Supplementary Materials: Lightweight equivariant model for efficient interatomic potential predictions

Ziduo Yang[1,2], Xian Wang[3], Yifan Li[1], Qiujie Lv[1,2], Calvin Yu-Chian Chen[2,4,5*], and Lei Shen[1*]

[1]*Department of Mechanical Engineering, National University of Singapore, 9 Engineering Drive 1, 117575, Singapore*

[2]*Artificial Intelligence Medical Research Center, School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen, 518107, China*

[3]*Department of Physics, National University of Singapore, 2 Science Drive 3, 117551, Singapore*

[4]*AI for Science (AI4S)-Preferred Program, Peking University Shenzhen Graduate School, Shenzhen, 518055, China and*

[5]*School of Electronic and Computer Engineering, Peking University Shenzhen Graduate School, Shenzhen, 518055, China*

(*Corresponding author(s): cy@pku.edu.cn; shenlei@nus.edu.sg)

1. METHODS

A. Problem definition

In this work, the atomic structure is represented as a 3D interaction graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{V}$ and $\mathcal{E}$ are sets of nodes and edges. $\mathcal{R}$ are sets of 3D coordinates of nodes. Each node is connected to its closest neighbors within a cutoff distance D with a maximum number of neighbors N, where D and N are predefined numbers. Each node $v_i \in \mathcal{V}$ has its scalar feature $\mathbf{x}_i \in \mathbb{R}^F$, vector feature $\vec{\mathbf{x}}_i \in \mathbb{R}^{F\times 3}$ (i.e., retaining F scalars and F vectors for each node), and 3D coordinate $\vec{\mathbf{r}}_i \in \mathbb{R}^3$. The scalar and vector features can be updated during training. We set the number of features F as a constant throughout the network. The scalar feature is initialized to an embedding only dependent on the atomic number as $\mathbf{x}_i^{(0)} = E(z_i) \in \mathbb{R}^F$, where z_i is the atomic number and E is an embedding layer that takes z_i as input and returns a F-dimensional feature. This embedding is similar to the one-hot vector but is trainable. The vector feature is set to $\vec{\mathbf{x}}_i^{(0)} = \vec{\mathbf{0}} \in \mathbb{R}^{F\times 3}$ at the initial step. We also define $\vec{\mathbf{r}}_{ij} = \vec{\mathbf{r}}_j - \vec{\mathbf{r}}_i$ as a vector from node v_i to node v_j. Norm $\|\cdot\|$ and dot product $\cdot$ are calculated along the spatial dimension, while concatenation $\oplus$ and Hadamard product $\circ$ are calculated along the feature dimension. Given a set of 3D interaction graphs, our goal is to learn a model f to predict the potentials and forces as $f(\mathcal{G}) = (e, \vec{\mathbf{F}})$, where $e \in \mathbb{R}^1$, $\vec{\mathbf{F}} \in \mathbb{R}^{M\times 3}$ and M is the number of nodes in $\mathcal{G}$.

B. Lightweight equivariant interaction graph neural network

LEIGNN iteratively updates the representation of a node through a two-phase process: message passing and message updating. During the message passing phase, a node can receive messages from its neighboring nodes, which progressively expands its accessible radius. In the message updating phase, LEIGNN only employs the message within the node itself (a node has F scalars and F vectors) to update node features (Fig. S1).

1. *Message passing phase*

In the message passing phase, a particular node v_i gathers messages from its neighbouring scalar $\mathbf{x}_j$ and vector $\vec{\mathbf{x}}_j$, resulting in intermediate scalar and vector variables $\mathbf{m}_i$ and $\vec{\mathbf{m}}_i$ as follows:

$$\mathbf{m}_i = \sum_{v_j \in \mathcal{N}(v_i)} (\mathbf{W}_h \mathbf{x}_j^{(t)}) \circ \lambda_h(\|\vec{\mathbf{r}}_{ji}\|) \tag{1}$$

$$\vec{\mathbf{m}}_i = \sum_{v_j \in \mathcal{N}(v_i)} (\mathbf{W}_u \mathbf{x}_j^{(t)}) \circ \lambda_u(\|\vec{\mathbf{r}}_{ji}\|) \circ \vec{\mathbf{x}}_j^{(t)} + (\mathbf{W}_v \mathbf{x}_j^{(t)}) \circ \lambda_v(\|\vec{\mathbf{r}}_{ji}\|) \circ \frac{\vec{\mathbf{r}}_{ji}}{\|\vec{\mathbf{r}}_{ji}\|} \tag{2}$$

Here, $\mathbf{W}_h, \mathbf{W}_u, \mathbf{W}_v \in \mathbb{R}^{F\times F}$ are learnable matrices. The functions λ_h, λ_u, and λ_v are the linear combination of Gaussian radial basis functions (RBF)[1]. For the Eqn. (2), the first term $(\mathbf{W}_u \mathbf{x}_j^{(t)}) \circ \lambda_u(\|\vec{\mathbf{r}}_{ji}\|) \circ \vec{\mathbf{x}}_j^{(t)}$ propagates directional information $\mathbf{x}_j^{(t)}$ obtained in the previous step to neighboring atoms, with $(\mathbf{W}_u \mathbf{x}_j^{(t)}) \circ \lambda_u(\|\vec{\mathbf{r}}_{ji}\|)$ acting as a gate signal to control how many signals in the previous step can be preserved. We interpret the second term as the force exerted by atom j on atom i, where $(\mathbf{W}_v \mathbf{x}_j^{(t)}) \circ \lambda_v(\|\vec{\mathbf{r}}_{ji}\|)$ is force magnitude and $\frac{\vec{\mathbf{r}}_{ji}}{\|\vec{\mathbf{r}}_{ji}\|}$ is the force direction. This is conceptually different from PaiNN's message function, which uses $\vec{\mathbf{r}}_{ij}$ instead of $\vec{\mathbf{r}}_{ji}$. Subsequently, $\sum_{v_j \in \mathcal{N}(v_i)} (\mathbf{W}_v \mathbf{x}_j^{(t)}) \circ \lambda_v(\|\vec{\mathbf{r}}_{ji}\|) \circ \frac{\vec{\mathbf{r}}_{ji}}{\|\vec{\mathbf{r}}_{ji}\|}$ represents the total force exerted on atom i and is a linear combination of forces exerted on it by all other atoms. It is noteworthy that we retain F vectors for each node, which means that the forces are calculated F times in parallel to provide more extensive vector representations. Fig. S1 (a) shows an example of how the F forces are calculated.

2. Message updating phase

The message updating phase aims to aggregate F scalars and vectors within $\mathbf{m}_i$ and $\vec{\mathbf{m}}_i$,respectively, to obtain new scalar $\mathbf{x}_i^{(t+1)}$ and new vector $\vec{\mathbf{x}}_i^{(t+1)}$. Fig. S1 (b) shows an example of how to aggregate F vectors within $\vec{\mathbf{m}}_i$. Two strategies are employed to extract the local and global structures from the $\vec{\mathbf{m}}_i$. The local structure is extracted using vector norm $\|\mathbf{V}\vec{\mathbf{m}}_i\|$ ($\mathbf{V} \in \mathbb{R}^{F\times F}$), while the global structure is obtained by projecting each vector onto a global vector. Specifically, we first define the global vector $\vec{\mathbf{m}}_{\mathcal{G}}$ as follows:

$$\vec{\mathbf{m}}_{\mathcal{G}} = \sum_{v_i \in \mathcal{G}} \big(\mathbf{W}_{sg} \mathbf{m}_i\big) \circ \big(\mathbf{W}_{vg} \vec{\mathbf{m}}_i\big) \tag{3}$$

where $\mathbf{W}_{sg} \in \mathbb{R}^{1\times F}$ and $\mathbf{W}_{vg} \in \mathbb{R}^{1\times F}$ are two learnable matrices compressing $\mathbf{m}_i$ and $\vec{\mathbf{m}}_i$ to a scalar and a three-dimensional vector, respectively. $\vec{\mathbf{m}}_{\mathcal{G}} \in \mathbb{R}^3$ is a global vector encoding the global structural information. We then use $\vec{\mathbf{m}}_{\mathcal{G}}$ as a reference coordinate system and project each local vector onto it.

$$\mathbf{s} = \big(\mathbf{W}_v \vec{\mathbf{m}}_i\big) \cdot \vec{\mathbf{m}}_{\mathcal{G}} \tag{4}$$

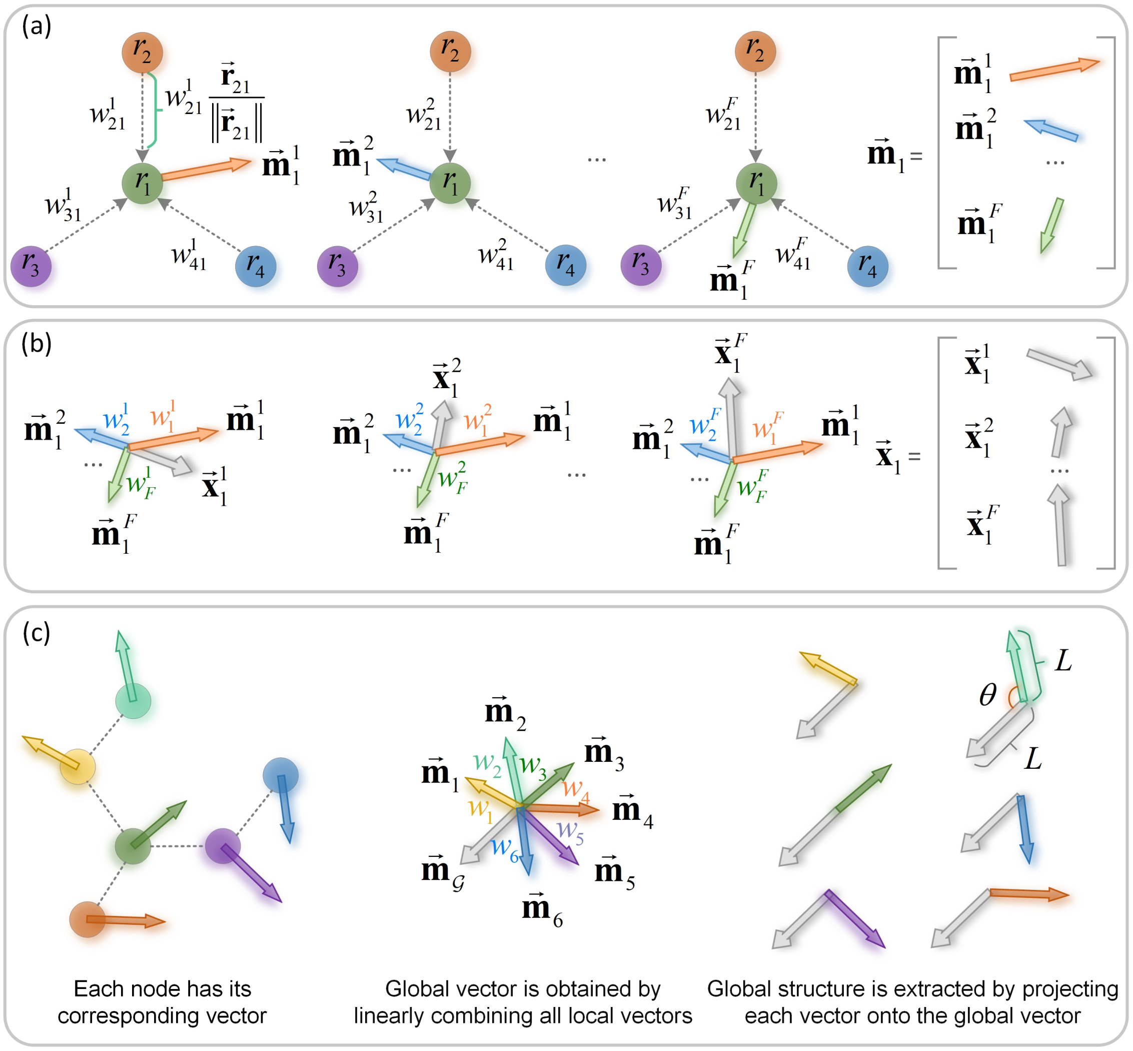
(a)
(b)
(c)
Each node has its corresponding vector
Global vector is obtained by linearly combining all local vectors
Global structure is extracted by projecting each vector onto the global vector

FIG. S1. An illustration of how vector representation evolves during message passing phase and message updating phase. (a) An example explains how to calculate the intermediate vector representation $\vec{\mathbf{m}}_1$. For the sake of simplicity, we ignore the first term of Eqn (2). The f-th vector in $\vec{\mathbf{m}}_1$ (i.e., $\vec{\mathbf{m}}_1^f$) can be interpreted as the force exerted by neighboring atom j on atom 1, where w_{j1}^f is force magnitude and $\frac{\vec{\mathbf{r}}_{j1}}{\|\vec{\mathbf{r}}_{j1}\|}$ is the force direction. Subsequently, the total force exerted on atom 1 is a linear combination of forces exerted on it by all other neighboring atoms j. Note that the forces are calculated F times in parallel. (b) The update to $\vec{\mathbf{x}}_1^f$ is achieved through a linear combination of F vectors within $\vec{\mathbf{m}}_1$. This computation is performed in parallel F times to obtain $\vec{\mathbf{x}}_1$. (c) An example explains how to project each local vector to the global vector via dot product.

where $\mathbf{W}_v \in \mathbb{R}^{F\times F}$, $\mathbf{s} \in \mathbb{R}^F$ is the extracted global structure. Therefore, by projecting each local vector $\vec{\mathbf{m}}_i$ onto the global vector $\vec{\mathbf{m}}_{\mathcal{G}}$, both the magnitude and angular information are implicitly extracted. Fig. S1 (c) shows an example of how to calculate the global vector and how to project each local vector onto it. Finally, the scalar representation $\mathbf{x}_i^{(t+1)}$ and vector representation $\vec{\mathbf{x}}_i^{(t+1)}$ is updated according to:

$$\mathbf{x}_i^{(t+1)} = \mathbf{W}_{u1}(\mathbf{m}_i \oplus \|\mathbf{V}\vec{\mathbf{m}}_i\| \oplus \mathbf{s}) + \mathbf{W}_{u2}\mathbf{m}_i \tag{5}$$

$$\vec{\mathbf{x}}_i^{(t+1)} = \big(\mathbf{W}_h(\mathbf{m}_i \oplus \|\mathbf{V}\vec{\mathbf{m}}_i\| \oplus \mathbf{s})\big) \circ \big(\mathbf{U}\vec{\mathbf{m}}_i\big) \tag{6}$$

where $\oplus$ is concatenation, $\mathbf{W}_{u1}, \mathbf{W}_h \in \mathbb{R}^{F\times 3F}$ and $\mathbf{W}_{u2}, \mathbf{U}, \mathbf{V} \in \mathbb{R}^{F\times F}$.

3. Predicting potentials and forces

To predict the potential e, we utilize an MLP layer $\phi : \mathbb{R}^F \to \mathbb{R}^1$. This layer learns atom-wise potentials $e_i \in \mathbb{R}^1$ from the scalar representation $\mathbf{x}_i^{(T)}$ which is obtained at the last graph convolution layer (referred to as the T-th layer). The total potential is then calculated as the sum of the atom-wise potentials:

$$e = \sum_{v_i \in \mathcal{G}} e_i \tag{7}$$

On the other hand, the forces are predicted using the vector representation $\vec{\mathbf{x}}_i^{(T)}$ as $\vec{\mathbf{F}} = \mathbf{W}_f \vec{\mathbf{x}}_i^{(T)}$, where $\mathbf{W}_f \in \mathbb{R}^{1\times F}$. Notably, this differs from the approaches employed by PaiNN[2] and NewtonNet[3], where forces are computed by differentiating the predicted energy with respect to the atom positions, i.e., $\vec{\mathbf{F}} = \frac{\partial e}{\partial \mathbf{r}}$. Such differentiation incurs additional computational costs.

2. PROOF OF INVARIANCE AND EQUIVALENCE

First, it can be easily verified that the predicted scalar potentials of LEIGNN exhibit invariance to translations and rotations. Specifically, the initial scalar feature $\mathbf{x}_i^{(0)}$ is invariant, and the update process of $\mathbf{x}_i^{(t)}$ solely relies on invariant operations, such as vector norm and dot product. Therefore, the predicted potentials of LEIGNN are invariant. On the other hand, the predicted vector forces are multiple linear combinations of unit vectors $\frac{\vec{\mathbf{r}}_{ji}}{\|\vec{\mathbf{r}}_{ji}\|}$, which are equivalent. A formal proof is as follows.

Let $\vec{\mathbf{g}} \in \mathbb{R}^3$ be a translation vector and $\mathbf{Q} \in \mathbb{R}^{3\times 3}$ be a unitary rotation matrix. We prove that the vector representations are invariant to the translations and equivariant to rotations. Since both translation and rotation do not affect the values of $\|\cdot\|$ or dot product, we can view them as scalars (i.e., invariant features). We then consider $\vec{\mathbf{m}}_i$ as a function of $\vec{\mathbf{x}}_i$, $\vec{\mathbf{x}}_j$ and $\vec{\mathbf{r}}_{ji}$, that is $\vec{\mathbf{m}}_i = \phi(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}_j, \vec{\mathbf{r}}_{ji})$. Note that $\vec{\mathbf{r}}_{ji} = \vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j$.

Invariance. Note that $(\vec{\mathbf{r}}_i + \vec{\mathbf{g}}) - (\vec{\mathbf{r}}_j + \vec{\mathbf{g}}) = \vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j$. Therefore, the output $\vec{\mathbf{m}}_i$ will be invariant to translation as

$$\vec{\mathbf{m}}_i = \phi\big(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}_j, (\vec{\mathbf{r}}_i + \vec{\mathbf{g}}) - (\vec{\mathbf{r}}_j + \vec{\mathbf{g}})\big) = \phi\big(\vec{\mathbf{r}}_{ij}, \vec{\mathbf{x}}_i, \vec{\mathbf{r}}_{ji}\big) \tag{8}$$

Since $\vec{\mathbf{x}}_i^{(t+1)}$ only depends on $\vec{\mathbf{m}}_i$ and scalars $\mathbf{m}_i \oplus \|\mathbf{V}\vec{\mathbf{m}}_i\| \oplus \mathbf{s}$ ($\mathbf{s}$ is global structural features and is invariant), we can conclude that the vector representations are invariant to translations.

Equivalence. Suppose $t = 0$ and note that $\vec{\mathbf{x}}_i^{(0)} = \vec{\mathbf{0}} \in \mathbb{R}^{F\times 3}$, we then have $\vec{\mathbf{m}}_i = \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{W}_v \mathbf{x}_j^{(0)} \circ \vec{\mathbf{r}}_{ji}$ (we ignore $\|\vec{\mathbf{r}}_{ji}\|$ for simplicity). Note that Hadamard product (i.e., $\circ$) can be viewed as left multiplication by a diagonal matrix $\mathbf{D}$. Therefore, we can rewrite Eqn. (6) as

$$\vec{\mathbf{x}}_i^{(1)} = \mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_v (\vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j) \tag{9}$$

where $\mathbf{D}_v = \mathbf{W}_v \mathbf{x}_j^{(t)}$ and $\mathbf{D}_h = \mathbf{W}_h(\mathbf{m}_i \oplus \|\mathbf{V}\vec{\mathbf{m}}_i\| \oplus \mathbf{s})$. Here, we consider the coordinates as a set of row vectors. Therefore, rotating the coordinates $\mathcal{R}$ can be achieved by multiplication with $\mathbf{Q}$ from the right. Thus, $\vec{\mathbf{x}}_i^{(1)}$ will be equivalent to rotation as

$$\begin{aligned}\vec{\mathbf{x}}_i^{(1)} &= \mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_v (\vec{\mathbf{r}}_i \mathbf{Q} - \vec{\mathbf{r}}_j \mathbf{Q}) \\ &= [\mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_v (\vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j)] \mathbf{Q}\end{aligned} \tag{10}$$

We can conclude $\vec{\mathbf{x}}_i^{(1)}$ is equivalent to rotation. Now suppose $\vec{\mathbf{x}}_i^{(t)}$ is already equivalent to rotation. Again, we can rewrite Eqn. (6) as

$$\vec{\mathbf{x}}_i^{(t+1)} = \mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_u \vec{\mathbf{x}}_j^{(t)} + \mathbf{D}_v (\vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j) \tag{11}$$

where $\mathbf{D}_u = \mathbf{W}_u \mathbf{x}_j^{(t)}$. Similarly, we have

$$\begin{aligned}\vec{\mathbf{x}}_i^{(t+1)} &= \mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_u \vec{\mathbf{x}}_j^{(t)} \mathbf{Q} + \mathbf{D}_v (\vec{\mathbf{r}}_i \mathbf{Q} - \vec{\mathbf{r}}_j \mathbf{Q}) \\ &= [\mathbf{D}_h \mathbf{U} \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{D}_u \vec{\mathbf{x}}_j^{(t)} + \mathbf{D}_v (\vec{\mathbf{r}}_i - \vec{\mathbf{r}}_j)] \mathbf{Q} \end{aligned} \tag{12}$$

Therefore, we can claim that the vector representations are equivalent to rotations.

3. IMPLEMENTATION DETAILS

The LEIGNN model is implemented using PyTorch. Experiments are conducted on an NVIDIA GeForce RTX A4000 with 16 GB of memory. The training objective aims to minimize the loss function defined as:

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \left(\alpha |e_n - e_n^l| + \beta \frac{1}{3M} \sum_{m=1}^{M} \sum_{k=1}^{3} |\vec{\mathbf{F}}_{nmk} - \vec{\mathbf{F}}_{nmk}^l| \right) \tag{13}$$

where e_n^l represents the ground truth energy of n-th sample, $\vec{\mathbf{F}}_{nmk}^l$ is the ground truth force of k-th dimension of m-th atom in n-th sample. The variables N and M denote the sample size and the number of atoms in each sample, and α and β denote the weights assigned to the energy and force losses, respectively.

For other baselines, we employ the recommended hyperparameters provided by the OCP`https://github.com/Open-Catalyst-Project/ocp` and MDsim`https://github.com/kyonofx/MDsim` source codes for the OC20 and MD17 datasets, respectively.

4. DATASETS FOR MD SIMULATIONS

1. *Solid LiPS*

In this simulation, we use $\mathrm{Li}_{6.75}\mathrm{P}_3\mathrm{S}_{11}$, a crystalline superionic lithium conductor widely used in battery development. The simulation cell comprises 83 atoms and serves as a representative system for studying kinetic properties in materials through MD simulations. We utilize the dataset from Batzner et al.'s work[4]. This AIMD trajectory consists of a total of 25,000 structures, with 19,000 randomly sampled structures used for training, 1,000 for validation, and the remainder for testing. The simulation employs a time step of 0.25 fs and a temperature of 520 K, with a Nosé–Hoover thermostat.

2. *Liquid* H_2O

Water, essential in biological and chemical processes, presents complexities in thermodynamics and phase behavior that make it challenging to simulate. For this system, our dataset comprises 100,000 structures. Of these, 72,000 structures are randomly sampled for training, 8,000 for validation, and the remaining 20,000 for testing. The simulation is set with a time step of 0.5 fs, a temperature of 300 K, and uses a Langevin thermostat. The duration of the simulation is 50 ps.

3. *Gas* CH_4

Methane is a potent greenhouse gas with significant implications for climate change and serves as the primary component of natural gas, a vital global energy source. Its molecular dynamics properties are crucial for understanding environmental impacts, energy applications, and broader scientific phenomena. For methane, our dataset is made up of 100,000 structures. From this, 72,000 structures are randomly sampled for training, 8,000 for validation, and the remaining 20,000 for testing. The simulation adopts a time step of 0.5 fs, a temperature of 300 K, and a Langevin thermostat. The simulation's duration is set at 50 ps.

The test results for the three systems, measured in terms of energy and forces MAE, can be found in Table S1.

TABLE S1. Performance of LEIGNN on test sets for the LiPS, H2O, and CH4 systems, evaluated in terms of energy MAE (meV) and forces MAE (meV/Å).

Dataset	LiPS	H2O	CH4
Energy	3.58	11.08	6.04
Forces	1.47	3.47	1.55

4. LEIGNN for MD simulations

We incorporate LEIGNN-based forces with the Atomic Simulation Environment (ASE)[5] to perform MD simulations. For the LiPS system, we use a Nosé-Hoover thermostat, while for water and methane, we opt for a Langevin thermostat. The parameters are consistent with those from the AIMD run.

5. PERFORMANCE INDICATORS

Mean absolute error (MAE). The performance of energy and force predictions is quantified using MAE, calculated as follows:

$$\mathrm{MAE}(e) = \frac{1}{N}\sum_{n=1}^{N} |e_n - e_n^l| \tag{14}$$

$$\mathrm{MAE}(\vec{\mathbf{F}}) = \frac{1}{3NM}\sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{k=1}^{3} |\vec{\mathbf{F}}_{nmk} - \vec{\mathbf{F}}^l_{nmk}| \tag{15}$$

Distribution of interatomic distances. The distribution of interatomic distances ($h(r)$) offers a concise representation of a system's 3D structure and has been explored in prior research [6]. For a specific configuration denoted as x, $h(r)$ is determined using the equation:

$$h(r) = \frac{1}{N(N-1)}\sum_{i}^{N}\sum_{j\neq i}^{N} \delta(r - \|x_i - x_j\|) \tag{16}$$

In this equation, r represents the distance from a reference particle, while N denotes the total particle count. The indices i and j identify the atom pairs contributing to the distance statistics. The function δ is the Dirac Delta function, employed to extract value distributions. To determine the ensemble average, $h(r)$ is computed and then averaged across frames.

Radial distribution function (RDF). RDF, a valuable simulation observable, characterizes the structural and thermodynamic properties of the system. By its definition, the RDF elucidates the variation of density as a function of distance from a reference particle. The RDF for a particular configuration x is computed as follows:

$$\mathrm{RDF}(r) = \frac{1}{4\pi r^2}\frac{1}{N\rho}\sum_{i=1}^{N}\sum_{j\neq i}^{N} \delta(r - \|x_i - x_j\|) \tag{17}$$

In this equation, r represents the distance from a reference particle, N is the total number of particles, and i, j are indices referring to the atom pairs contributing to distance statistics. Additionally, ρ denotes the system's density, and δ is the Dirac Delta function, applied for value distribution extraction. The ensemble average is calculated by averaging RDF(r) over frames. The final RDF MAE is obtained by integrating over r:

$$\text{MAE(RDF)} = \int_{r=0}^{\infty} |\langle \text{RDF}(r) \rangle - \langle \hat{\text{RDF}}(r) \rangle| dr \tag{18}$$

where $\langle \cdot \rangle$ indicates the averaging operator, $\langle \text{RDF}(r) \rangle$ is the reference equilibrium RDF, and $\langle \hat{\text{RDF}}(r) \rangle$ is the RDF predicted by the model.

Stability criterion. In the study, stability is defined as the ability to retain low-energy configurations[6]. For systems with periodic boundary conditions, we monitor stability by tracking equilibrium statistics. We denote a simulation as 'unstable' at time T if

$$\int_{r=0}^{\infty} \| \langle \text{RDF}(r) \rangle - \langle \hat{\text{RDF}}(r) \rangle_{t=T}^{T+\tau} \| dr > \Delta \tag{19}$$

where τ refers to a short time window and Δ represents the stability threshold. We set $\Delta = 1.0$, which is the same as in MDsim[6]. The stability of a model is subsequently assessed based on how long it maintains stability during the simulation.

TABLE S2. Comparison results of the proposed LEIGNN and baselines on S2EF task of four external validation sets of OC20 in terms of energy MAE (meV) and forces MAE (meV/Å), where all models are trained on OC20-200K.

Model	ID		OOD Ads.		OOD Cat.		OOD Both	
	Energy	Forces	Energy	Forces	Energy	Forces	Energy	Forces
CGCNN	1111	75.0	1261	80.7	1097	74.1	1383	91.9
SchNet	975	59.6	1077	66.8	975	59.5	1204	77.9
PaiNN	493	53.2	614	60.3	530	53.3	749	70.3
DimeNet++	497	48.7	547	55.9	522	48.6	671	65.3
GemNet-dT	443	**41.3**	**516**	**46.7**	548	**41.6**	717	**55.3**
LEIGNN	**436**	46.6	541	52.5	**467**	46.8	**636**	62.3

6. MODEL PERFORMANCE ON OC20-200K DATASET

For the OC20-200K dataset, GemNet-dT demonstrates superior performance in force prediction, with LEIGNN coming in second, as indicated in Table S2. However, for energy prediction, LEIGNN still excels beyond other benchmark models, except in the OOD Ads. scenario. Overall, our results indicate that LEIGNN's implicit consideration of structural features can yield competitive performance compared to DimeNet++ and GemNet-dT which explicitly incorporate angular features.

7. MODEL PERFORMANCE ON MD17 DATASET

The MD17 dataset comprises eight molecular dynamics simulations featuring small organic molecules[7]. Each of these simulations represents a unique trajectory for a single molecule, covering a broad range of conformations. The purpose of this task is to predict energies and forces corresponding to each of these trajectories. We opt for a training set of 19,000 entries, a validation set of 1,000 entries, and a test set of 10,000 entries. It's worth emphasizing that the same sets are utilized across all model training, validation, and testing phases. Table S3 presents the comparative results of the models. LEIGNN demonstrates superior performance to DimeNet++ across all 16 targets, and outperforms or matches GemNet-dT on 8 out of the 16 targets. This indicates that models based on two-body interactions, such as LEIGNN, are able to compete with models based on many-body interactions, such as DimeNet++ and GemNet-dT. These findings attest to the efficacy of LEIGNN, even when compared with more complex models based on many-body interactions.

8. MODEL PERFORMANCE ON ISO17 DATASET

The ISO17 dataset constitutes a collection of MD trajectories simulated for 129 distinct organic isomers[1]. Despite all isomers having the same C7O2H10 composition, they each have a unique structure. Each isomer's trajectory consists of 5,000 snapshots, yielding a total of 645,000 unique snapshots. These snapshots are divided into three non-overlapping sets according to the original work[1]: a training set and two testing sets. The training set contains 80% of the randomly selected snapshots from the MD trajectories of 80% of the 129 isomers. The first test set, termed as the 'test within' set, contains the remaining 20%

TABLE S3. Mean absolute errors on MD17 dataset for energy and force predictions in terms of energy MAE (meV) and forces MAE (meV/Å), respectively. In each cell, the errors are written in energy MAE and forces MAE pairs.

	CGCNN	SchNet	PaiNN	DimeNet++	GemNet-dT	LEIGNN
Aspirin	144.7, 610.4	12.1, 21.3	19.7, 3.4	17.8, 5.5	9.9, 3.8	7.9, 3.2
Benzene	7.4, 67.8	3.9, 12	80.0, **5.9**	10, 8.3	**2.8**, 6.5	3.3, 7.3
Ethanol	24.1, 200.5	2.2, 4.6	9.0, 2.4	9.7, 2.0	**2.2**, **1.3**	3.1, 1.8
Malonaldehyde	44.5, 338.4	3.7, 8.2	9.3, 3.1	11.4, 3.0	3.9, **2.4**	**3.9**, 2.8
Naphtalene	34.9, 170.5	9.9, 11.5	8.1, 1.3	10.5, 2.5	**2.8**, 1.3	3.0, **0.9**
Salicylic acid	40.7, 198.4	8.1, 14.9	11.6, 2.2	16.9, 5.3	5.8, 3.2	**5.1**, **2.0**
Toluene	26.4, 153.8	6.0, 9.9	7.2, 1.4	9.7, 2.2	**2.8**, 1.3	3.2, **1.1**
Uracil	28.2, 183.6	5.3, 10.6	6.5, 1.4	11.3, 2.4	3.2, 1.7	**2.7**, **1.2**

of snapshots drawn from the same molecules used in the training set. The second, more challenging, test set comprises all snapshots from the other 20% of the 129 molecules, not included in the training set, and is referred to as the 'test other' set. The first test set aims to gauge the capability of GNNs to interpolate the forces of unfamiliar geometries for known molecules, i.e., molecules present in the training data. In contrast, the second test set evaluates LEIGNN's proficiency at generalizing to unfamiliar molecules, i.e., molecules never encountered during training. A validation set, including 4K examples, is utilized to select the optimal model for testing. We compare LEIGNN with the originally reported results of SchNet[1] and GNNFF[8]. Note that the training, validation, and test sets are the same for the three methods. The performance is measured based on energy MAE and forces MAE.

Table S4 summarizes the test results. In the 'test within' scenario, LEIGNN achieves an energy MAE of 0.004, representing a substantial improvement of 75% over SchNet's performance. Similarly, the forces MAE of LEIGNN is 0.003, marking an impressive improvement of 91.67% over the previous best model, GNNFF. In the more challenging 'test other' scenario, LEIGNN achieves an energy MAE of 0.050, reflecting an enhancement of 51.92% compared to SchNet's performance. Moreover, LEIGNN's forces MAE is 0.029, which is

67.05% superior to the performance of GNNFF. These results demonstrate that LEIGNN significantly boosts performance in both in-domain and out-of-domain scenarios compared to the previous best models.

TABLE S4. Comparison among SchNet, GNNFF, and LEIGNN in terms of energy MAE (eV) and forces MAE (eV/Å) on ISO17 database. In each cell, the errors are written in energy MAE and forces MAE pairs. "-" indicates lack of data.

	SchNet	GNNFF	LEIGNN
Test within	0.016, 0.043	-, 0.036	**0.004**, **0.003**
Test other	0.104, 0.095	-, 0.088	**0.050**, **0.029**

9. ABLATION STUDY

To demonstrate the effectiveness of our innovative strategy for extracting global structure from the vector representation, we compare LEIGNN with a baseline named LEIGNN_noG which does not incorporate the global structure. As shown in Table S5, the superior performance of LEIGNN in energy prediction can largely be attributed to its integration of global structures. Comparisons between LEIGNN and LEIGNN_noG illustrate that LEIGNN significantly surpasses LEIGNN_noG in energy prediction. This result is reasonable as the process of extracting global structures from vector representation involves converting vectors to scalars. Energy prediction relies heavily on scalar representation, which could explain why LEIGNN exhibits substantial improvement in energy prediction, while improvements in force prediction are modest.

10. MOLECULAR DYNAMICS SIMULATIONS

In Fig. S2, we compare the element-specific probability distribution function (PDF) of force magnitudes between the LEIGNN MD run and the DFT run. The PDFs of LEIGNN and DFT demonstrate significant agreement for all element types. This indicates that LEIGNN successfully captures the force magnitudes, achieving substantial consistency with

TABLE S5. Comparing LEIGNN with LEIGNN_noG on the OC20-50K and OC20-200K datasets.

Dataset	Models	ID		OOD Ads.		OOD Cat.		OOD Both	
		Energy	Forces	Energy	Forces	Energy	Forces	Energy	Forces
OC20-50K	LEIGNN_noG	561	56.5	710	61.6	607	56.0	874	72.9
	LEIGNN	**543**	**55.4**	**655**	**60.2**	**585**	**54.9**	**778**	**71.0**
	Improvement	3.20%	1.95%	7.75%	2.27%	3.62%	1.96%	10.98%	2.61%
OC20-200K	LEIGNN_noG	451	47.3	565	53.4	511	47.4	706	63.3
	LEIGNN	**436**	**46.6**	**541**	**52.5**	**467**	**46.8**	**636**	**62.3**
	Improvement	3.33%	1.48%	4.25%	1.68%	8.61%	1.27%	9.92%	1.58%

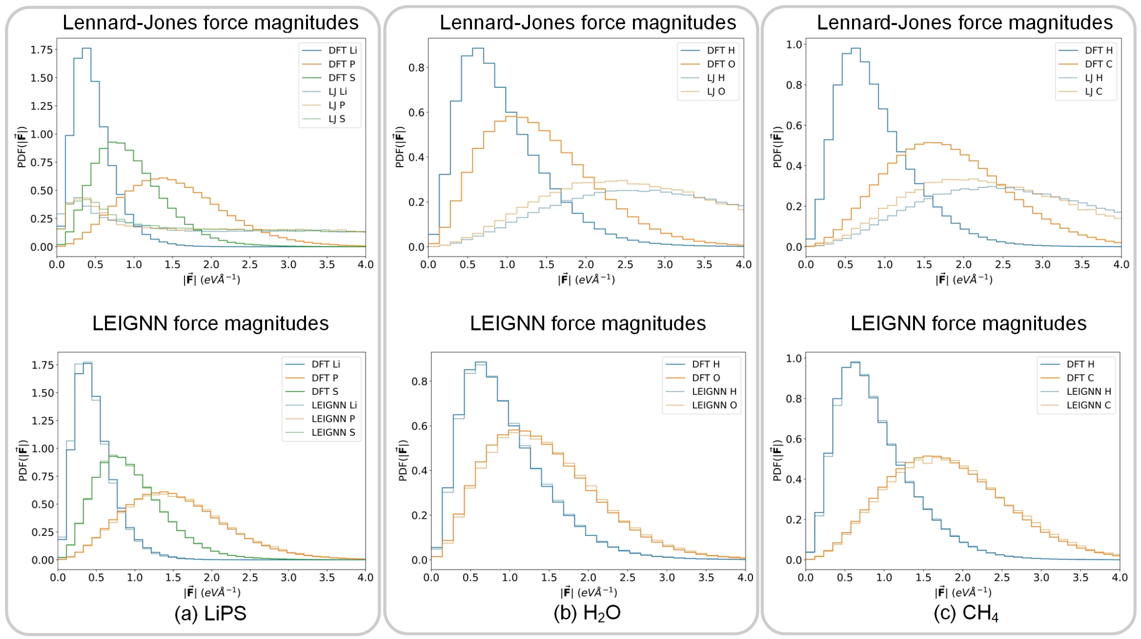

FIG. S2. Probability distribution function (PDF) of force magnitudes for (a) LiPS, (b) H2O, and (c) CH4

DFT predictions.

[1] K. Schütt, P.-J. Kindermans, H. E. Sauceda Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, Schnet: A continuous-filter convolutional neural network for modeling quantum interactions,

Advances in neural information processing systems **30** (2017).

[2] K. Schütt, O. Unke, and M. Gastegger, Equivariant message passing for the prediction of tensorial properties and molecular spectra, in *International Conference on Machine Learning* (PMLR, 2021) pp. 9377–9388.

[3] M. Haghighatlari, J. Li, X. Guan, O. Zhang, A. Das, C. J. Stein, F. Heidar-Zadeh, M. Liu, M. Head-Gordon, L. Bertels, *et al.*, Newtonnet: A newtonian message passing network for deep learning of interatomic potentials and forces, Digital Discovery **1**, 333 (2022).

[4] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky, E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials, Nature communications **13**, 2453 (2022).

[5] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, *et al.*, The atomic simulation environment—a python library for working with atoms, Journal of Physics: Condensed Matter **29**, 273002 (2017).

[6] X. Fu, Z. Wu, W. Wang, T. Xie, S. Keten, R. Gomez-Bombarelli, and T. S. Jaakkola, Forces are not enough: Benchmark and critical evaluation for machine learning force fields with molecular simulations, Transactions on Machine Learning Research (2023).

[7] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller, Machine learning of accurate energy-conserving molecular force fields, Science advances **3**, e1603015 (2017).

[8] C. W. Park, M. Kornbluth, J. Vandermause, C. Wolverton, B. Kozinsky, and J. P. Mailoa, Accurate and scalable graph neural network force field and molecular dynamics with direct force architecture, npj Computational Materials **7**, 73 (2021).