E(n) Equivariant Graph Neural Networks (V. Satorras et al., ICML'21)

Zhiming Xu
zx2rw@virginia.edu

School of Engineering and Applied Science
University of Virginia

January 7, 2022

Table of Contents

Group

A group is a set $\mathcal{G}$ together with a binary operation on it. The operation, denoted by $\cdot$, combines any two elements $a, b \in \mathcal{G}$, to $a \cdot b$, which satisfies

- *Associativity.* $\forall a, b, c \in \mathcal{G}, (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
- *Identity element.* $\exists e \in \mathcal{G}, \forall a \in G, e \cdot a = a \cdot e = a$.
- *Inverse element.* $\forall a \in \mathcal{G}, \exists b \in \mathcal{G}, a \cdot b = b \cdot a = e$. Such b can be denoted as a^{-1}.

Examples: The set of integers with addition. The set of non-zero real numbers with multiplication.

Transformation, Equivariance, Invariance

- *Transformation.* A function (usually geometrically meaningful) f that maps a set X to itself: $f : X \to X$.
- *Equivariance.* A function $\phi : X \to Y$ is said to be equivariant if $S(\phi(\boldsymbol{x})) = \phi(T(\boldsymbol{x}))$, where T is a transformation on X, and S is a transformation on Y.
- *Invariance*. A function $\phi : X \to Y$ is said to be invariant if $\phi(T(\boldsymbol{x})) = \phi(\boldsymbol{x})$, where T is a transformation on X.

Equi- and Invariance

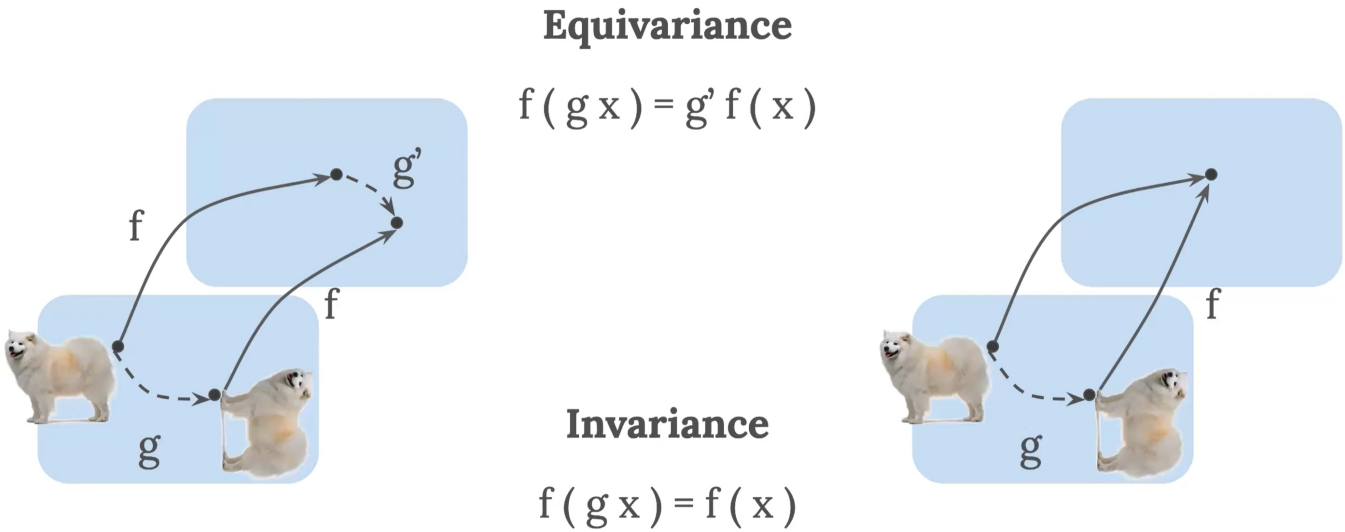

Figure 1: An example of the difference between equivariance and invariance[1]. "Equi" means we can find another transformation g' to make up for the difference made by g after f. "In" means that the difference made by g does not have effect on f.

[1] https://www.youtube.com/watch?v=03MbWVlbefM&t=1393s

Equivariance

In this paper, three kinds of equivariance are considered on $\phi(\boldsymbol{x}) = \boldsymbol{y}$.

- *Translation* equivariance. $T(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{t}$, $\phi(\boldsymbol{x} + \boldsymbol{t}) = \boldsymbol{y} + \boldsymbol{t}$.
- *Rotation* equivariance. $T(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x}$, $\phi(\boldsymbol{Q}\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{y}$.
- *Permutation* equivariance. $T(\boldsymbol{x}) = \mathrm{P}(\boldsymbol{x})$, $\phi(\mathrm{P}(\boldsymbol{x})) = \mathrm{P}(\boldsymbol{y})$.

The Euclidean group $E(n)$ comprises all translations, rotations and reflections of Euclidean space $\mathbb{E}^n$.

Table of Contents

Graph Neural Networks

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\forall v_i \in \mathcal{V}, e_{ij} \in \mathcal{E}$, $\mathcal{N}(i) = \{j \mid e_{ij} \in \mathcal{E}\}$, graph neural networks work as follows

$$\boldsymbol{m}_{ij} = \phi_e(\boldsymbol{h}_i^l, \boldsymbol{h}_j^l, a_{ij})$$

$$\boldsymbol{m}_i = \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij}$$

$$\boldsymbol{h}_i^{l+1} = \phi_h(\boldsymbol{h}_i^l, \boldsymbol{m}_i).$$

Graph Neural Networks (cont.)

We can easily prove that GNNs are permutation equivariant. Suppose $\mathrm{P}: \mathcal{V} \to \mathcal{V}, \forall e_{ij} \in \mathcal{E}, e_{\mathrm{P}(i)\,\mathrm{P}(j)} \in \mathcal{E}$. We have

$$\begin{aligned}
\boldsymbol{m}_{\mathrm{P}(i)\,\mathrm{P}(j)} &= \phi_e\left(\boldsymbol{h}^l_{\mathrm{P}(i)}, \boldsymbol{h}^l_{\mathrm{P}(j)}, a_{\mathrm{P}(i)\,\mathrm{P}(j)}\right) \\
\boldsymbol{m}_{\mathrm{P}(i)} &= \sum_{\mathrm{P}(j) \in \mathcal{N}(\mathrm{P}(i))} \boldsymbol{m}_{\mathrm{P}(i)\,\mathrm{P}(j)} \\
\boldsymbol{h}^{l+1}_{\mathrm{P}(i)} &= \phi_n\left(\boldsymbol{h}^l_{\mathrm{P}(i)}, \boldsymbol{m}_{\mathrm{P}(i)}\right).
\end{aligned}$$

Therefore, we have $\boldsymbol{h}_i = \boldsymbol{h}_{\mathrm{P}(i)} \implies \mathrm{GNN}(\mathrm{P}(\boldsymbol{x})) = \mathrm{P}(\mathrm{GNN}(\boldsymbol{x}))$, i.e. GNN is equivariant under permutation P.

Equivariant Graph Neural Networks

In addition to the node features and edges before, a new node attribute, n-dimensional coordinate $\boldsymbol{x}_i \in \mathbb{R}^n$ is introduced as an additional feature for v_i. Equivariant graph neural networks aim to preserve the equivariance w.r.t. rotations and translations on the set of coordinates.

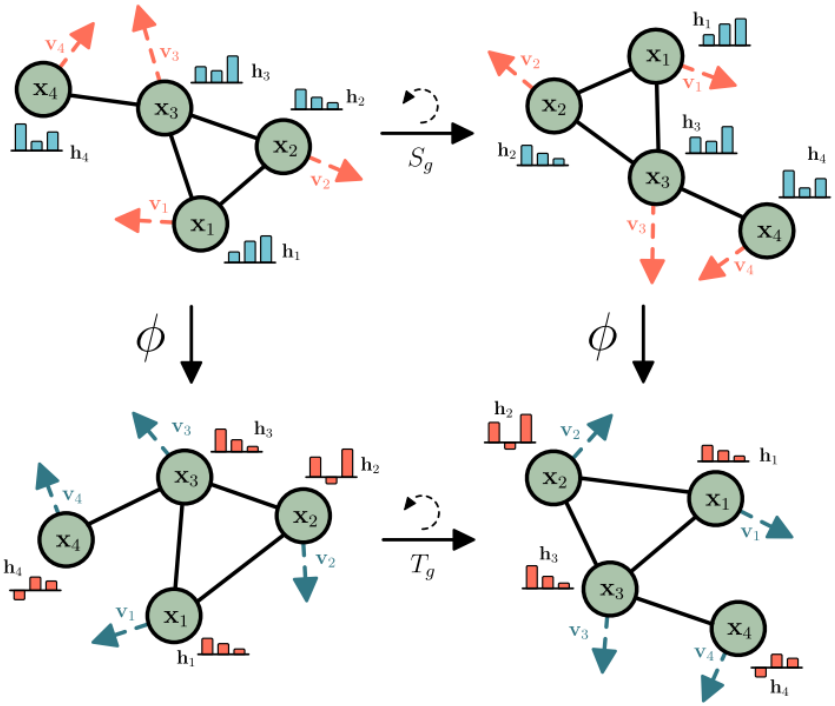

Figure 2: An exmaple of rotation equivariance on GNN ϕ from this paper.

Equivariant Graph Neural Networks (cont.)

Each equivariant graph convolutional layer (EGCL) takes the form of $\boldsymbol{h}^{l+1}, \boldsymbol{x}^{l+1} = \mathrm{EGCL}(\boldsymbol{h}^{l}, \boldsymbol{x}^{l}, \mathcal{E})$.

$$
\begin{aligned}
\boldsymbol{m}_{ij} &= \phi_e\left(\boldsymbol{h}_i^l, \boldsymbol{h}_j^l, \left\|\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right\|^2, a_{ij}\right) \\
\boldsymbol{x}_i^{l+1} &= \boldsymbol{x}_i^l + C \sum_{j \neq i} \left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right) \phi_x(\boldsymbol{m}_{ij}) \\
\boldsymbol{m}_i &= \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij} \\
\boldsymbol{h}_i^{l+1} &= \phi_h\left(\boldsymbol{h}_i^l, \boldsymbol{m}_i\right).
\end{aligned}
$$

Equivariant Graph Neural Networks (cont.)

The major differences between EGCL and GCL is the computation of $\boldsymbol{m}_{ij}$ and $\boldsymbol{x}_i^{l+1}$.

- For each message $\boldsymbol{m}_{ij}$ along the edge between v_i and v_j, the squared Euclidean distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$ between the two nodes are used.
- For each coordinate transformation, $\phi_x : \mathbb{R}^f \to \mathbb{R}$ gives the weight for each relative differences $\boldsymbol{x}_i - \boldsymbol{x}_j$. $C = 1/(M-1)$ averages the weighted sum.

Proof of Equivariance

Intuitively, the distance operations in EGCL are invariant to $E(n)$ transformations, i.e., translation and rotation, and the relative displacement is equivariant to them.

More formally, we can show that the operation computing $\boldsymbol{m}_{ij}$ is

- *Translation Invariant.* $T(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{g}$.
  $\left\|\boldsymbol{x}_i^l + \boldsymbol{g} - (\boldsymbol{x}_j^l + \boldsymbol{g})\right\| = \left\|\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right\|$. Therefore, $\boldsymbol{m}_{ij}$ is invariant to translation.
- *Rotation Invariant.* $T(\boldsymbol{x}) = \boldsymbol{Q}\boldsymbol{x}$.
  $\left\|\boldsymbol{Q}\boldsymbol{x}_i^l - \boldsymbol{Q}\boldsymbol{x}_j^l\right\|^2 = \left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right)\boldsymbol{Q}^\top\boldsymbol{Q}\left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right) = \left\|\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right\|^2$.
  Therefore, $\boldsymbol{m}_{ij}$ is invariant to rotation.

Proof of Equivariance (cont.)

Additionally, we can show that the operation computing $\boldsymbol{x}_i$ is translation and rotation equivariant ($\boldsymbol{m}_{ij}$ is proved to be invariant).

$$
\begin{aligned}
&\boldsymbol{Q}\boldsymbol{x}_i^{l+1} + \boldsymbol{g} + C\sum_{j\neq i}\left(\boldsymbol{Q}\boldsymbol{x}_i^l + \boldsymbol{g} - \boldsymbol{Q}\boldsymbol{x}_j^l - \boldsymbol{g}\right)\phi_x(\boldsymbol{m}_{ij}) \\
&= \boldsymbol{Q}\boldsymbol{x}_i^l + \boldsymbol{g} + \boldsymbol{Q}C\sum_{j\neq i}\left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right)\phi_x(\boldsymbol{m}_{ij}) \\
&= \boldsymbol{Q}\left(\boldsymbol{x}_i^l + C\sum_{j\neq i}\left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right)\phi_x(\boldsymbol{m}_{ij})\right) + \boldsymbol{g} \\
&= \boldsymbol{Q}\boldsymbol{x}_i^{l+1} + \boldsymbol{g}.
\end{aligned}
$$

Extend EGNNs for Velocity

Apart from equivariant to the coordinates $\boldsymbol{x}$, EGNNs can be further extended to be equivariant to the velocity $\boldsymbol{v}$ and keep track of the momentum of each node.

$$\boldsymbol{h}^{l+1}, \boldsymbol{x}^{l+1}, \boldsymbol{v}^{l+1} = \mathrm{EGCL}\left(\boldsymbol{h}^{l}, \boldsymbol{x}^{l}, \boldsymbol{v}^{l}, \mathcal{E}\right)$$

$$\boldsymbol{v}_i^{l+1} = \underbrace{\phi_v\left(\boldsymbol{h}_i^l\right)\boldsymbol{v}_i^l}_{\text{“acceleration”}} + \underbrace{C\sum_{j\neq i}\left(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\right)\phi_x(\boldsymbol{m}_{ij})}_{\text{coordinate transformation}}$$

$$\boldsymbol{x}_i^{l+1} = \boldsymbol{x}_i^l + \underbrace{\boldsymbol{v}_i^{l+1}}_{\substack{\text{“displacement”} \\ \text{in unit time}}} .$$

The equivariance can be shown w.r.t. $\boldsymbol{x}$ and $\boldsymbol{v}$ similarly while the invariance w.r.t. $\boldsymbol{m}_{ij}$ stays the same.

Inferring the Edges

In certain graphs, there might be no adjacency matrix, such as a point cloud. Such graphs are often assumed to be fully connected, thus greatly increasing the complexity in message aggregation. This can be alleviated by adding another neural network $\phi_{inf} : \mathbb{R}^f \to [0, 1]$.

$$\boldsymbol{m}_i = \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij} = \sum_{j \neq i} e_{ij} \boldsymbol{m}_{ij}.$$

$$e_{ij} = \begin{cases} 1, & (v_i, v_j) \in \mathcal{E} \\ 0, & \text{o.w.} \end{cases} \quad \text{for graphs with } \mathcal{E},$$

$$e_{ij} = \phi_{inf}\left(\boldsymbol{m}_{ij}\right) \text{ for graphs without } \mathcal{E}.$$

Table of Contents

N-body System

In a dynamic system, a function defines the time-dependence of a point or a set of points in geometrical space. This paper considers the charged particles N-body experiment in 3D space.

There are five particles carrying either a positive or negative charge. The input is the initial positions $\boldsymbol{x}^{(0)} = \left\{\boldsymbol{x}_1^{(0)}, \cdots, \boldsymbol{x}_5^{(0)}\right\}$, initial velocities $\boldsymbol{v}^{(0)} = \left\{\boldsymbol{v}_1^{(0)}, \cdots, \boldsymbol{v}_5^{(0)}\right\}$, and charges $\boldsymbol{c} = \{c_1, \cdots, c_5\} \in \{-1, 1\}^5$.

N-body System (cont.)

- *Implementation.* The EGNNs with velocity is used. The norm of velocity is transformed into $\boldsymbol{h}$, and the attraction/repulsion between points is transformed into a_{ij}.
- *Results.*

Method	MSE	Forward time (s)
Linear	0.0819	.0001
SE(3) Transformer	0.0244	.1346
Tensor Field Network	0.0155	.0343
Graph Neural Network	0.0107	.0032
Radial Field	0.0104	.0039
EGNN	**0.0071**	.0062

Figure 3: Mean squared errors of the position prediction.

Graph Autoencoder

This paper tests EGNNs as a graph autoencoder. The autoencoder represents a graph in adjacency matrix $\boldsymbol{A}$ and node feature matrix $\boldsymbol{X}$ into $\boldsymbol{Z} \in \mathbb{R}^{|\mathcal{V}| \times d_h}$, and then reconstructs $\boldsymbol{A}$ from $\boldsymbol{Z}$.

The following example shows a special case of a symmetric plain graph. GNNs on this graph is unable to distinguish different nodes since their neighbor topology are the same.

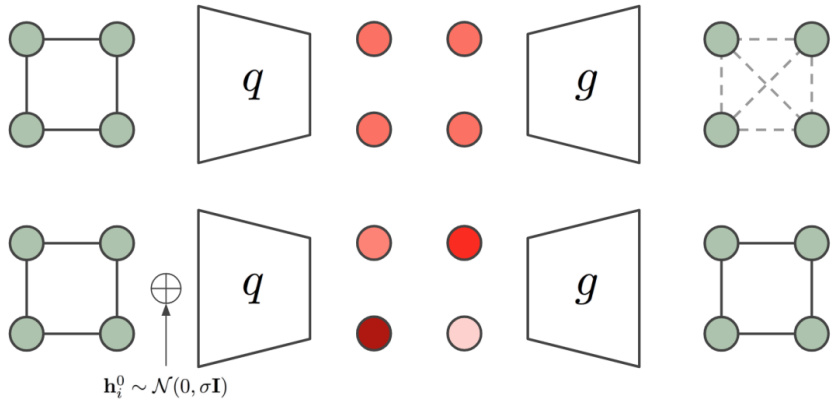

Figure 4: A symmetric cycle graph where all nodes have the same topology.

Graph Autoencoder (cont.)

The symmetric problem in Fig. 4 can be solved by associated each node with a vector sampled from $\mathcal{N}(\mathbf{0}, \sigma \boldsymbol{I})$. However, GNNs need to generalize to the noise distribution.

By using the sampled noises as $\boldsymbol{x}$ in EGNNs, they become translation and rotation equivariant, which makes the generalization easier. The authors provide another explanation that this makes the node representations from structural to positional, where equivariance helps.

	Community Small			Erdos&Renyi		
Encoder	BCE	% Error	F1	BCE	% Error	F1
Baseline	-	31.79	.0000	-	25.13	0.000
GNN	6.75	1.29	0.980	14.15	4.62	0.907
Noise-GNN	3.32	0.44	0.993	4.56	1.25	0.975
Radial Field	9.22	1.19	0.981	6.78	1.63	0.968
EGNN	**2.14**	**0.06**	**0.999**	**1.65**	**0.11**	**0.998**

Figure 5: Metrics on graph autoencoder experiments.

QM9

EGNNs are used to predict molecular properties associated with quantum chemistry. These properties are invariant to translations and rotations. Since in this dataset, each molecule is in a stationary state, no position updates on $\boldsymbol{x}$ are performed.

By simply using the distances, EGNNs can achieve on-par performance compared to complicated models that consider angles or spherical harmonics.

Task Units	α bohr^3	$\Delta\varepsilon$ meV	$\varepsilon_{\text{HOMO}}$ meV	$\varepsilon_{\text{LUMO}}$ meV	μ D	C_ν cal/mol K	G meV	H meV	R^2 bohr^3	U meV	U_0 meV	ZPVE meV
NMP	.092	69	43	38	.030	.040	19	17	.180	20	20	1.50
Schnet	.235	63	41	34	.033	.033	14	14	.073	19	14	1.70
Cormorant	.085	61	34	38	.038	.026	20	21	.961	21	22	2.03
L1Net	.088	68	46	35	.043	.031	14	14	.354	14	13	1.56
LieConv	.084	49	30	25	.032	.038	22	24	.800	19	19	2.28
DimeNet++*	.044	33	25	20	.030	.023	8	7	.331	6	6	1.21
TFN	.223	58	40	38	.064	.101	-	-	-	-	-	-
SE(3)-Tr.	.142	53	35	33	.051	.054	-	-	-	-	-	-
EGNN	.071	48	29	25	.029	.031	12	12	.106	12	11	1.55

Figure 6: Mean absolute errors of twelve prediction targets from QM9 dataset. EGNNs have similar performance to state-of-the-art DimeNet++ [2].

Table of Contents

Summary

- Propose Equivariant Graph Neural Networks (EGNNs) w.r.t. $E(n)$ transformations, i.e. translation and rotation.
- Besides equivariant to positions, EGNNs can be equivariant to velocities in dynamic systems as well.
- Experimentally and theoretically prove that relative distances are sufficient to define molecular geometry. Using distances alone in EGNNs achieves competitive performance on related tasks.

Thank You for Your Attention

Q & A

References

[1] V. G. Satorras, E. Hoogeboom, and M. Welling, "E (n) equivariant graph neural networks," *arXiv preprint arXiv:2102.09844*, 2021.

[2] J. Klicpera, S. Giri, J. T. Margraf, and S. Günnemann, "Fast and uncertainty-aware directional message passing for non-equilibrium molecules," *arXiv preprint arXiv:2011.14115*, 2020.