Accurate implementation of DeepMD-v2 potential calculation in HALMD for single species, extension to multi-species and Automatic-Differentiation-Based Force Computation

Sandip Kumar Sah

M.Sc. Computational Science, Freie Universität Berlin

January 1, 2026

Supervisor: Prof. Dr. Felix Höfling

Content I

Motivation I

- Classical potentials are efficient but inaccurate for complex chemistry, while *ab initio* methods are accurate but computationally expensive [5, 6].
- Machine-learned potentials (e.g., DeepMD) achieve near–DFT accuracy at classical MD cost [1, 8].
- Current DeepMD–HALMD integration is limited to single-species systems [4].
- This thesis enables scalable, accurate multi-species DeepMD simulations in HALMD.

Objectives and Scope

- Extend the DeepMD integration in HALMD beyond the single-species implementation of Cruz to full **multi-species DeepMD-v2** inference [4].
- Implement the missing core components of the DeepMD-v2 pipeline:
  - periodic coordinate handling, ghost atoms, and multi-type neighbor lists,
  - species-dependent filter and descriptor networks,
  - normalization, scaling, smooth cutoffs, and accurate force backpropagation

  [8, 9].
- Focus on the **inference stage** (energy and force evaluation) for multi-component materials using the DeepPot-SE descriptor; model training and alternative descriptors are outside the scope.

HALMD: High-Accuracy Large-Scale MD Engine

- HALMD is a high-performance, open-source MD framework for large-scale simulations, optimized for numerical accuracy and GPU acceleration [3].
- Atomic motion follows Newton's equations of motion:

$$m_i \frac{d^2 \mathbf{r}_i}{dt^2} = -\nabla_i U(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N),$$

- Traditionally uses efficient empirical interatomic potentials, which are limited for complex chemistry [5, 2].
- This thesis extends HALMD with multi-species DeepMD-v2 force fields, enabling near *ab initio* accuracy at classical MD cost [8, 9].

Neural Network Potentials and Deep Potential MD(DeepMD)-v2

- **Neural Network Potentials** learn interatomic interactions from *ab initio* data, achieving near–DFT accuracy at classical MD cost [1, 7].
- Energy is written as a sum of atomic contributions:

$$E = \sum_i E_i(\mathcal{R}_i),$$

  ensuring locality and linear scaling [1].
- **DeepMD-v2**: modern NNP with smooth, symmetry-preserving descriptors, explicit multi-species support, and efficient GPU execution [8, 9].
- Species-specific embedding and fitting networks predict energies; forces follow from

$$\mathbf{F}_i = -\frac{\partial E}{\partial \mathbf{r}_i}.$$

Overview of Implementation

- DeepMD-v2 is integrated into HALMD through a modular, end-to-end inference pipeline.
- Model parameters are extracted from `frozen_model.pb` and `input.json` and converted into a compact HDF5 format for efficient loading.
- HALMD reconstructs local atomic environments using the exact DeepMD-v2 convention:
  - periodic coordinate wrapping,
  - ghost-cell extension,
  - species-ordered neighbor selection.
- From the reconstructed environment, HALMD computes descriptors and evaluates species-specific neural networks to obtain atomic energies and forces.

DeepMD–HALMD Energy Evaluation Pipeline

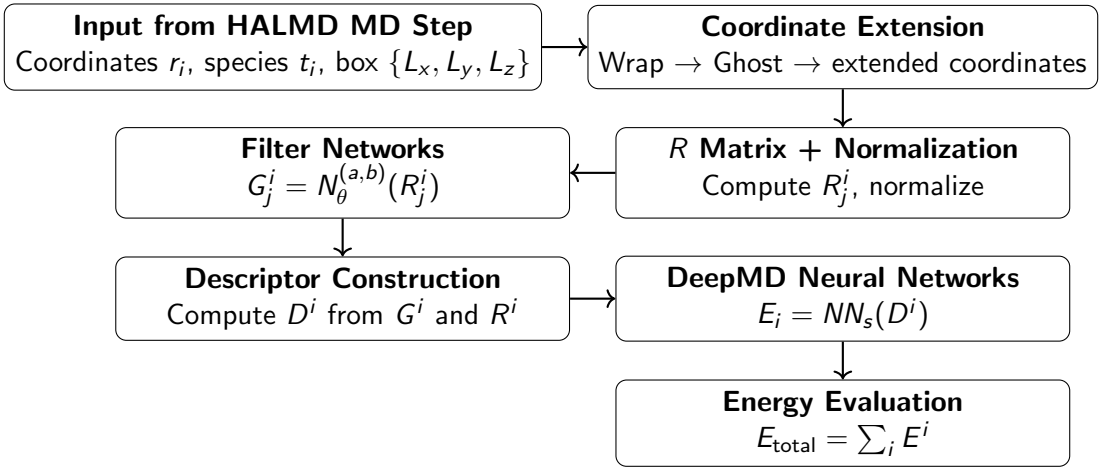

Figure: DeepMD–HALMD energy evaluation pipeline implemented in this thesis.

Model Parameter Extraction

- DeepMD-v2 models are stored as `frozen_model.pb` with hyperparameters in `input.json` [8, 9].
- All parameters are extracted from the TensorFlow graph and converted into a compact HDF5 format for efficient inference in HALMD.
- The extracted model contains:
  - descriptor attributes (cutoffs, normalization, species mapping),
  - species-dependent embedding (filter) networks,
  - species-dependent fitting networks,
  - final outputs: energies, forces, and virials.
- The hierarchical HDF5 model enables full multi-species DeepMD-v2 inference in HALMD, without TensorFlow, extending earlier single-species implementations [4].

Coordinate System Extension

- **Problem:** HALMD's native neighbor handling uses the *minimum-image convention*, sufficient for classical MD, but incompatible with DeepMD's descriptor construction [3, 8, 9].
- **DeepMD Requirement:** Descriptors are computed from *absolute atomic coordinates* inside the primary cell, requiring explicit:
  - coordinate wrapping,
  - ghost-cell periodic expansion,
  - species-aware neighbor organization,
  - descriptor normalization.
- **Implemented Solution:** A new DeePMD-style environment constructor that exactly reproduces the DeepMD-v2 preprocessing pipeline, replacing HALMD's original environment construction [4].
- **Impact:** Enables correct descriptor evaluation and full compatibility with multi-species DeepMD-v2 models.

Explicit Coordinate Wrapping

- DeepMD requires all atoms to be mapped into the primary cell using fractional coordinates:

$$\mathbf{s}_i = \mathbf{r}_i \mathbf{H}^{-1}, \qquad \tilde{\mathbf{s}}_i = \mathbf{s}_i - \lfloor \mathbf{s}_i \rfloor, \qquad \tilde{\mathbf{r}}_i = \tilde{\mathbf{s}}_i \mathbf{H}.$$

- This procedure is mandatory for general (non-orthorhombic) simulation cells and matches DeepMD's internal pipeline [8].

Explicit Coordinate Wrapping visualization

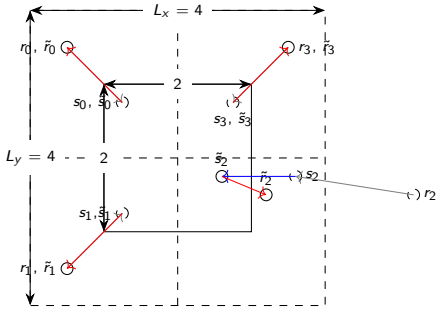

Figure: Explicit coordinate wrapping illustrated in stages: real coordinates r_i, fractional coordinates $s_i = \mathbf{r}_i \mathbf{H}^{-1}$, and wrapped fractional coordinates $\tilde{s}_i$.

Ghost-Cell Periodic Extension

- **Problem:** HALMD's neighbor list applies the minimum-image convention, which is sufficient for classical MD but does *not* provide the full periodic environment required by DeepMD descriptors [3, 8, 9].
- **DeepMD Requirement:** Every atom must retain a complete neighborhood within the cutoff radius r_c, including neighbors across periodic boundaries.
- **Implemented Solution:** Full **ghost-cell tiling** — explicit replication of the simulation cell in all spatial directions before neighbor construction.
- **Impact:** Ensures descriptor correctness and identical environments for atoms near boundaries and in the interior, enabling full DeepMD-v2 compatibility.

Ghost-Cell Construction

- Periodic images are generated from wrapped coordinates:

$$\mathbf{r}_i^{(\mathrm{s})} = \tilde{\mathbf{r}}_i + s_x L_x\,\mathbf{e}_x + s_y L_y\,\mathbf{e}_y + s_z L_z\,\mathbf{e}_z, \quad s_\alpha \in [-n_{\mathrm{buff},\alpha},\, n_{\mathrm{buff},\alpha}]$$

- Buffer size ensures full cutoff coverage:

$$n_{\mathrm{buff},\alpha} = \left\lceil \frac{r_c}{L_\alpha} \right\rceil$$

- For cubic cells with $n_{\mathrm{buff}} = 1$: $3^3 = 27$ replicated cells.

Ghost-Cell construction visualization

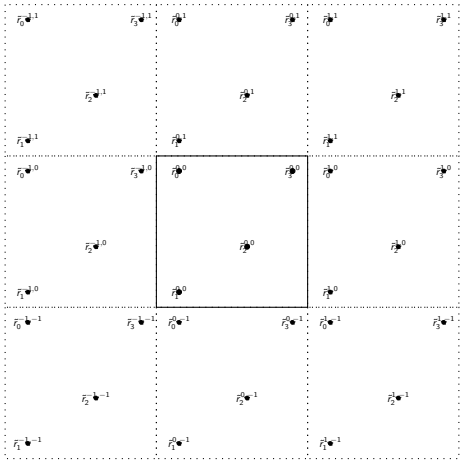

Figure: Periodic replication of the computational cell.

Construction of the Embedding Matrix G I

- The DeepPot-SE descriptor maps each normalized inverse distance

$$\hat{s}_{ij} = \hat{R}^i_{j,0}$$

to a learned embedding vector using a species-pair–specific filter network.

- For each central–neighbor species pair (a, b), DeepMD-v2 defines a dedicated neural network

$$N_\theta^{(a,b)} : \mathbb{R} \to \mathbb{R}^M,$$

producing the embedding row

$$G^i_{j:} = N_\theta^{(a,b(j))}(\hat{s}_{ij}).$$

- All embedding rows are stacked to form

$$G^i \in \mathbb{R}^{N_c^{(\text{total})} \times M}, \quad N_c^{(\text{total})} = \sum_b N_c^{(b)}.$$

Construction of the Embedding Matrix G II

- Different neighbor species use different filter networks, enabling chemically accurate multi-species modeling.
- HALMD stores the full network family as

$$\texttt{networks}[a][b] \equiv N_\theta^{(a,b)},$$

and selects the correct one at runtime for each neighbor.

Construction of the Embedding Matrix G I

- The DeepPot-SE descriptor converts the normalized inverse distance

$$\hat{s}_{ij} = \hat{R}^i_{j,0}$$

into a learned embedding using species-pair filter networks.

- For each central–neighbor species pair (a, b), DeepMD-v2 defines a dedicated MLP

$$N_\theta^{(a,b)} : \mathbb{R} \to \mathbb{R}^M,$$

producing embedding rows

$$G^i_{j:} = N_\theta^{(a,b(j))}(\hat{s}_{ij}).$$

- Stacking all embedding rows yields

$$G^i \in \mathbb{R}^{N_c^{(\text{total})} \times M}, \qquad N_c^{(\text{total})} = \sum_b N_c^{(b)}.$$

Construction of the Embedding Matrix G II

- Different neighbor species use different networks even for identical geometry — a key mechanism for accurate multi-species modeling.
- HALMD stores all filter networks as `networks`$[a][b]$ and selects them dynamically at runtime.

Descriptor Computation I

- The DeepPot-SE descriptor combines geometric information $\hat{R}^i$ and learned embeddings G^i to form the final descriptor D^i used to predict atomic energy.
- After neighbor construction, the normalized geometric matrix is

$$\hat{R}^i \in \mathbb{R}^{N_c^{(\text{total})} \times 4},$$

  containing inverse distances and angular components.
- Each row is embedded via a species-pair neural network:

$$G^i_{j:} = N_\theta^{(a,b(j))}(\hat{s}_{ij}), \qquad G^i \in \mathbb{R}^{N_c^{(\text{total})} \times M}.$$

- The quadratic descriptor is then constructed as

$$D^i = \frac{1}{(N_c^{(\text{total})})^2} \left((G^i)^T \hat{R}^i\right)\left((\hat{R}^i)^T G^{i,\text{trunc}}\right).$$

Descriptor Computation II

- Flattening D^i produces the final descriptor vector

$$\mathbf{D}^i \in \mathbb{R}^{MM'},$$

which is passed to the fitting network to predict energy and forces.

Potential Energy Evaluation I

- DeepMD expresses the total energy as a sum of atomic energies:

$$E_{\text{tot}} = \sum_{i=1}^{N} E^i.$$

- For each atom i of species s_i, the atomic energy is obtained from a **species-dependent fitting network**:

$$E^i = \text{NN}_{s_i}(\mathbf{D}^i) + b_{s_i}.$$

- DeepMD-v2 defines one fitting network per species; HALMD reconstructs all networks directly from the frozen TensorFlow graph.
- The descriptor vector $\mathbf{D}^i$ is guaranteed to match the exact structure expected by each fitting network (ordering, dimension, normalization).
- The new HALMD implementation extends previous work from single-species to full **multi-species** DeepMD-v2 inference.

Potential Energy Evaluation II

- HALMD reproduces DeepMD-v2 energies to floating-point precision for monoatomic and multi-component systems.

Overview

- In DeepMD, atomic forces are obtained from the energy by

$$\mathbf{F}_i = -\frac{\partial E_{\text{tot}}}{\partial \mathbf{r}_i}.$$

- The total energy depends on atomic coordinates through multiple stages:

$$\mathbf{r} \;\to\; R \;\to\; G \;\to\; D \;\to\; E.$$

- Force evaluation therefore requires a structured application of the chain rule:

$$\frac{\partial E}{\partial \mathbf{r}} = \frac{\partial E}{\partial D}\frac{\partial D}{\partial G}\frac{\partial G}{\partial R}\frac{\partial R}{\partial \mathbf{r}}.$$

- HALMD reproduces the DeepMD-v2 derivative pipeline by combining:
  - automatic differentiation for all neural networks,
  - analytic derivatives for geometric and descriptor operations.
- This yields forces that match DeepMD-v2 outputs to floating-point precision.

Automatic Differentiation (AD) I

- Force computation in DeepMD requires derivatives through geometry, embedding networks, descriptor construction, and fitting networks.
- Analytic differentiation is feasible for geometric operations, but neural networks contain nonlinear activations and residual connections — making analytic derivatives impractical.
- HALMD therefore uses **automatic differentiation (AD)** for all neural-network components.
- **Current HALMD implementation: Forward-mode AD**
  - Efficient when the number of inputs is small.
  - Used for embedding networks:

$$\hat{s}_{ij} \to G_j^i, \quad \frac{\partial G_j^i}{\partial \hat{s}_{ij}}.$$

  - Also used to compute $\partial E_i / \partial D_{i,\alpha}$ in the fitting network.
- **Future improvement: Reverse-mode AD**
  - Optimal for scalar outputs with many inputs.

Automatic Differentiation (AD) II

- Would greatly accelerate fitting-network differentiation.

- HALMD uses a **hybrid approach**:
  - analytic derivatives for geometry,
  - AD for neural networks.

Derivatives of the Geometry Matrix R

- Forces require derivatives of each geometry row

$$\frac{\partial R_{j\alpha}^{(i)}}{\partial \mathbf{r}_i}.$$

- Each row depends on the relative displacement

$$\mathbf{r}_{ij} = \mathbf{r}_j - \mathbf{r}_i, \quad r_{ij} = \|\mathbf{r}_{ij}\|.$$

- Geometry features:

$$R_{j0}^{(i)} = s(r_{ij}), \qquad R_{j\alpha}^{(i)} = s(r_{ij})\frac{d_\alpha}{r_{ij}}.$$

- All derivatives follow directly from the multivariate chain rule.

Derivative of the Embedding Matrix G

- Each embedding row is produced by a species-dependent network:

$$G_{j:}^{(i)} = N_\theta^{(a,b)}(\hat{s}_{ij}), \qquad \hat{s}_{ij} = \hat{R}_{j0}^{(i)}.$$

- Therefore, $G^{(i)}$ depends on geometry only through the single scalar $\hat{s}_{ij}$.
- Its derivative reduces to a one–dimensional neural-network derivative:

$$\frac{\partial G_{jp}^{(i)}}{\partial \hat{R}_{j0}^{(i)}} = \frac{d\, N_{\theta,p}^{(a,b)}}{d\hat{s}}(\hat{s}_{ij}), \qquad p = 1, \ldots, M.$$

- All other geometric components do **not** affect G:

$$\frac{\partial G_{jp}^{(i)}}{\partial \hat{R}_{j\alpha}^{(i)}} = 0, \qquad \alpha \neq 0.$$

Descriptor Structure

- The descriptor for atom i is built from

$$G^{(i)} \in \mathbb{R}^{N_c \times M}, \qquad \hat{R}^{(i)} \in \mathbb{R}^{N_c \times 4}.$$

- Intermediate contraction:

$$C^{(i)} = \frac{1}{N_c} (G^{(i)})^{\mathsf{T}} \hat{R}^{(i)}, \qquad C^{(i)} \in \mathbb{R}^{M \times 4}.$$

- Quadratic (permutation-invariant) descriptor:

$$D^{(i)}_{\text{ext}} = C^{(i)} (C^{(i)})^{\mathsf{T}}.$$

- Only first M' columns are used:

$$D^{(i)} = D^{(i)}_{\text{ext}}[:, 1 : M'].$$

Descriptor Derivative: Key Results

- Component form of the quadratic descriptor:

$$(D_{\text{ext}}^{(i)})_{mn} = \sum_{\alpha=1}^{4} C_{m\alpha}^{(i)} C_{n\alpha}^{(i)}.$$

- Derivative w.r.t. intermediate matrix:

$$\frac{\partial (D_{\text{ext}}^{(i)})_{mn}}{\partial C_{p\beta}^{(i)}} = \delta_{mp} C_{n\beta}^{(i)} + \delta_{np} C_{m\beta}^{(i)}.$$

- Intermediate derivative decomposition:

$$\frac{\partial C^{(i)}}{\partial \hat{R}^{(i)}} = \underbrace{\left.\frac{\partial C^{(i)}}{\partial \hat{R}^{(i)}}\right|_{G}}_{\text{explicit}} + \underbrace{\left.\frac{\partial C^{(i)}}{\partial G^{(i)}}\right|_{\hat{R}} \frac{\partial G^{(i)}}{\partial \hat{R}^{(i)}}}_{\text{implicit}}.$$

Fitting Network Derivative

- Atomic energy for atom i:

$$E^i = \mathrm{NN}_{s_i}\big(\mathrm{vec}(D^i)\big), \qquad D^i \in \mathbb{R}^{M\times M'}.$$

- For force assembly we require the Jacobian

$$J^i = \frac{\partial E^i}{\partial D^i} \in \mathbb{R}^{M\times M'}.$$

- The fitting network is a deep nonlinear mapping (fully-connected layers, activations, residuals).
- Explicit analytic differentiation is impractical $\rightarrow$ **automatic differentiation (AD)** is used.

Forward-Mode AD in HALMD

- HALMD uses **forward-mode AD** (Boost.Autodiff).
- For each descriptor component $\alpha = 1, \ldots, MM'$:
  1. Seed $\mathrm{vec}(D^i)_\alpha$ with unit tangent.
  2. Propagate through the full fitting network.
  3. Extract $\partial E^i / \partial \mathrm{vec}(D^i)_\alpha$.
- After all seeds:

$$\frac{\partial E^i}{\partial \mathrm{vec}(D^i)} \in \mathbb{R}^{MM'}.$$

- Reshape to obtain the descriptor Jacobian:

$$J^i = \mathrm{reshape}_{M \times M'} \left(\frac{\partial E^i}{\partial \mathrm{vec}(D^i)} \right).$$

Final Force Assembly

- Total force on atom k:

$$\mathbf{F}_k = -\frac{\partial E}{\partial \mathbf{r}_k} = -\sum_i \frac{\partial E^i}{\partial \mathbf{r}_k}.$$

- Descriptor chain:

$$\mathbf{r} \to \hat{R}^i \to G^i \to C^i \to D^i \to E^i.$$

- Using the chain rule:

$$\mathbf{F}_k = -\sum_i \sum_{m,n} J^i_{mn} \frac{\partial D^i_{mn}}{\partial \mathbf{r}_k},$$

where

$$J^i = \frac{\partial E^i}{\partial D^i}.$$

Explicit DeepMD-v2 Force Expression

- Descriptor derivative:

$$\frac{\partial D^i_{mn}}{\partial \hat{R}^i_{j\alpha}} \quad \text{(from descriptor analysis)}.$$

- Geometry derivative:

$$\frac{\partial \hat{R}^i_{j\alpha}}{\partial \mathbf{r}_k} \quad \text{(analytic geometry)}.$$

- Final force:

$$\boxed{\mathbf{F}_k = -\sum_i \sum_j \sum_{m=1}^{M} \sum_{n=1}^{M'} J^i_{mn} \frac{\partial D^i_{mn}}{\partial \hat{R}^i_{j\alpha}} \frac{\partial \hat{R}^i_{j\alpha}}{\partial \mathbf{r}_k}}.$$

- Hybrid structure:
  - **Neural:** J^i, $\partial G^i / \partial \hat{R}$ (via AD)
  - **Analytic:** $\partial \hat{R} / \partial \mathbf{r}$

Energy Agreement

- HALMD reproduces DeepMD energies to single-precision floating-point accuracy.
- Verified for:
  - Cu (Model A),
  - HEA,
  - Garnet,
  - Cu (Model B with different statistics).
- Agreement holds across:
  - increasing system sizes,
  - chemically complex multi-species systems,
  - different normalization statistics.
- Confirms correctness of:
  - descriptor construction,
  - embedding networks,
  - fitting networks,
  - energy bias terms.

Energy Agreement Across All Models and Species

Model	Species	Atoms	DeepMD (eV)	HALMD (eV)	Match
Cu A	1	4	−6673.84	−6673.84	✓
Cu A	1	32	−53412.4	−53412.4	✓
Cu A	1	108	−180303	−180303	✓
Cu A	1	256	−427425	−427425	✓
Cu B	1	4	−7.7443	−7.7443	✓
Cu B	1	32	−89.4412	−89.4412	✓
Cu B	1	108	−335.96	−335.96	✓
Cu B	1	256	−836.414	−836.414	✓
HEA	5	16	−66.7551048	−66.7551	✓
HEA	5	128	−895.4982878	−895.498	✓
HEA	5	432	−3490.860446	−3490.86	✓
Garnet	5	161	−108445	−108445	✓

- Agreement holds for monoatomic and multi-species systems.
- Verified across different training statistics and descriptor normalizations.
- Confirms correctness of full DeepMD-v2 inference in HALMD.

Conclusion & Outlook

- First fully explicit *DeepMD-v2 implementation in HALMD* with multi-species descriptors, energies, and force derivatives.
- Verified energy agreement for monoatomic and complex multi-species systems enables *accurate large-scale ML-driven molecular dynamics* without framework overhead.
- This work enables:
  - faster ML-based simulations,
  - improved HPC scalability,
  - closer integration of ML and physics-based modeling.
- With completed forces and further optimisation, HALMD can become a *high-performance platform for next-generation materials simulation*.

References I

[1] Jörg Behler and Michele Parrinello. "Generalized neural-network representation of high-dimensional potential-energy surfaces". In: *Physical Review Letters* 98.14 (2007), p. 146401. DOI: 10.1103/PhysRevLett.98.146401.

[2] Donald W. Brenner et al. "A second-generation reactive empirical bond order (REBO) potential energy expression for hydrocarbons". In: *Journal of Physics: Condensed Matter* 14.4 (2002), pp. 783–802. DOI: 10.1088/0953-8984/14/4/312.

[3] Peter H Colberg and Felix Höfling. "Highly accelerated simulations of glassy dynamics using GPUs: Caveats on limited floating-point precision". In: *Computer Physics Communications* 182.5 (2011), pp. 1120–1129.

[4] Andres Cruz. "Deep Neural Networks Potentials for Scalable Molecular Dynamics Simulations on Accelerator Hardware". Master's Thesis. Freie Universität Berlin, Sept. 2025.

References II

[5] Murray S. Daw and Michael I. Baskes. "Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals". In: *Physical Review B* 29.12 (1984), pp. 6443–6453. DOI: 10.1103/PhysRevB.29.6443.

[6] Dominik Marx and Jürg Hutter. *Ab initio molecular dynamics: basic theory and advanced methods*. Cambridge University Press, 2009.

[7] Frank Noé et al. "Machine learning for molecular simulation". In: *Annual review of physical chemistry* 71.1 (2020), pp. 361–390.

[8] Han Wang, Linfeng Zhang, Jiequn Han, et al. "DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics". In: *Computer Physics Communications* 228 (2018), pp. 178–184.

[9] Jinzhe Zeng et al. "DeePMD-kit v2: A software package for deep potential models". In: *The Journal of Chemical Physics* 159.5 (2023).

Thank You!