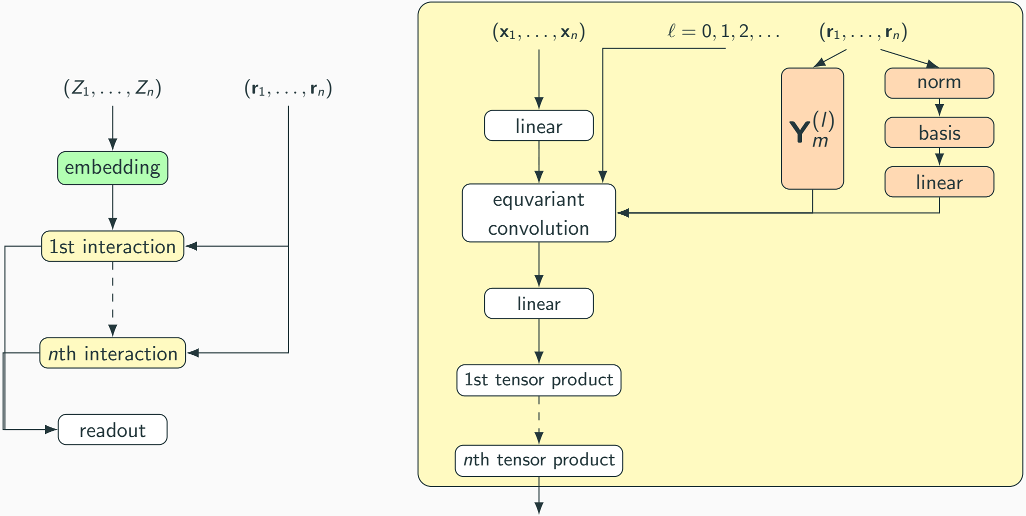
NeQUIP

- Bessels functions instead of Gaussians





MACE



$(Z_1, \ldots, Z_n)$
$(\mathbf{r}_1, \ldots, \mathbf{r}_n)$
embedding
1st interaction
nth interaction
readout
$(\mathbf{x}_1, \ldots, \mathbf{x}_n)$
$\ell = 0, 1, 2, \ldots$
$(\mathbf{r}_1, \ldots, \mathbf{r}_n)$
linear
$\mathbf{Y}_m^{(l)}$
norm
basis
linear
equivariant convolution
linear
1st tensor product
nth tensor product

Install for tomorrow:

- `ASE`
- `torch`
- `e3nn`
- `mace-torch` (from pip install is enough)
- `torch-scatter` see url

Workshop github: