Equivariant Graph Hierarchy-based Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Equivariant Graph neural Networks (EGNs) are powerful in characterizing the dynamics of multi-body physical systems. Existing EGNs conduct *flat* message passing, which, yet, is unable to capture the spatial/dynamical hierarchy for complex systems particularly, limiting substructure discovery and global information fusion. In this paper, we propose Equivariant Hierarchy-based Graph Networks (EGHNs) which consist of the three key components: generalized Equivariant Matrix Message Passing (EMMP), E-Pool, and E-UnPool. In particular, EMMP is able to improve the expressivity of conventional equivariant message passing, E-Pool assigns the quantities of the low-level nodes into high-level clusters, while E-UnPool leverages the high-level information to update the dynamics of the low-level nodes. As their names imply, both E-Pool and E-UnPool are guaranteed to be E(n)-equivariant to meet the physical symmetry. Considerable experimental evaluations verify the effectiveness of our EGHN on several applications including multi-object dynamics simulation, motion capture, and protein dynamics modeling.

1 Introduction

Understanding the multi-body physical systems is vital to numerous scientific problems, from microscopically how a protein with thousands of atoms acts and folds in the human body to macroscopically how celestial bodies influence each other's movement. While this is exactly an important form of expert intelligence, researchers have paid attention to teaching a machine to discover the physical rules from the observational systems through end-to-end trainable neural networks. Specifically, it is natural to use Graph Neural Networks (GNNs), which is able to model the relations between different bodies into a graph and the inter-body interaction as the message passing thereon [1, 15, 24, 25, 20].

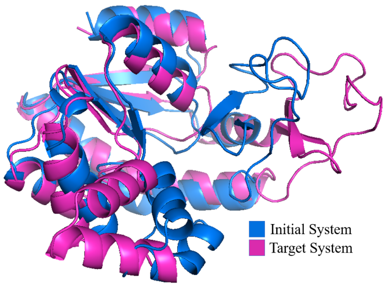

Figure 1: The folding dynamics of proteins in the cartoon format.

More recently, Equivariant GNNs (EGNs) [28, 8, 7, 26] have become a crucial kind of tool for representing multi-body systems. One desirable property is that their outputs are equivariant with respect to any translation/orientation/reflection of the inputs. With this inductive bias encapsulated, EGN permits the symmetry that the physical rules keep unchanged regardless of the reference coordinate system, enabling more enhanced generalization ability. Nevertheless, current EGNs only conduct *flat* message passing in the sense that each layer of message passing in EGN is formulated in the same graph space, where the spatial and dynamical information can only be propagated

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

node-wisely and locally. By this design, it is difficult to discover the hierarchy of the patterns within complex systems.

Hierarchy is common in various domains. Imagine a complex mechanical system, where the particles are distributed on different rigid objects. In this case, for the particles on the same object, their states can be explained as the relative states to the object (probably the center) plus the dynamics of the object itself. We can easily track the behavior of the system if these "implicit" objects are detected automatically by the model we use. Another example, as illustrated in Figure 1, is the dynamics of a protein. Most proteins fold and change in the form of regularly repeating local structures, such as α-helix, β-sheet and turns. By applying a hierarchical network, we are more capable of not only characterizing the conformation of a protein, but also facilitating the propagation between thousands of atoms in a protein by a more efficient means. There are earlier works proposed for hierarchical graph modeling [12, 5, 32, 3, 17], but these studies focus mainly on generic graph classification, and more importantly, they are not equivariant.

In this paper, we propose Equivariant Graph Hierarchy-based Network (EGHN), an end-to-end trainable model to discover local substructures of the input systems, while still maintaining the Euclidean equivariance. In a nutshell, EGHN is composed of an encoder and a decoder. The encoder processes the input system from fine-scale to coarse-scale, where an Equivariant-Pooling (E-Pool) layer is developed to group the low-level particles into each of a certain number of clusters that are considered as the particles of the next layer. By contrast, the decoder recovers the information from the coarse-scale system to the fine-scale one, by using the proposed Equivariant-Up-Pooling (E-UnPool) layer. Both E-Pool and E-UnPool are equivariant with regard to Euclidean transformations via our specific design. EGHN is built upon a generalized equivariant layer, which passes directional matrices over edges other than passing vectors in EGNN [26].

To verify the effectiveness of EGHN, we have simulated a new task extended from the N-body system [15], dubbed M-complex system, where each of the M complexes is a rigid object comprised of a set of particles, and the dynamics of all complexes are driven by the electromagnetic force between particles. In addition to M-complex, we also carry out evaluations on two real applications: human motion caption [4] and the Molecular Dynamics (MD) of proteins [27]. For all tasks, our EGHN outperforms state-of-the-art EGN methods, indicating the efficacy and necessity of the proposed hierarchical modeling idea.

2 Related Work

GNNs for modeling physical interaction. Graph Neural Networks (GNNs) have been widely investigated for modeling physical systems with multiple interacting objects. As pioneer attempts, Interaction Networks [1], NRI [15], and HRN [19] have been introduced to reason about the physical interactions. With the development of neural networks enforced by physical priors, many works resort to injecting physical knowledge into the design of GNNs. As an example, inspired by HNN [11], HOGN [24] models the evolution of interacting systems by Hamiltonian equations to obtain energy conservation. Another interesting feature of physical systems lies in Euclidean equivariance, *i.e.*, translation, rotation, and reflection. Several works first approach translation equivariance [29, 25, 20, 30]. Yet, dealing with rotation equivariance is non-trivial. TFN [28] and SE(3)-Transformer [8] leverages the irreducible representation of the SO(3) group, while LieConv [7] and LieTransformer [14] extend the realization of equivariance to Lie group. Apart from these works that resort to group representation theory, a succinct equivariant message passing scheme on E(n) group is depicted in EGNN [26]. GMN [13] further involves equivariant forward kinematics modeling particularly for constrained systems. [2] generalizes EGNN to involve covariant information with steerable vectors. [21] leverages frame averaging for general equivariance. [18] mainly studies sign and basis invariance. Despite the rich literature, these models either violate the equivariance, or inspect the system at a single granularity, both of which are vital aspects when tackling highly complicated systems like proteins.

Hierarchical GNNs. There are also works that explore the representation learning of GNNs in hierarchies. Several GNNs [12, 5, 31] adopt graph coarsening algorithms to view the graph in multiple granularities. [9] leverages a U-net architecture with top-k pooling. Another line of work injects learnable pooling modules into the model. A differentiable pooling scheme DiffPool [32] has been introduced to learn a permutation-invariant pooling in an end-to-end manner. [3] replaces the

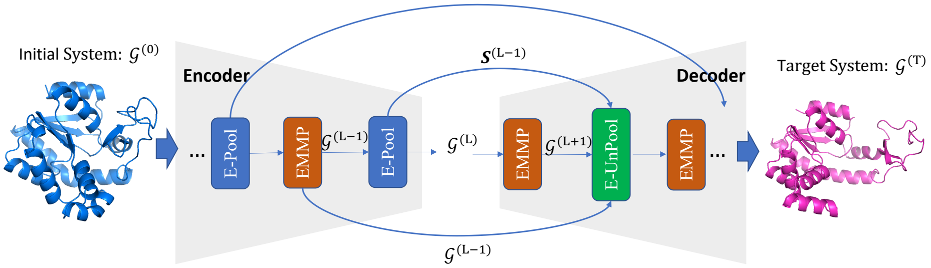

Figure 2: Illustration of the proposed EGHN. It consists of an encoder and a decoder, which are equipped with E-Pool and E-UnPool, respectively. E-UnPool takes as the input the previous output and the score matrix $\boldsymbol{S}$ from E-Pool and output the low-level system $\mathcal{G}$.

aggregation in DiffPool by node dropping for saving the computational cost. [17] further incorporates self-attention mechanism into the pooling network. [6] leverages junction tree to model molecular graph in multiple hierarchies. Nevertheless, these techniques, although permutation equivariant, lack the guarantee of geometric equivariance, limiting their generalization on real-world 3D physical data.

3 The Proposed EGHN

In this section, we first introduce the notations and formulation of our task, and then follow them up by presenting the design of the EMMP layer, which is the basic function in EGHN. Upon EMMP, we provide the details of how the proposed E-Pool and E-UnPool work. Finally, we describe the instantiation of the entire architecture.

3.1 Notations and Formulation

Each input multi-body system is modeled as a graph $\mathcal{G}$ consisting of N particles (nodes) $\mathcal{V}$ and the interactions (edges) $\mathcal{E}$ among them. For each node i, it is assigned with a feature tuple $(\boldsymbol{Z}_i^{(0)}, \boldsymbol{h}_i^{(0)})$, where the directional matrix $\boldsymbol{Z}_i^{(0)} \in \mathbb{R}^{n\times m}$ is composed of m n-dimension vectors, such as the concatenation of position $\boldsymbol{x}_i \in \mathbb{R}^3$ and velocity $\boldsymbol{v}_i \in \mathbb{R}^3$, leading to $\boldsymbol{Z}_i^{(0)} = [\boldsymbol{x}_i, \boldsymbol{v}_i] \in \mathbb{R}^{3\times 2}$; $\boldsymbol{h}_i \in \mathbb{R}^c$ is the non-directional feature, such as the category of the atom in molecules. The edges are represented by an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, which can either be constructed according to the geometric distance or physical connectivity. We henceforth abbreviate the entire information of a system, *i.e.*, $(\{\boldsymbol{Z}_i^{(0)}, \boldsymbol{h}_i^{(0)}\}_{i=1}^N, \boldsymbol{A})$ as the notation $\mathcal{G}^{\text{in}}$ if necessary.

We are mainly interested in investigating the dynamics of the input system $\mathcal{G}^{\text{in}}$. To be formal, given the initial state $(\boldsymbol{Z}_i^{(0)}, \boldsymbol{h}_i^{(0)})$ of each particle, our task is to find out a function ϕ to predict its future state $\boldsymbol{Z}_i^{(T)}$ given the interactions between particles. As explored before [28, 8, 7, 26], ϕ is implemented as a GNN to encode the inter-particle relation. In addition, it should be equivariant to any translation/reflection/rotation of the input states, so as to obey the physics symmetry about the coordinates. It means, $\forall g \in \mathrm{E}(n)$ that defines the Euclidean group [26],

$$\phi(\{g\cdot \boldsymbol{Z}_i^{(0)}\}_{i=1}^N, \cdots) = g\cdot \phi(\{\boldsymbol{Z}_i^{(0)}\}_{i=1}^N, \cdots), \tag{1}$$

where $g\cdot \boldsymbol{Z}_i^{(0)}$ conducts the orthogonal transformation as $\boldsymbol{R}\boldsymbol{Z}_i^{(0)}$ for both the position and velocity vectors and is additionally implemented as the translation $\boldsymbol{x}_i + \boldsymbol{b}$ for the position vector; the ellipsis denotes the input variables uninfluenced by g, including $\boldsymbol{h}_i^{(0)}$ and $\boldsymbol{A}$.

As discussed in Introduction, existing equivariant models [28, 8, 7, 26] are unable to mine the hierarchy within the dynamics of the input system by flat message passing. To address this pitfall, EGHN is formulated in the encoder-decoder form:

$$\mathcal{G}^{\text{high}} = \text{Encode}(\mathcal{G}^{\text{in}}), \mathcal{G}^{\text{out}} = \text{Decode}(\mathcal{G}^{\text{high}}, \mathcal{G}^{\text{in}}). \tag{2}$$

Here, as illustrated in Figure 2, the encoder aims at clustering the particles of $\mathcal{G}^{\text{in}}$ with similar dynamics into a group that is treated as the particle in the high-level graph $\mathcal{G}^{\text{high}}$ (the number of the nodes in $\mathcal{G}^{\text{high}}$ is smaller than $\mathcal{G}^{\text{in}}$). We have developed a novel component, E-Pool to fulfill this goal.

As for the decoder, it recovers the information of all particles in the original graph space under the guidance of the high-level system $\mathcal{G}^{\text{high}}$, which is accomplished by the proposed E-UnPool. It is worth mentioning that both E-Pool and E-UnPool, as their names imply, are equivariant, and they are mainly built upon an expressive and generalized equivariant message passing layer, EMMP. To facilitate the understanding of our model, we first introduce the details of this layer in what follows.

3.2 Equivariant Matrix Message Passing

Given input features $\{(\boldsymbol{Z}_i, \boldsymbol{h}_i)\}_{i=1}^N$, EMMP performs information aggregation on the same graph to obtain the new features $\{(\boldsymbol{Z}_i', \boldsymbol{h}_i')\}_{i=1}^N$.

$$\boldsymbol{H}_{ij}, \boldsymbol{h}_{ij} = \mathrm{MLP}\left(\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}, \boldsymbol{h}_i, \boldsymbol{h}_j\right), \tag{3}$$

$$\boldsymbol{M}_{ij} = \hat{\boldsymbol{Z}}_{ij} \boldsymbol{H}_{ij}, \tag{4}$$

$$\boldsymbol{h}_i' = \mathrm{MLP}\left(\boldsymbol{h}_i, \sum\nolimits_{j \in \mathcal{N}(i)} \boldsymbol{h}_{ij}\right), \tag{5}$$

$$\boldsymbol{Z}_i' = \boldsymbol{Z}_i + \sum\nolimits_{j \in \mathcal{N}(i)} \boldsymbol{M}_{ij}, \tag{6}$$

The dimension of the output features could be different from the input, unless the row dimension of $\boldsymbol{Z}_i'$ should keep the same as $\boldsymbol{Z}_i$ (*i.e.* equal to n). In detail, one EMMP layer is updated by Eq. 3-6, where $\mathrm{MLP}(\cdot)$ is a Multi-Layer Perceptron, $\mathcal{N}(i)$ collects the neighbors of i, and $\hat{\boldsymbol{Z}}_{ij} \in \mathbb{R}^{n \times 2m} = (\boldsymbol{Z}_i - \bar{\boldsymbol{Z}}, \boldsymbol{Z}_j - \bar{\boldsymbol{Z}})$ is a concatenation of the translated matrices on the edge ij. $\bar{\boldsymbol{Z}}$ is the mean of all nodes for the position vectors and zero for other vectors. With the subtraction of $\bar{\boldsymbol{Z}}$, $\hat{\boldsymbol{Z}}_{ij}$ is ensured to be translation invariant, and then $\boldsymbol{Z}_i'$ is translation equivariant after the addition of $\boldsymbol{Z}_i$ in Eq. 6. Specifically, the MLP in Eq. 3 takes as input the concatenation of the E(n)-invariant $\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}, \boldsymbol{h}_i$, and $\boldsymbol{h}_j$, mapping from $\mathbb{R}^{2m \times 2m + 2c}$ to $\mathbb{R}^{2m \times m + c}$, and the output is split into $\boldsymbol{H}_{ij} \in \mathbb{R}^{2m \times m}$ and $\boldsymbol{h}_{ij} \in \mathbb{R}^c$. The formal proof for the E(n)-equivariance of EMMP is deferred to Appendix.

Distinct from EGNN [26], the messages to pass in EMMP are directional matrices other than vectors. Although GMN [13] has also explored the matrix form, it is just a specific case of our EMMP by simplifying $\hat{\boldsymbol{Z}}_{ij} = \boldsymbol{Z}_i - \boldsymbol{Z}_j$. Indeed, we have the following theorem for the comparison of expressivity between EMMP, EGNN, and GMN, with the proof in Appendix.

Theorem 1. *EMMP can reduce to EGNN and GMN by specific choices of MLP in Eq. 3.*

Besides, since taking the inner product might induce a larger variance in the scale of input, in our implementation we also enforce a normalization $\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij} / \|\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}\|_F$ before feeding the invariant $\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}$ into the MLP in Eq. 3, following the suggestion by GMN for better numerical stability.

3.3 Equivariant Pooling

Inspired by DiffPool, we propose E-Pool, an equivariant pooling module. Formally, E-Pool coarsens the low-level system $\mathcal{G}^{\text{low}} = (\{(\boldsymbol{Z}_i^{\text{low}}, \boldsymbol{h}_i^{\text{low}})\}_{i=1}^N, \boldsymbol{A}^{\text{low}})$ into an abstract and high-level system $\mathcal{G}^{\text{high}} = (\{(\boldsymbol{Z}_i^{\text{high}}, \boldsymbol{h}_i^{\text{high}})\}_{i=1}^K, \boldsymbol{A}^{\text{high}})$ with fewer particles, $K < N$. For this purpose, we first perform EMMP (Eq. 3-6) over the input system $\mathcal{G}$ to capture the local topology of each node. Then we apply the updated features of each node to predict which cluster it belongs to. This can be realized by a SoftMax layer to output a soft score for each of the K clusters. The cluster is deemed as a node of the high-level system, and its features are computed as a weighted combination of the low-level nodes with the scores it just derives. In summary, we proceed:

$$\{\boldsymbol{Z}_i^{'}, \boldsymbol{h}_i^{'}\}_i^N = \mathrm{EMMP}(\{\boldsymbol{Z}_i^{\text{low}}, \boldsymbol{h}_i^{\text{low}}\}_i^N, \boldsymbol{A}^{\text{low}}), \tag{7}$$

$$\boldsymbol{s}_i = \mathrm{Softmax}(\mathrm{MLP}(\boldsymbol{h}_i^{'})), \tag{8}$$

$$\boldsymbol{Z}_j^{\text{high}} = \frac{1}{\sum_{i=1}^N s_{ij}} \sum_{i=1}^N s_{ij} \boldsymbol{Z}_i^{'}, \tag{9}$$

$$\boldsymbol{h}_j^{\text{high}} = \frac{1}{\sum_{j=1}^N s_{ij}} \sum_{i=1}^N s_{ij} \boldsymbol{h}_i^{\text{low}}, \tag{10}$$

$$\boldsymbol{A}^{\text{high}} = \boldsymbol{S}^\top \boldsymbol{A}^{\text{low}} \boldsymbol{S}, \tag{11}$$

where Eq. 8 maps the invariant feature $\boldsymbol{h}_i'$ into the score $\boldsymbol{s}_i \in \mathbb{R}^K$ of cluster assignment with Softmax performed long the feature dimension, and the score matrix is given by $\boldsymbol{S} = [s_{ij}]_{N \times K}$ with $\boldsymbol{s}_i$ being its i-th row. By this design, it is tractable to verify that E-Pool is guaranteed to be E(n) equivariant (also permutation equivariant). Specifically, the division by the row-wise sum $\sum_{i=1}^N s_{ij}$ in Eq. 9 is essential, as it permits the translation equivariance, that is, $\frac{1}{\sum_{i=1}^N s_{ij}} \sum_{i=1}^N s_{ij}(\boldsymbol{Z}_i^{'} + \boldsymbol{b}) = \left(\frac{1}{\sum_{i=1}^N s_{ij}} \sum_{i=1}^N s_{ij} \boldsymbol{Z}_i^{'}\right) + \boldsymbol{b}$. This particular property distinguishes our pooling from traditional non-equivariant graph pooling [32, 17]. Notice that the normalization in Eq. 10 is unnecessary since $\boldsymbol{h}_i$ is a non-directional vector, but it is still adopted in line with Eq. 9. In practice, it is difficult to attain desirable clusters by using the SoftMax layer solely; instead, the pooling results are enhanced if we regulate the training process with an extra reconstruction loss related to the score matrix, whose formulation will be given in § 3.5.

3.4 Equivariant UnPooling

E-UnPool maps the information of the high-level system $\mathcal{G}^{\text{high}}$ back to the original system space $\mathcal{G}^{\text{low}}$, leading to an output system $\mathcal{G}^{\text{out}}$. We project the features back to the space of the original low-level system by using the transposed scores derived in E-Pool. Then, the projected features along with the low-level features are integrated by an E(n) equivariant function to give the final output. The procedure of E-UnPool is given by Eq. 12-15, where $\hat{\boldsymbol{Z}}_i = [\boldsymbol{Z}_i^{\text{low}} - \bar{\boldsymbol{Z}}^{\text{low}}; \boldsymbol{Z}_i^{\text{agg}} - \bar{\boldsymbol{Z}}^{\text{agg}}]$ is the column-wise concatenation of the mean-translated low-level matrix $\boldsymbol{Z}_i^{\text{low}}$ and the high-level matrix $\boldsymbol{Z}_i^{\text{agg}}$, analogous to Eq. 3. One interesting point is that Eq. 12 is naturally equivariant in terms of translations, even without the normalization term used in Eq. 9. This is because the score matrix is summed to 1 for each row, indicating that $\sum_{j=1}^{K} s_{ij}(\boldsymbol{Z}_j^{\text{high}} + \boldsymbol{b}) = \sum_{j=1}^{K} s_{ij}\boldsymbol{Z}_j^{\text{high}} + \boldsymbol{b}$. We have the following theorem guaranteeing the equivariance, with all proofs deferred to Appendix.

$$\boldsymbol{Z}_i^{\text{agg}} = \sum_{j=1}^{K} s_{ij}\boldsymbol{Z}_j^{\text{high}}, \tag{12}$$

$$\boldsymbol{h}_i^{\text{agg}} = \sum_{j=1}^{K} s_{ij}\boldsymbol{h}_j^{\text{high}}, \tag{13}$$

$$\boldsymbol{h}_i^{\text{out}} = \text{MLP}\left(\hat{\boldsymbol{Z}}_i^\top \hat{\boldsymbol{Z}}_i, \boldsymbol{h}_i^{\text{low}}, \boldsymbol{h}_i^{\text{agg}}\right), \tag{14}$$

$$\boldsymbol{Z}_i^{\text{out}} = \hat{\boldsymbol{Z}}_i \boldsymbol{h}_i^{\text{out}} + \boldsymbol{Z}_i^{\text{agg}}, \tag{15}$$

Theorem 2. *EMMP, E-Pool, and E-UnPool are all E(n)-equivariant.*

3.5 Instantiation of the Architecture

The overall architecture constitutes an equivariant U-Net [23] with skip-connections. We design the overall architecture as a sequence of EMMP, E-Pool, and E-UnPool in an encoder-decoder fashion, as depicted in Figure 2. The encoder is equipped with a certain number of E-Pools and EMMPs, while the decoder is realized with E-UnPools and EMMPs. For each E-UnPool in the decoder, as already defined in § 3.4, it is fed with the output of the previous layer, the score matrix $\boldsymbol{S}$ from E-Pool, and the low-level system $\mathcal{G}$ from EMMP in the corresponding layers of the encoder. Here, the so-called corresponding layers in E-Pool and E-UnPool are referred to the ones arranged in an inverse order; for example, in Figure 2, the final E-Pool corresponds to the first E-UnPool. With such design, it is straightforward, by the conclusion of Theorem 3, that the resulting EGHN still satisfies E(n)-equivariance.

There is always one EMMP layer prior to each E-Pool or E-UnPool. This external EMMP plays a different role from the internal EMMP used in E-Pool (Eq. 7). One crucial difference is that they leverage different adjacency matrices. As we have introduced before, the adjacency matrix $\boldsymbol{A}$ can either be specified by geometric distance, *i.e.*, distance-based, or physical connectivity, *i.e.* connectivity-based. **1.** The external EMMP exploits a *distance-based* $\boldsymbol{A}_{\text{global}}$ whose element is valued if the distance between two particles is less than a threshold; by such means, we are able to characterize the force interaction between any two particles even they are physically disconnected. In higher-layer external EMMP, its $\boldsymbol{A}_{\text{global}}$ is created as a re-scored form (akin to Eq. 11) of $\boldsymbol{A}_{\text{global}}$ in lower layer, where the score matrix is obtained by its front E-Pool. **2.** For the internal EMMP in E-Pool, it applies a *connectivity-based* $\boldsymbol{A}_{\text{local}}$ that exactly reflects the physical connection between particles, for example, it is valued 1 if there is a bond between two atoms. In this way, E-Pool pays more attention to locally-connected particles when conducting clustering. Another point is that the external EMMP is relaxed as EGNN for modeling the radial interaction, whereas the internal EMMP uses the generalized form in § 3.2. As we will show in our experiments in § 4.4 and Appendix D.2, such design yields more favorable results compared with using any one of $\boldsymbol{A}_{\text{global}}$ and $\boldsymbol{A}_{\text{local}}$ only.

The training objective of EGHN is given by:

$$\mathcal{L} = \sum_{i=1}^{N} \|\boldsymbol{Z}_i^{\text{out}} - \boldsymbol{Z}_i^{\text{gt}}\|_F^2 + \lambda \sum_{l=1}^{L} \|(\boldsymbol{S}^{(l)})^\top \boldsymbol{A}^{(l-1)} \boldsymbol{S}^{(l)} - \boldsymbol{I}\|_F^2, \tag{16}$$

Table 1: Prediction error ($\times 10^{-2}$) on various types of simulated datasets. The "Multiple System" contains $J = 5$ different systems. For each column, $(M, N/M)$ indicates that each system contains M complexes of average size N/M. Results averaged across 3 runs. "OOM" denotes out of memory.

	Single System				Multiple Systems			
	(3, 3)	(5, 5)	(5, 10)	(10, 10)	(3, 3)	(5, 5)	(5, 10)	(10, 10)
Linear	35.15±0.01	35.22±0.00	30.14±0.00	31.44±0.01	35.91±0.01	35.29±0.01	30.88±0.01	32.49±0.01
TFN [28]	25.11±0.15	29.35±0.17	26.01±0.22	OOM	27.33±0.21	29.01±0.13	25.57±0.14	OOM
SE(3)-Tr. [8]	27.12±0.26	28.87±0.09	24.48±0.35	OOM	28.14±0.16	28.66±0.10	25.00±0.28	OOM
MPNN [10]	16.00±0.11	17.55±0.19	16.15±0.08	15.91±0.15	16.76±0.13	17.58±0.11	16.55±0.21	16.05±0.16
RF [16]	14.20±0.09	18.37±0.12	17.08±0.03	18.57±0.30	15.17±0.10	18.55±0.12	17.24±0.11	19.34±0.25
EGNN [26]	12.69±0.19	15.37±0.13	15.12±0.11	14.64±0.27	13.33±0.12	15.48±0.16	15.29±0.12	15.02±0.18
EGHN	**11.58**±0.01	**14.42**±0.08	**14.29**±0.40	**13.09**±0.66	**12.80**±0.56	**14.85**±0.03	**14.50**±0.08	**13.11**±0.92

where $\|\cdot\|_F$ computes the Frobenius norm, L is the number of E-Pools in the encoder, and λ is the trade-off weight. The first term is to minimize the mean-square-error between the output state $\boldsymbol{Z}_i^{\text{out}}$ and the ground truth $\boldsymbol{Z}_i^{\text{gt}}$. The second term is the connectivity loss that encourages more connects within the pooling nodes and less cuts among pooling clusters [33]. For training stability, we first perform row-wise normalization of $(\boldsymbol{S}^{(l)})^\top \boldsymbol{A}^{(l-1)} \boldsymbol{S}^{(l)}$ before substituting it into Eq. 16.

4 Experiments

We contrast the performance of the proposed EGHN against a variety of baselines including the equivariant and non-equivariant GNNs, on one simulation task: the M-complex system, and the two real-world applications: human motion capture and molecular dynamics on proteins. We also carry out a complete set of ablation studies to verify the optimal design of our model.

4.1 Simulation Dataset: M-complex System

Data generation. We extend the N-body simulation system from [15] and generate the M-complex simulation dataset, in order to introduce hierarchical structures in the data. Specifically, we initialize a system with N charged particles $\{\boldsymbol{x}_i, \boldsymbol{v}_i, c_i\}_{i=1}^N$ distributed on M disjoint complex objects $\{\mathcal{S}_j\}_{j=1}^M$, where $\boldsymbol{x}_i, \boldsymbol{v}_i, c_i$ are separately the position, velocity, and charge for each particle. Within each complex $\mathcal{S}_j$, the particles are connected by rigid sticks, yielding geometric objects like sicks, triangles, tetrahedrons, etc. The dynamics of all M complexes are driven by the electromagnetic force between every pair of particles. The task here is to predict the final positions $\{\boldsymbol{x}_i^T\}_i^N$ of all particles when $T = 1500$ given their initial positions and velocities. Without knowing which complex each particle belongs to, we will also test if our EGHN can group the particles correctly just based on the distribution of the trajectories. We independently sample J different systems, each of which has M complexes with the number of particles sampled from a uniform distribution with mean N/M. A dataset consists J systems with M complexes, N/M average size of complex is abbreviated as $(M, N/M, J)$. We adopt Mean Squared Error (MSE) as the evaluation metric for the experiments.

Implementation details. We assign the node feature as the norm of the velocity $\|\boldsymbol{v}_i\|_2$, and the edge attribute as $c_i c_j$ for the edge connecting node i and j, following the setting in [26]. We also concatenate an indicator, which is set as 1 if a stick presents and 0 otherwise, to the edge feature, similar to [13]. We use a fully connected graph (without self-loops) as $\boldsymbol{A}_{\text{global}}$, since the interaction force spans across each pair of particles in the system. The adjacency matrix $\boldsymbol{A}$ reflects the connectivity of the particles formed by the complexes. We set the number of clusters the same as the number of complexes in the dataset. The comparison models include: Linear Prediction (Linear) [26], SE(3)-Transformer (SE(3)-Tr.) [8], Radial-Field (RF) [16], GNN and EGNN [26]. For all these models, we employ the codes and architectures implemented by [26]. Detailed hyper-parameter settings are in Appendix.

Results. Table 1 reports the overall performance of the comparison models on eight simulation datasets with different configurations. From Table 1, we have the following observations: **1.** Clearly, EGHN surpasses all other approaches in all cases, demonstrating the general superiority of its design.

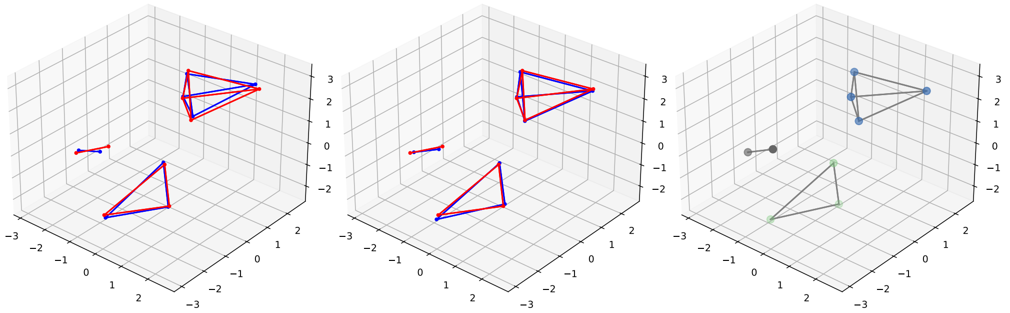

Figure 3: Visualization on M-complex systems. *Left*: the prediction of EGNN. *Middle*: the prediction of EGHN. *Right*: the pooling results of EGHN with each color indicating a cluster. In the left and middle figure, ground truth in red, and prediction in blue. Best viewed by colour printing.

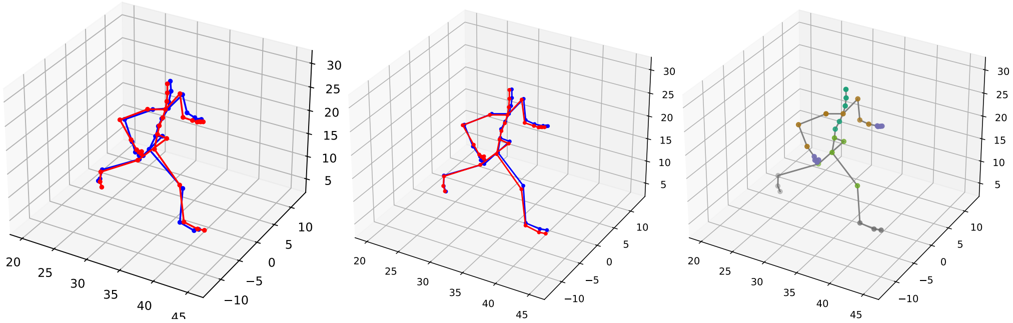

Figure 4: Visualization on Motion Capture. *Left*: the prediction of EGNN. *Middle*: the prediction of EGHN. *Right*: the pooling results of EGHN with each color indicating a cluster. In the left and middle figure, ground truth in red, and prediction in blue. Best viewed by zooming in.

2. Increasing the number of complexes (M) or the number of particles (N) always increases the complexity of the input system, but this does not necessarily hinder the performance of EGHN. For example, in both the single-system and multiple-system cases, EGHN even performs better when the system is changed from $(5, 5)$ to $(5, 10)$ and $(10, 10)$. We conjecture that, with more particles/complexes, larger systems also provide more data samples to enhance the training of EGHN. **3.** When increasing the diversity of systems (J) by switching from the single-system mode to multi-system mode, the performance of EGHN only drops slightly, indicating its adaptability to various scenarios. **Visualization.** we visualize in Figure 3 the predictions of EGNN and our EGHN on the $(3, 3, 1)$ scenario. We find that EGHN predicts the movements of the rigid objects more accurately than EGNN, especially for the large objects. In the right sub-figure, we also display the pooling results of EGHN, outputted by the score matrix of the final E-Pool layer. It is observed that EGHN is able to detect the correct cluster for each particle. This is interesting and it can justify the worth of designing hierarchical architecture for multi-body system modeling.

4.2 Motion Capture

We further evaluate our model on CMU Motion Capture Databse [4]. We primarily focus on two activities, namely *walking* (Subject #35) [15] and *running* (Subject #9). With regard to walking, we leverage the random split adopted by [13], which includes 200 frame pairs for training, 600 for validation, and another 600 for testing. As for running, we follow a similar strategy and obtain a split with 200/240/240 frame pairs. The interval between each pair is 30 frames in both scenarios. In this task the joints are edges and their intersections are the nodes.

Implementation details. As discussed in [8], many real-world tasks, including our motion capture task here, break the Euclidean symmetry along the gravity axis ($z-$axis), and it is beneficial to make the equivariant models aware of where the top is. To this end, we augment the node feature by the coordinate of the $z-$axis, resulting in models that are height-aware while still equivariant in the horizontal directions. This operation is also applied to all baselines. Since the interaction of human

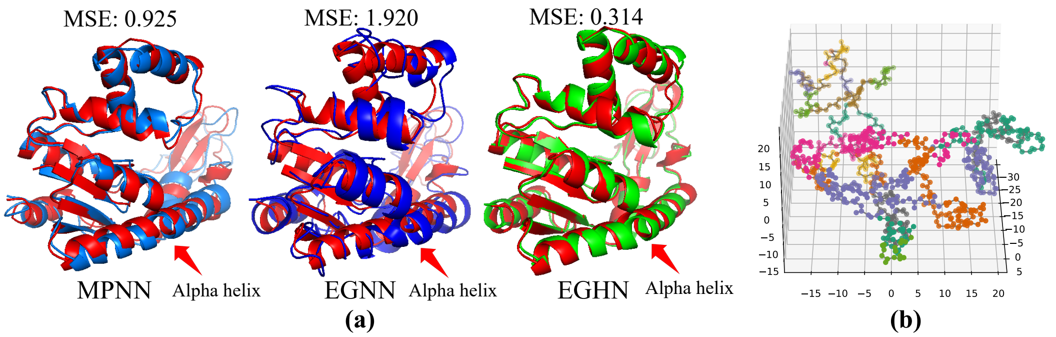

Figure 5: Visualization on the MDAnalysis dataset. (a) The predictions of MPNN, EGNN, and EGHN. Ground truth is in red. The top-1 MSE of EGHN is also much smaller that that of MPNN and EGHN. (b) The pooling assignment of EGHN.

body works along the joints, we propose to involve the edge in $\boldsymbol{A}_{\text{global}}$ if it connects the nodes within two hops in $\mathcal{G}$. For the number of clusters K, we empirically find that $K = 5$ yields promising results for both walking and running.

Results. Table 2 summarizes the whole results of all models on two subjects. Here, we supplement an additional baseline GMN [13] for its promising performance on this task. Excitingly, EGHN outperforms all compared baselines by a large margin on both activities. Particularly, on Subject #35, the prediction error of EGHN is 8.5×10^{-2}, which is much lower than that of the best baseline, *i.e.*, GMN (21.6×10^{-2}). **Visualization.** To investigate why EGHN works, we depict the skeletons estimated by both EGNN and EGHN on Subject #9 in Figure 4. It shows that EGHN is able to capture more fine-grained details on certain parts (*e.g.* the junction between the legs and the body) than EGNN. When we additionally visualize the pooling outcome in the right sub-figure, we interestingly find that EGHN is capable of classifying the two right-left hands into the same cluster even they are spatially disconnected. A similar result is observed for the arms and feet. This is reasonable as EGHN checks not only if two particles are spatially close to each other but also if they share the similar dynamics.

Table 2: MSE ($\times 10^{-2}$) on the motion capture dataset averaged across 3 runs.

	Subject #35	Subject #9
	Walk	Run
MPNN [10]	36.1 ±1.5	66.4 ±2.2
RF [16]	188.0 ±1.9	521.3±2.3
TFN [28]	32.0 ±1.8	56.6 ±1.7
SE(3)-Tr. [8]	31.5 ±2.1	61.2 ±2.3
EGNN [26]	28.7 ±1.6	50.9 ±0.9
GMN [13]	21.6 ±1.5	44.1 ±2.3
EGHN	**8.5** ±2.2	**25.9** ±0.3

4.3 Molecular Dynamics on Proteins

We adopt AdK equilibrium trajectory dataset [27] via MDAnalysis toolkit [22] to evaluate our hierarchical model. The AdK equilibrium trajectory dataset involves the MD trajectory of apo adenylate kinase simulated with explicit water and ions in NPT at 300 K and 1 bar. The atoms' positions of the protein are saved every 240 ps for a total of 1.004 μs as frames.

Implementation details. We split the dataset into train/validation/test sets along the timeline that contain 2481/827/878 frame pairs respectively. We choose $T = 15$ as the span between the input and prediction frames. We ignore the hydrogen atoms to focus on the prediction of large atoms. We further establish the global adjacency matrix as the neighboring atoms within a distance of 10Å. The atoms' velocities of the protein at each frame are computed by subtracting the positions to the previous frame's positions. We further leverage MDAnalysis to extract the protein backbone in order to reduce the data scale. Even so, TFN and SE(3)-Transformer still run out of memory, and thus we compare our model with the rest of baselines. Detailed hyper-parameters are in Appendix.

Results. The prediction MSE is depicted in Table 3. Our EGHN yields significantly lower error on protein MD compared with the baselines, achieving 1.843 MSE, while the second best model MPNN has an MSE of 2.322. However, MPNN is non-equivariant, and we find

Table 3: Prediction error (MSE) on protein MD.

Linear	RF [16]	MPNN [10]	EGNN [26]	EGHN
2.890	2.846	2.322	2.735	**1.843**

that its MSE will dramatically increase to 605.7 if we apply a random rotation of the protein during testing. Compared with EGNN, our EGHN exhibits its superiority thanks to the hierarchical modeling, particularly favorable on large and complex systems like proteins.

Qualitative comparisons. We visualize the protein structure of top-1 predictions generated by different models in cartoon format in Fig. 5 (a), with more visualization examples provided in Appendix. In Fig. 5 (a), the structure in red indicates the ground truth, while the other colors indicate the prediction. We can observe that EGHN tracks the folding and dynamics of the protein more precisely than the baselines. For example, in the the bottom region, EGHN gives a close-fitting result of the alpha helix structure while the predictions from MPNN and EGNN have an obvious shift compared with the ground truth. To validate the power of the E-Pool, we further visualize the pooling clusters in Fig. 5 (b). Interestingly, the pooling assignment exhibits certain clusters in some structures of the protein. It suggests that EGHN discovers local repetitive sub-structures of the protein; for instance, it detects the alpha helix structure in the middle of the protein.

4.4 Ablation Studies

We investigate the necessity of our proposed components on motion capture dataset in Table 4. We study the following questions:

Table 4: Ablation studies on the motion capture dataset. Numbers are MSE ($\times 10^{-2}$).

	Subject #35	Subject #9
	Walk	Run
EGHN ($K = 5$)	**8.5**	**25.9**
EGHN ($K = 3$)	10.1	41.4
EGHN ($K = 8$)	14.9	26.8
w/o Equivariance	19.7	40.9
w/o Hierarchy	21.9	42.1
Replace by EGNN	22.3	42.5
w/o Connectivity loss	10.5	28.8
$\boldsymbol{A}_{\text{global}}$ only	17.4	31.5
$\boldsymbol{A}_{\text{local}}$ only	16.8	33.5

Q1. *How will the performance of EGHN change, if we vary the number of clusters (K)?* We modify the number of clusters K from 5 to 3 and 8, both of which yield worse performance. Specifically, we find that decreasing K on "Run" results in a larger degradation of performance, possibly because the activity "Run" is with complicated kinematics and it will be more difficult to learn if the joints are shared across a too small number of clusters. We provide potential guidance on choosing K in Appendix D.1. **Q2.** *How do our proposed two components EMMP and hierarchical modeling contribute?* We replace all EMMP layers in our model by typical non-equivariant MPNN, and the performance drops from 8.5 to 19.7 on Walk, supporting that maintaining equivariance is vital. We further set $\boldsymbol{s}_i = \mathbf{1}_i$ in all E-Pool and E-UnPool and observe that removing hierarchy is detrimental to accurate prediction. Moreover, by replacing all EMMPs with EGNNs, the performance also drops, which aligns with our analysis on the stronger expressivity of EMMP over EGNN. Complete studies are deferred to Appendix D.2 and D.3. **Q3.** *How does the connectivity loss (the second term in Eq. 16) help?* By dropping the connectivity loss, we observe a larger prediction error. This justifies the necessity of using the connectivity loss to focus more on intra-cluster connections against the inter-cluster edges. **Q4.** *How about using the same adjacency matrix for all EMMP instead of distinguishing them as $\boldsymbol{A}_{global}$ in the external EMMPs and $\boldsymbol{A}_{local}$ in internal EMMPs as discussed in § 3.5?* When we apply $\boldsymbol{A}_{\text{global}}$ or $\boldsymbol{A}_{\text{local}}$ for all EMMPs, the performance drops dramatically, implying that the external and internal EMMPs play different roles in EGHN, and should be equipped with different adjacency matrices to model the interactions of different scopes.

5 Discussion

Limitation. In the current form the number of clusters K is fixed in EGHN as an empirical hyperparameter. Future works include extending E-Pool to dynamically adjust K for systems with different scales for enhancing the flexibility of the hierarchical model.

Conclusion. In this paper, we propose Equivariant Graph Hierarchy-based Network (EGHN) to model and represent the dynamics of multi-body systems. EGHN leverages E-Pool to group the low-level nodes into clusters, and E-UnPool to restore the low-level information from the high-level systems with the aid of the corresponding E-Pool layer. The fundamental layer of EGHN lies in Equivariant Matrix Message Passing (EMMP) to characterize the topology and dynamics expressively. Experimental evaluations on M-complex systems, Motion-Capture, and protein MD, show that EGHN consistently outperforms other non-hierarchical EGNs as well as non-equivariant GNNs.

References

[1] Peter W Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *arXiv preprint arXiv:1612.00222*, 2016. 1, 2

[2] Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve e(3) equivariant message passing. In *International Conference on Learning Representations*, 2022. 2, 17

[3] Cătălina Cangea, Petar Veličković, Nikola Jovanović, Thomas Kipf, and Pietro Liò. Towards sparse hierarchical graph classifiers, 2018. 2

[4] CMU. Carnegie-mellon motion capture database. 2003. 2, 7

[5] Chenhui Deng, Zhiqiang Zhao, Yongyu Wang, Zhiru Zhang, and Zhuo Feng. Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding. In *International Conference on Learning Representations*, 2020. 2

[6] M. Fey, J. G. Yuen, and F. Weichert. Hierarchical inter-message passing for learning on molecular graphs. In *ICML Graph Representation Learning and Beyond (GRL+) Workhop*, 2020. 3

[7] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, pages 3165–3176. PMLR, 2020. 1, 2, 3

[8] Fabian B Fuchs, Daniel E Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020. 1, 2, 3, 6, 7, 8, 14, 17

[9] Hongyang Gao and Shuiwang Ji. Graph u-nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2083–2092. PMLR, 09–15 Jun 2019. 2

[10] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017. 6, 8, 13, 17

[11] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[12] Fenyu Hu, Yanqiao Zhu, Shu Wu, Liang Wang, and Tieniu Tan. Hierarchical graph convolutional networks for semi-supervised node classification. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, (IJCAI)*, 2019. 2

[13] Wenbing Huang, Jiaqi Han, Yu Rong, Tingyang Xu, Fuchun Sun, and Junzhou Huang. Equivariant graph mechanics networks with constraints. In *International Conference on Learning Representations*, 2022. 2, 4, 6, 7, 8, 12, 13, 14, 17

[14] Michael J Hutchinson, Charline Le Lan, Sheheryar Zaidi, Emilien Dupont, Yee Whye Teh, and Hyunjik Kim. Lietransformer: equivariant self-attention for lie groups. In *International Conference on Machine Learning*, pages 4533–4543. PMLR, 2021. 2

[15] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018. 1, 2, 6, 7

[16] Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: sampling configurations for multi-body systems with symmetric energies. *arXiv preprint arXiv:1910.00753*, 2019. 6, 8, 13

[17] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *Proceedings of the 36th International Conference on Machine Learning*, 09–15 Jun 2019. 2, 3, 5

[18] Derek Lim, Joshua David Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022. 2

[19] Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li Fei-Fei, Joshua B Tenenbaum, and Daniel LK Yamins. Flexible neural representation for physics prediction. *arXiv preprint arXiv:1806.08047*, 2018. 2

[20] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2020. 1, 2

[21] Omri Puny, Matan Atzmon, Heli Ben-Hamu, Edward J Smith, Ishan Misra, Aditya Grover, and Yaron Lipman. Frame averaging for invariant and equivariant network design. *arXiv preprint arXiv:2110.03336*, 2021. 2

[22] Richard J. Gowers, Max Linke, Jonathan Barnoud, Tyler J. E. Reddy, Manuel N. Melo, Sean L. Seyler, Jan Domański, David L. Dotson, Sébastien Buchoux, Ian M. Kenney, and Oliver Beckstein. MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In Sebastian Benthall and Scott Rostrup, editors, *Proceedings of the 15th Python in Science Conference*, pages 98 – 105, 2016. 8

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 5

[24] Alvaro Sanchez-Gonzalez, Victor Bapst, Kyle Cranmer, and Peter Battaglia. Hamiltonian graph networks with ode integrators. *arXiv preprint arXiv:1909.12790*, 2019. 1, 2

[25] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pages 8459–8468. PMLR, 2020. 1, 2

[26] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021. 1, 2, 3, 4, 6, 8, 12, 13, 17

[27] Sean Seyler and Oliver Beckstein. Molecular dynamics trajectory for benchmarking mdanalysis, 6 2017. *URL: https://figshare. com/articles/Molecular_dynamics_ trajectory_for_benchmarking_MDAnalysis/5108170, doi*, 10:m9. 2, 8

[28] Nathaniel Thomas, 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. *arXiv preprint arXiv:1802.08219*, 2018. 1, 2, 3, 6, 8, 14, 17

[29] Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2019. 2

[30] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds, 2018. 2

[31] Yifan Xing, Tong He, Tianjun Xiao, Yongxin Wang, Yuanjun Xiong, Wei Xia, David Wipf, Zheng Zhang, and Stefano Soatto. Learning hierarchical graph neural networks for image clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3467–3477, October 2021. 2

[32] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 2, 5

[33] Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. In *International Conference on Learning Representations*, 2020. 6

A Proofs

A.1 Proof of Theorem 1

Theorem 3. *EMMP can reduce to EGNN and GMN by specific choices of MLP in Eq. 3.*

Proof. For simplicity, we denote $\boldsymbol{Z}_i - \bar{\boldsymbol{Z}}$ as $\bar{\boldsymbol{Z}}_i$, which infers $\bar{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}}_j = \boldsymbol{Z}_i - \boldsymbol{Z}_j$.

For EMMP, GMN [13], and EGNN [26], we rewrite their messages (Eq. 3-4) below.

$$
\begin{aligned}
\boldsymbol{M}_{ij}^{\text{EMMP}} &= \hat{\boldsymbol{Z}}_{ij}\text{MLP}_1\left(\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}\right), \\
&= \begin{bmatrix}\bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j\end{bmatrix} \text{MLP}_1\left(\begin{bmatrix}\bar{\boldsymbol{Z}}_i^\top \bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_i^\top \bar{\boldsymbol{Z}}_j \\ \bar{\boldsymbol{Z}}_j^\top \bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j^\top \bar{\boldsymbol{Z}}_j\end{bmatrix}\right). \\
\boldsymbol{M}_{ij}^{\text{GMN}} &= (\boldsymbol{Z}_i - \boldsymbol{Z}_j)\text{MLP}_2\left((\boldsymbol{Z}_i - \boldsymbol{Z}_j)^\top(\boldsymbol{Z}_i - \boldsymbol{Z}_j)\right). \\
\boldsymbol{M}_{ij}^{\text{EGNN}} &= (\boldsymbol{x}_i - \boldsymbol{x}_j)\text{MLP}_3\left((\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\right).
\end{aligned}
$$

1. We first prove that EMMP can reduce to GMN.

Let $\text{MLP}_1 = f_{\text{out}} \circ \text{MLP}_2 \circ f_{\text{in}}$, where $f_{\text{in}}(\begin{bmatrix}\boldsymbol{a}_{11} & \boldsymbol{a}_{12} \\ \boldsymbol{a}_{21} & \boldsymbol{a}_{22}\end{bmatrix}) = (\boldsymbol{a}_{11} - \boldsymbol{a}_{12}) - (\boldsymbol{a}_{21} - \boldsymbol{a}_{22})$, $f_{\text{out}}(\boldsymbol{a}) = \begin{bmatrix}\boldsymbol{a} \\ -\boldsymbol{a}\end{bmatrix}$, and "$\circ$" is the function composition. By this relaxation, EMMP reduces to:

$$
\begin{aligned}
\boldsymbol{M}_{ij}^{\text{EMMP}} &= \begin{bmatrix}\bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j\end{bmatrix} f_{\text{out}} \circ \text{MLP}_2 \circ f_{\text{in}}\left(\begin{bmatrix}\bar{\boldsymbol{Z}}_i^\top \bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_i^\top \bar{\boldsymbol{Z}}_j \\ \bar{\boldsymbol{Z}}_j^\top \bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j^\top \bar{\boldsymbol{Z}}_j\end{bmatrix}\right), \\
&= \begin{bmatrix}\bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j\end{bmatrix} f_{\text{out}} \circ \text{MLP}_2\left(\bar{\boldsymbol{Z}}_i^\top(\bar{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}}_j) - \bar{\boldsymbol{Z}}_j^\top(\bar{\boldsymbol{Z}}_i - \bar{\boldsymbol{Z}}_j)\right), \\
&= \begin{bmatrix}\bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j\end{bmatrix} f_{\text{out}}\left(\text{MLP}_2\left((\boldsymbol{Z}_i - \boldsymbol{Z}_j)^\top(\boldsymbol{Z}_i - \boldsymbol{Z}_j)\right)\right), \\
&= \begin{bmatrix}\bar{\boldsymbol{Z}}_i & \bar{\boldsymbol{Z}}_j\end{bmatrix} \begin{bmatrix}\text{MLP}_2\left((\boldsymbol{Z}_i - \boldsymbol{Z}_j)^\top(\boldsymbol{Z}_i - \boldsymbol{Z}_j)\right) \\ -\text{MLP}_2\left((\boldsymbol{Z}_i - \boldsymbol{Z}_j)^\top(\boldsymbol{Z}_i - \boldsymbol{Z}_j)\right)\end{bmatrix}, \\
&= (\boldsymbol{Z}_i - \boldsymbol{Z}_j)\text{MLP}_2\left((\boldsymbol{Z}_i - \boldsymbol{Z}_j)^\top(\boldsymbol{Z}_i - \boldsymbol{Z}_j)\right), \\
&= \boldsymbol{M}_{ij}^{\text{GMN}}.
\end{aligned}
$$

2. We then prove that GMN can reduce to EGNN using similar derivations as above.

Denote $\boldsymbol{Z}_i = [\boldsymbol{x}_i, \boldsymbol{v}_i]$, and we can similarly let $\text{MLP}_2 = f_{\text{out}} \circ \text{MLP}_3 \circ f_{\text{in}}$, where $f_{\text{in}}(\begin{bmatrix}\boldsymbol{a}_{11} & \boldsymbol{a}_{12} \\ \boldsymbol{a}_{21} & \boldsymbol{a}_{22}\end{bmatrix}) = \boldsymbol{a}_{11}$, and $f_{\text{out}}(\boldsymbol{a}) = \begin{bmatrix}\boldsymbol{a} \\ \boldsymbol{0}\end{bmatrix}$. Therefore, we have that

$$
\begin{aligned}
\boldsymbol{M}_{ij}^{\text{GMN}} &= [\boldsymbol{x}_i - \boldsymbol{x}_j \quad \boldsymbol{v}_i - \boldsymbol{v}_j]\, f_{\text{out}} \circ \text{MLP}_3 \circ f_{\text{in}}\left(\begin{bmatrix}(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j) & (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{v}_i - \boldsymbol{v}_j) \\ (\boldsymbol{v}_i - \boldsymbol{v}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j) & (\boldsymbol{v}_i - \boldsymbol{v}_j)^\top(\boldsymbol{v}_i - \boldsymbol{v}_j)\end{bmatrix}\right), \\
&= [\boldsymbol{x}_i - \boldsymbol{x}_j \quad \boldsymbol{v}_i - \boldsymbol{v}_j]\, f_{\text{out}} \circ \text{MLP}_3\left((\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\right), \\
&= [\boldsymbol{x}_i - \boldsymbol{x}_j \quad \boldsymbol{v}_i - \boldsymbol{v}_j] \begin{bmatrix}\text{MLP}_3\left((\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\right) \\ \boldsymbol{0}\end{bmatrix}, \\
&= (\boldsymbol{x}_i - \boldsymbol{x}_j)\text{MLP}_3\left((\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)\right), \\
&= \boldsymbol{M}_{ij}^{\text{EGNN}},
\end{aligned}
$$

which concludes the proof. □

This theorem basically implies that the expressivity of our EMMP is stronger than that of GMN or EGNN.

A.2 Proof of Theorem 2

Theorem 4. *EMMP, E-Pool, and E-UnPool are all E(n)-equivariant.*

Proof. **1.** We first prove that EMMP is E(n)-equivariant.

For any $g \in \mathrm{E}(n)$, we have $g \cdot \boldsymbol{Z} = \boldsymbol{R}\boldsymbol{Z} + \boldsymbol{b}$ where $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$, $\boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{I}$ and $\boldsymbol{b} \in \mathbb{R}^3$. We use the superscript $*$ to denote the resulting output after applying the group action g to the input. Initially, we have $\boldsymbol{Z}^* = \boldsymbol{R}\boldsymbol{Z} + \boldsymbol{b}$, and $\boldsymbol{h}_i^* = \boldsymbol{h}_i$. Similarly, $\bar{\boldsymbol{Z}}^* = \boldsymbol{R}\bar{\boldsymbol{Z}} + \boldsymbol{b}$. We proceed the proof step by step, following the definition of EMMP in Eq. 3-6:

$$\begin{aligned}
\hat{\boldsymbol{Z}}_{ij}^* &= [\boldsymbol{Z}_i^* - \bar{\boldsymbol{Z}}^*, \boldsymbol{Z}_j^* - \bar{\boldsymbol{Z}}^*] = [\boldsymbol{R}\boldsymbol{Z}_i + \boldsymbol{b} - (\boldsymbol{R}\bar{\boldsymbol{Z}} + \boldsymbol{b}), \boldsymbol{R}\boldsymbol{Z}_j + \boldsymbol{b} - (\boldsymbol{R}\bar{\boldsymbol{Z}} + \boldsymbol{b})] = \boldsymbol{R}\hat{\boldsymbol{Z}}_{ij},\\
\boldsymbol{H}_{ij}^* &= \mathrm{MLP}\left((\boldsymbol{R}\hat{\boldsymbol{Z}}_{ij})^\top \boldsymbol{R}\hat{\boldsymbol{Z}}_{ij}, \boldsymbol{h}_i, \boldsymbol{h}_j\right) = \mathrm{MLP}\left(\hat{\boldsymbol{Z}}_{ij}^\top \hat{\boldsymbol{Z}}_{ij}, \boldsymbol{h}_i, \boldsymbol{h}_j\right) = \boldsymbol{H}_{ij},\\
\boldsymbol{M}_{ij}^* &= \hat{\boldsymbol{Z}}_{ij}^* \boldsymbol{H}_{ij}^* = \boldsymbol{R}\hat{\boldsymbol{Z}}_{ij}\boldsymbol{H}_{ij} = \boldsymbol{R}\boldsymbol{M}_{ij},\\
\boldsymbol{h}_i'^* &= \mathrm{MLP}\left(\boldsymbol{h}_i, \sum_{j\in\mathcal{N}(i)} \boldsymbol{H}_{ij}\right) = \boldsymbol{h}_i',\\
\boldsymbol{Z}_i'^* &= \boldsymbol{R}\boldsymbol{Z} + \boldsymbol{b} + \sum_{j\in\mathcal{N}(i)} \boldsymbol{R}\boldsymbol{M}_{ij} = \boldsymbol{R}(\boldsymbol{Z} + \sum_{j\in\mathcal{N}(i)} \boldsymbol{M}_{ij}) + \boldsymbol{b} = \boldsymbol{R}\boldsymbol{Z}_i' + \boldsymbol{b},
\end{aligned}$$

which verifies that EMMP is E(n)-equivariant.

2. We then prove that E-Pool is E(n)-equivariant.

$$\begin{aligned}
\boldsymbol{Z}_j^{\text{high},*} &= \frac{1}{\sum_{i=1}^N s_{ij}} \sum_{i=1}^N s_{ij}(\boldsymbol{R}\boldsymbol{Z}_i' + \boldsymbol{b}) = \boldsymbol{R}(\frac{1}{\sum_{i=1}^N s_{ij}} \sum_{i=1}^N s_{ij}\boldsymbol{Z}_i') + \boldsymbol{b} = \boldsymbol{R}\boldsymbol{Z}_j^{\text{high}} + \boldsymbol{b},\\
\boldsymbol{h}_j^{\text{high},*} &= \frac{1}{\sum_{j=1}^N s_{ij}} \sum_{i=1}^N s_{ij}\boldsymbol{h}_i^{\text{low}} = \boldsymbol{h}_j^{\text{high}},\\
\boldsymbol{A}^{\text{high},*} &= \boldsymbol{S}^\top \boldsymbol{A}^{\text{low}} \boldsymbol{S} = \boldsymbol{A}^{\text{high}},
\end{aligned}$$

which clearly shows that E-Pool is E(n)-equivariant, while the high-level adjacency matrix $\boldsymbol{A}^{\text{high}}$ is E(n)-invariant, which is crucial for maintaining the equivariance of the high-level EMMP.

3. Finally we prove that E-UnPool is E(n)-equivariant.

$$\begin{aligned}
\boldsymbol{Z}_i^{\text{agg},*} &= \sum_{j=1}^K s_{ij}(\boldsymbol{R}\boldsymbol{Z}_j^{\text{high}} + \boldsymbol{b}) = \boldsymbol{R}(\sum_{j=1}^K s_{ij}\boldsymbol{Z}_j^{\text{high}}) + \boldsymbol{b} = \boldsymbol{R}\boldsymbol{Z}_i^{\text{agg}} + \boldsymbol{b},\\
\boldsymbol{h}_i^{\text{agg},*} &= \boldsymbol{h}_i^{\text{agg}},\\
\boldsymbol{h}_i^{\text{out},*} &= \mathrm{MLP}\left((\boldsymbol{R}\hat{\boldsymbol{Z}}_i)^\top(\boldsymbol{R}\hat{\boldsymbol{Z}}_i), \boldsymbol{h}_i^{\text{low}}, \boldsymbol{h}_i^{\text{agg}}\right) = \boldsymbol{h}_i^{\text{out}},\\
\boldsymbol{Z}_i^{\text{out},*} &= \boldsymbol{R}\hat{\boldsymbol{Z}}_i\boldsymbol{h}_i^{\text{out}} + \boldsymbol{R}\boldsymbol{Z}_i^{\text{agg}} + \boldsymbol{b} = \boldsymbol{R}\boldsymbol{Z}_i^{\text{out}} + \boldsymbol{b}.
\end{aligned}$$

□

Indeed, with Theorem 4 we immediately have that any cascade of EMMP, E-Pool, and E-UnPool is also E(n)-equivariant. This indicates that our resulting EGHN is E(n)-equivariant.

B Implementation Details

Baselines. For the baselines, we leverage the codebases maintained by [13][1] and [26][2], which are released under MIT license. We tune the hyper-parameters around the suggested hyper-parameters as specified in [13] and [26] for the baselines. Specifically, for MPNN [10], RF [16] and EGNN [26], we tune the learning rate from {1e-4, 5e-4, 1e-3}, weight decay {1e-12, 1e-10, 1e-8, 1e-4}, batch

[1] https://github.com/hanjq17/GMN

[2] https://github.com/vgsatorras/egnn

size {50, 100, 200}, hidden dim {32, 64, 128} and the number of layers {2, 4, 6, 8}. For TFN [28] and SE(3)-Transformer [8], we set the degree to 2 due to memory limitation, and select the learning rate from {5e-4, 1e-3, 5e-3}, weight decay {1e-10, 1e-8}, batch size {25, 50, 100}, hidden dim {32, 64} and the number of layers {2, 4}. We report the best results searched within these ranges of hyper-parameters for the baselines. We use an early-stopping of 50 epochs for all methods. Note that the kinematics decomposition trick in GMN [13] requires a specific design to enforce hard constraints for any new system, which cannot be directly applied to our simulation dataset and protein MD. Besides, both TFN and SE(3)-Transformer run out of memory on protein MD, and we thus omit their results in Table 3.

EGHN. For our EGHN, on simulation dataset, we use batch size 50, and the number of clusters the same as the complexes in the dataset. On motion capture, we use batch size 12, and the number of clusters $K = 5$ on both datasets. On MD dataset, we use batch size 8, and the number of clusters $K = 15$. Table 5 depicts the rest of detailed hyper-parameter configurations. Notably, to control the computational budget of EGHN compared with the baselines, we set the maximum number of encoder/decoder layers as 4, while for the baselines we set the maximum number of layers as 8, ensuring fair comparison. All experiments are conducted on NVIDIA Tesla V100 GPU.

Table 5: Hyper-parameters of EGHN.

Dataset	learning rate	λ	weight decay	Encoder Layer	Decoder Layer
(3, 3, 1)	0.0005	4	1e-4	4	2
(3, 3, 5)	0.001	4	1e-4	4	2
(5, 5, 1)	0.0003	2	1e-6	4	2
(5, 5, 5)	0.001	0.1	1e-12	4	2
(5, 10, 1)	0.0001	4	1e-4	2	2
(5, 10, 5)	0.0005	4	1e-4	4	2
(10, 10, 1)	0.0005	2	1e-6	4	2
(10, 10, 5)	0.0003	1	1e-8	4	2
Mocap Walk	0.0004	1	1e-6	2	2
Mocap Run	0.0003	1	1e-6	4	1
MD	0.0005	0.1	1e-8	4	2

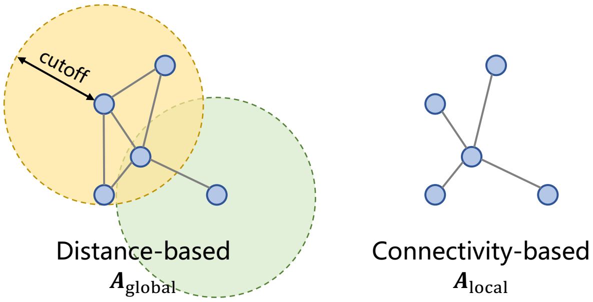

Figure 6: An illustration of A_{global} and A_{local}.

Besides, to gain more insights of our design of A_{global} and A_{local}, we provide an illustration in Fig. 6. Our intuition is that the relation modeling in different hierarchy levels might contain different semantics. For example, in the external EMMP, we use A_{global} since we would like the model to capture and gather the interaction forces based on the distance between nodes (atoms). As for the internal EMMP, the topology of the graph, *i.e.*, the connectivity, plays an important role in determining the topological information (such as the bond connection in molecules and proteins) which is crucial for performing pooling and unpooling. Our connectivity loss, by sharing a similar idea, also enforces a stronger connectivity on the pooling assignment by encouraging connected nodes to be pooled into the same cluster and penalizing the others. By this design, EGHN is designed to be more flexible and the ablations also verify the efficacy of leveraging A_{global} and A_{local} in external and internal EMMP, respectively.

Furthermore, in order to keep a fair comparison between EGHN and the baselines, we augment the edge feature of the baselines by taking into account the information of $\boldsymbol{A}_{\text{global}}$ and $\boldsymbol{A}_{\text{local}}$. Specifically, for the set of edges we employ $\boldsymbol{A}_{\text{global}}$, while extending a channel on the edge feature by an indicator function that takes the value 1 if this edge also belongs to $\boldsymbol{A}_{\text{local}}$ and 0 otherwise. On all the three datasets, it is satisfied that $\boldsymbol{A}_{\text{local}}$ is always a subset of $\boldsymbol{A}_{\text{global}}$ by our choices. Therefore, through such augmentation, we exactly keep the same edge information between EGHN and baselines without any unfairness.

Our implementation is provided in the following anonymous repository https://anonymous.4open.science/r/EGHN_code.

More explanations on the connectivity loss. Intuitively, the connectivity loss encourages pooling assignments with more edges within the pooled clusters and fewer in between. In particular, the loss reaches its minimum, *i.e.*, 0, if and only if node i and j belong to the same cluster for each edge $(i, j) \in \mathcal{E}$.

C Learning Curve

We provide the learning curve of EGHN and EGNN on (3, 3, 1) of the M-complex dataset. It is illustrated that EGHN converges faster and the corresponding testing loss is lower as well, yielding better performance than EGNN.

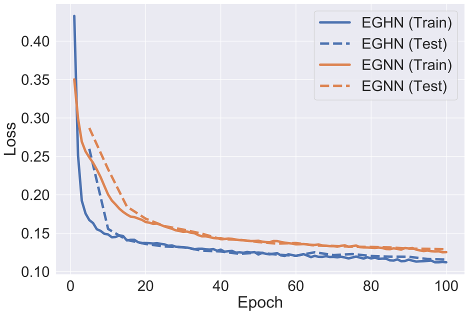

Figure 7: The learning curves of EGHN and EGNN on (3, 3, 1) of the M-complex dataset.

D More ablation studies

D.1 The impact of the number of clusters K

We thoroughly investigate how the number of clusters influence the model performance on all datasets. For M-complex System, we sweep over 1 to 5 in the Complex (3, 3) single system. For Mocap dataset, we sweep over 1 to 8. For Protein MD, we vary K from 1, 5, 10, 15, 20, 25. The results are depicted in Table 6, 7, and 8. We also provide the number of nodes of each system in these tables. A visualization can be found in Fig. 8.

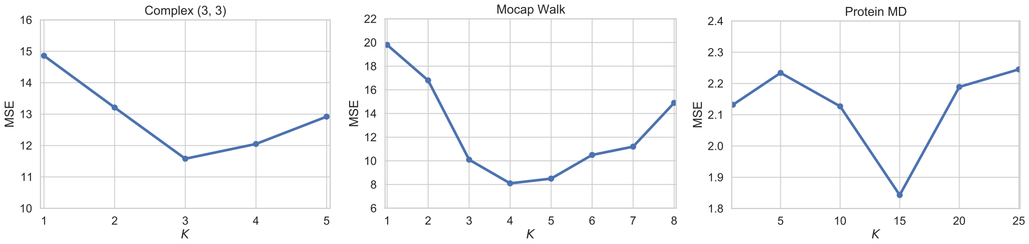

Figure 8: Prediction MSE *w.r.t.* the number of clusters K.

Table 6: MSE ($\times 10^{-2}$) on Complex (3, 3) *w.r.t.* the number of clusters K.

9 nodes	1	2	3	4	5
MSE	14.86	13.21	**11.58**	12.05	12.92

Table 7: MSE ($\times 10^{-2}$) on Mocap Walk *w.r.t.* the number of clusters K.

31 nodes	1	2	3	4	5	6	7	8
MSE	19.8	16.8	10.1	**8.1**	8.5	10.5	11.2	14.9

Table 8: MSE ($\times 10^{-2}$) on Protein MD *w.r.t.* the number of clusters K.

855 nodes	1	5	10	15	20	25
MSE	2.132	2.234	2.127	**1.843**	2.189	2.245

We have these investigations: **1.** On all datasets, the performance degenerates when $K = 1$, since all nodes in the system are pooled into one cluster and therefore there are no learnable cluster assignments. It verifies the necessity of modeling hierarchies in multi-body systems. **2.** The systems with larger scale enjoys larger K in practice. It indicates that for the systems with larger number of nodes, it is beneficial to choose larger K to better model their complex hierarchies. **3.** For the Complex (3,3) system, it is interesting that the best performance is obtained when $K = 3$, since it contains 3 disjoint complexes. This implies that it is also possible to choose K by some prior knowledge assessed from data.

D.2 The choice of internal and external modules

In this subsection we provide ablation study that compares the performance of different choices between internal/external EMMP/EGNN. The experimental results are exhibited in Table 9.

Table 9: MSE ($\times 10^{-2}$) on two motion capture datasets and two M-Complex systems.

Internal	External	Mocap Walk	Mocap Run	Complex (3, 3)	Complex (5, 5)
EGNN	EGNN	22.3	42.5	12.51	15.77
EMMP	EGNN	8.5	21.9	**11.58**	14.42
EMMP	EMMP	**8.1**	**21.1**	11.82	**14.36**

We have the following observations:

- When applying EMMP in either internal or external message passing, the performance consistently improves against EGNN. This verifies that the proposed EMMP is potentially more advantageous on modeling interactions, which aligns with our theoretical analysis that EMMP is more expressive than EGNN (c.f. Theorem 3).
- Compared with external EMMP, more significant improvements are obtained when applying EMMP as the internal message passing layers (e.g., 22.3 → 8.5 on MocapWalk). Note that the internal message passing layers are those right before our pooling layer, which are responsible for passing and aggregating messages towards the high-level cluster nodes. Therefore, we speculate the reason might be that compared with the flat message passing layers (the external EMMPs), the internal EMMPs require much higher expressivity and capacity since they need to fuse the message of all nodes towards their corresponding cluster nodes.
- In the Complex (3, 3) scenario, changing from EGNN to EMMP in external message passing slightly affects the performance, probably because the interactions between nodes in M-complex are Coulomb forces which can be well covered by EGNN. Nevertheless, on

the mocap dataset where interactions are much more complicated, leveraging EMMP is consistently more advantageous over EGNN.

D.3 The hierarchy ablation study with identity assignments.

We summarize in Table 10 the results of more ablation studies on all datasets (simulation, mocap, and protein), where EGHN w/o hier is implemented by setting the cluster assignment to identity, *i.e.*, $\mathbf{s}_i = \mathbf{1}_i$.

Table 10: MSE ($\times 10^{-2}$) on five datasets with and without identity assignments.

	Complex (3,3)	Complex (5,5)	Mocap Walk	Mocap Run	Protein MD
EGHN	**11.58**	**14.42**	**8.5**	**25.9**	**1.8**4
EGHN w/o hier	12.24	15.18	21.9	42.1	2.00

As illustrated in Table 10, the hierarchical structure is consistently beneficial to the model performance across M-complex simulation, Motion Capture, and Protein MD. This supports the validity and efficacy of our designed equivariant hierarchy module.

E Training time comparison

We evaluate the training time on simulation and motion capture datasets for the baselines and EGHN. Table 11 depicts the average training time per epoch (in seconds). All models are trained on a NVIDIA V100 GPU.

Table 11: The average training time per epoch (in seconds) on two datasets.

	MPNN [10]	TFN [28]	SE(3)-Tr. [8]	EGNN [26]	GMN [13]	EGHN
Complex (3, 3)	1.21	7.81	23.25	1.45	1.58	1.69
MocapWalk	0.92	6.85	18.96	1.21	1.49	1.41

EGHN is almost as efficient as EGNN and GMN, while only adding marginal computational overhead compared to MPNN, since the computations related to equivariance and pooling are efficient. The irreps-based methods TFN and SE(3)-Transformer yield significantly longer training time.

F Comparison with additional baselines

We also compare with SEGNN [2] on M-complex systems. The results are in Table 12. SEGNN performs better than EGNN particularly when the system is large (*e.g.*, on (5, 10) or (10, 10). Still, EGHN consistently outperforms these baselines by a significant margin.

Table 12: Prediction error ($\times 10^{-2}$) on various types of simulated datasets. The "Multiple System" contains $J = 5$ different systems. For each column, $(M, N/M)$ indicates that each system contains M complexes of average size N/M. Results averaged across 3 runs. "OOM" denotes out of memory.

	Single System				Multiple Systems			
	(3, 3)	(5, 5)	(5, 10)	(10, 10)	(3, 3)	(5, 5)	(5, 10)	(10, 10)
EGNN [26]	12.69	15.37	15.12	14.64	13.33	15.48	15.29	15.02
SEGNN [2]	14.04	15.62	15.01	14.31	13.88	16.01	15.41	14.78
EGHN	**11.58**	**14.42**	**14.29**	**13.09**	**12.80**	**14.85**	**14.50**	**13.11**

G More Visualizations

In this section, we provide more visualization results. Figure 10, Figure 11, and Figure 12 illustrate more visualization examples on (5, 5, 1) of the simulation dataset, walking on the motion capture dataset, and the MD dataset, respectively.

We further provide more predictions and pooling results of EGHN in Fig. 9. It is observed that EGHN gives accurate predictions with desirable pooling assignments.

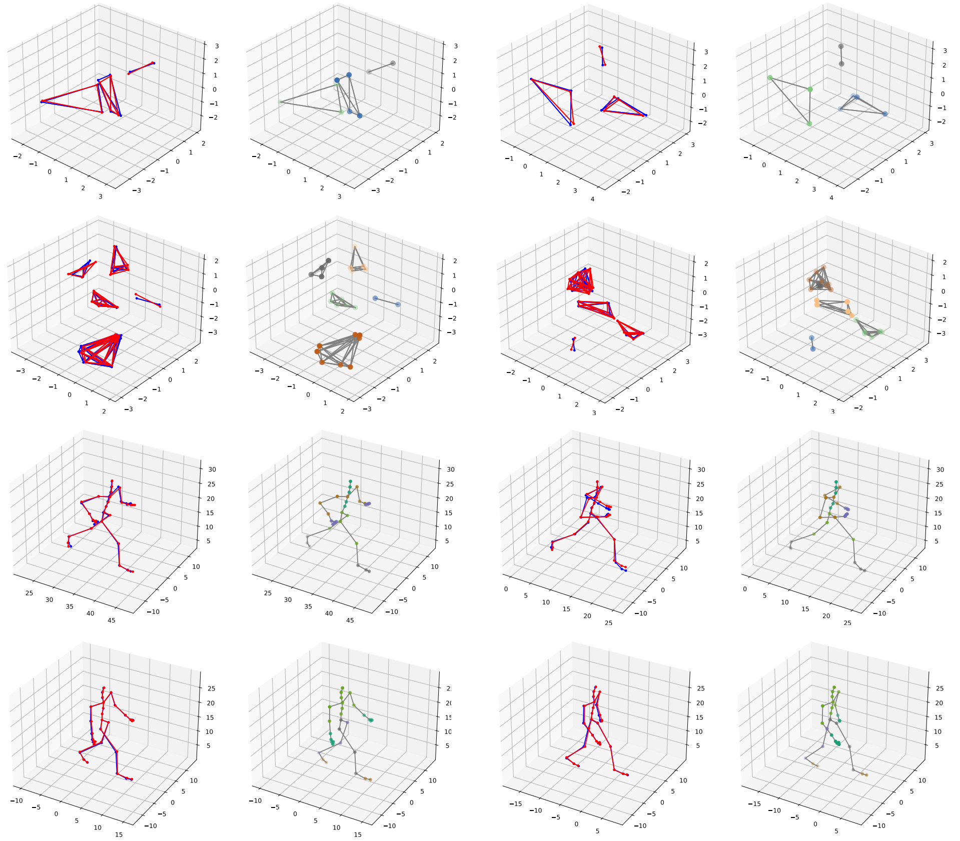

Figure 9: More visualizations and pooling results. Ground truth in red. The prediction of EGHN in blue.

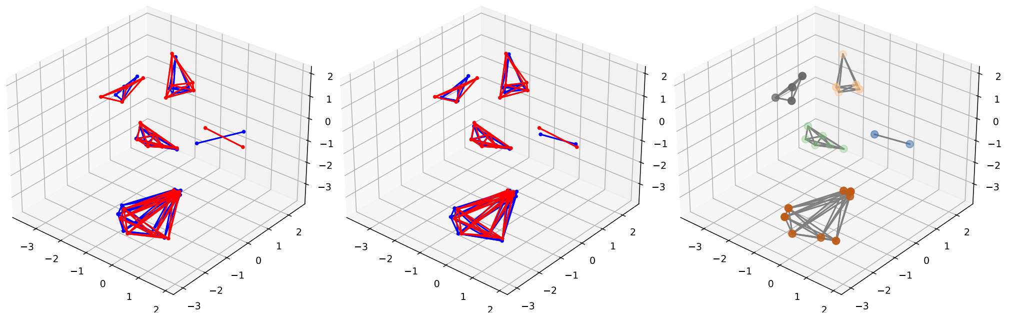

Figure 10: Visualization on M-complex dataset. *Left*: the prediction of EGNN. *Middle*: the prediction of EGHN. *Right*: the pooling results of EGHN with each color indicating a cluster. Ground truth in red, and prediction in blue. Best viewed by colour printing and zooming in.

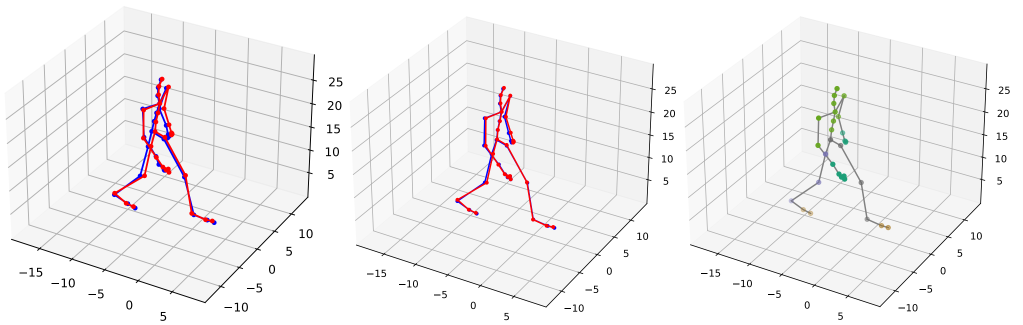

Figure 11: Visualization on Mocap Walk. *Left*: the prediction of EGNN. *Middle*: the prediction of EGHN. *Right*: the pooling results of EGHN with each color indicating a cluster. Ground truth in red, and prediction in blue. Best viewed by colour printing and zooming in.

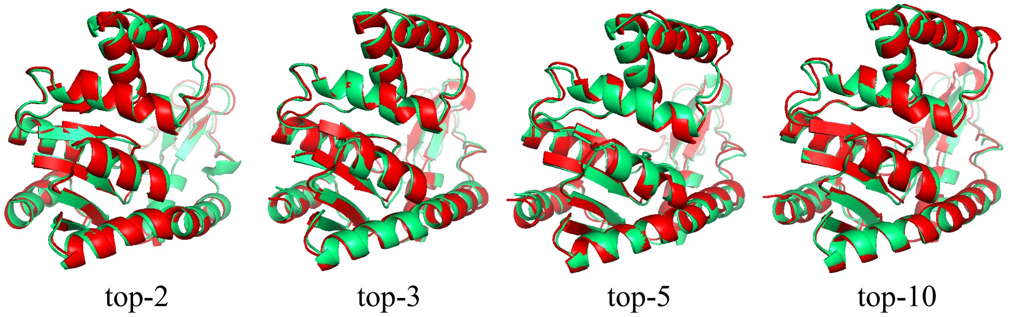

Figure 12: More visualizations on protein MD. Ground truth in red. The prediction of EGHN in green.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] In Appendix.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Appendix.
   (b) Did you include complete proofs of all theoretical results? [Yes] In Appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Yes. See Sec. 4 and Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Table 1, 2.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] In Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. 4.
   (b) Did you mention the license of the assets? [Yes] In Appendix.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]