



MSU Graduate Theses

Spring 2023


Machine Learning Strategies for Potential Development in High-Entropy Driven Nickel-Based Superalloys

Marium Mostafiz Mou

Missouri State University, mm665s@MissouriState.edu

As with any intellectual project, the content and views expressed in this thesis may be considered objectionable by some readers. However, this student-scholar's work has been judged to have academic value by the student's thesis committee members trained in the discipline. The content and views expressed in this thesis are those of the student-scholar and are not endorsed by Missouri State University, its Graduate College, or its employees.

Follow this and additional works at: <https://bearworks.missouristate.edu/theses>

 Part of the [Artificial Intelligence and Robotics Commons](#), [Atomic, Molecular and Optical Physics Commons](#), and the [Metallurgy Commons](#)

Recommended Citation

Mou, Marium Mostafiz, "Machine Learning Strategies for Potential Development in High-Entropy Driven Nickel-Based Superalloys" (2023). *MSU Graduate Theses*. 3860.
<https://bearworks.missouristate.edu/theses/3860>

This article or document was made available through BearWorks, the institutional repository of Missouri State University. The work contained in it may be protected by copyright and require permission of the copyright holder for reuse or redistribution.

For more information, please contact bearworks@missouristate.edu.

**MACHINE LEARNING STRATEGIES FOR POTENTIAL DEVELOPMENT IN HIGH-
ENTROPY DRIVEN NICKEL-BASED SUPERALLOYS**

A Master's Thesis

Presented to

The Graduate College of

Missouri State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science, Materials Science

By

Marium Mostafiz Mou

May 2023

Copyright 2023 by Marium Mostafiz Mou

MACHINE LEARNING STRATEGIES FOR POTENTIAL DEVELOPMENT IN HIGH-ENTROPY DRIVEN NICKEL-BASED SUPERALLOYS

Physics, Astronomy, and Materials Science

Missouri State University, May 2023

Master of Science

Marium Mostafiz Mou

ABSTRACT

In this study, I developed Deep Learning interatomic potentials to model a multi-phase and multi-component system of Ni-based Superalloys. The system has up to three major phase constituents, namely Gamma, Gamma Prime, and Transition-metal rich Carbide. I utilized invariant scalar-based and/or equivariant, tensor-based neural network (NN) approach as implemented in DEEPM, NEQUIP/ALLEGRO codes, respectively, and Moment Tensor Potential (MTP). For the training and validation sets, I employed the ab-initio molecular dynamics (AIMD) trajectory results and ground state DFT calculations, including the energy, force, and virial database from highly diverse compositions, temperatures, and pressures following a “High Entropy Strategy.” The Deep learning potential was systematically developed for 4, 5, 7, and 10 component systems based on the complexity level of the phase mixtures. To optimize the hyperparameters, I used a series of machine learning (ML) algorithms to lower the RMSE of the force components and then compare the accuracy of both the potentials developed using the two types of Deep Learning potentials through a variety of large-scale molecular dynamics (MD) simulations. The GPU-based supercomputer support from NERSC (Perlmutter) is gratefully acknowledged.

KEYWORDS: superalloy, nickel, machine learning, neural network, molecular dynamics, high entropy

**MACHINE LEARNING STRATEGIES FOR POTENTIAL DEVELOPMENT IN HIGH-
ENTROPY DRIVEN NICKEL-BASED SUPERALLOYS**

By

Marium Mostafiz Mou

A Master's Thesis
Submitted to the Graduate College
Of Missouri State University
In Partial Fulfillment of the Requirements
For the Degree of Master of Science, Materials Science

May 2023

Approved:

Ridwan Sakidja, Ph.D., Thesis Committee Chair

Robert Mayanovic, Ph.D., Committee Member

Kartik C. Ghosh, Ph.D., Committee Member

Julie Masterson, Ph.D., Dean of the Graduate College

In the interest of academic freedom and the principle of free speech, approval of this thesis indicates the format is acceptable and meets the academic criteria for the discipline as determined by the faculty that constitute the thesis committee. The content and views expressed in this thesis are those of the student-scholar and are not endorsed by Missouri State University, its Graduate College, or its employees.

ACKNOWLEDGEMENTS

I'd like to express my gratitude to my advisor, Dr. Ridwan Sakidja, for all his advice, support, and encouragement throughout this journey. I would like to express my heartfelt appreciation to all the faculty and staff in the Physics, Astronomy, and Materials Science Department for providing excellent learning opportunities.

I am grateful to DOE's National Energy Technology Laboratory (Grant No. FE0031554) and the National Energy Research Scientific Computing Center for their computing assistance (NERSC). Finally, I just wanted to take a moment to express my deepest gratitude to my husband, Farhan Ishrak. His unwavering encouragement and belief in me helped me push through the toughest of times. I am so grateful to have him by my side, and I couldn't have asked for a better partner.

I dedicate this thesis to my parents, without whom none of this would be possible.

TABLE OF CONTENTS

| | |
|--|----|
| INTRODUCTION | 1 |
| Literature Review | 1 |
| Nickel Based Superalloy | 4 |
| High Entropy Alloy | 5 |
| Computational Methods for Materials Research | 7 |
| Review of the Models of Interatomic Interaction | 7 |
| Machine Learning Interatomic Potentials | 13 |
| METHODS | 34 |
| High Entropy Alloy Structures for Ab-initio Simulation | 34 |
| Machine Learning Interatomic Potentials Training | 37 |
| RESULTS AND DISCUSSION | 40 |
| DeePMD Model Accuracy | 40 |
| DeePMD Model Verification using MD Simulations | 42 |
| DeePMD Model Accuracy with Higher K-Points Input | 49 |
| Allegro Model Accuracy | 50 |
| Allegro Model Verification using MD Simulations | 53 |
| Allegro Data Efficiency | 54 |
| Hyperparameter Optimization | 56 |
| MTP Model Accuracy | 58 |
| MTP Model Verification using MD Simulations | 59 |
| MTP Active Learning and Future work | 61 |
| CONCLUSION | 64 |
| REFERENCES | 65 |

LIST OF TABLES

| | |
|---|----|
| Table 1. Selection of ab initio data used for model training of HEA 4, 5, 7 | 36 |
| Table 2. Cut off radius and error | 39 |
| Table 3. Error analysis of three models of DeePMD | 42 |
| Table 4. Predicted CIJ | 46 |
| Table 5. Error analysis of HEA7 | 50 |
| Table 6. Error analysis of three models of Allegro | 51 |
| Table 7. Hyperparameters and RMSE | 57 |

LIST OF FIGURES

| | |
|---|----|
| Figure 1. Nickel based superalloy microstructures | 4 |
| Figure 2. General representation of a neural network | 14 |
| Figure 3. NN suggested by Behler and Parrinello | 16 |
| Figure 4. Allegro Network (a) Model architecture (b) Details a tensor product layer | 26 |
| Figure 5. High entropy alloy strategy | 34 |
| Figure 6. Input structures for the DFT calculation. (a) FCC (b) HCP (c) Random/melt (d) BCC 35 | |
| Figure 7. Energy matching by DeePMD of (a) HEA4 (b) HEA7 (c) HEA10 as predicted by DeePMD potentials in y-axis vs training-DFT data(x-axis) | 40 |
| Figure 8. Virial matching by DeePMD potentials of (a) HEA4 (b) HEA7 (c) HEA10 as predicted in y-axis vs training-DFT data(x-axis) | 41 |
| Figure 9. Compression test under various temperatures using HEA10 potential (a) the model (b) stress vs. strain graph (c) the dislocation band generation | 43 |
| Figure 10: Compression test under various temperatures using HEA10 potential (a) the model (b) stress vs. strain graph (c) the dislocation band generation | 44 |
| Figure 11. Coefficient of thermal expansion at 300K using HEA7 potential (a) HEA 5(Ni,Al,Co,Cr,Fe) (b) HEA 4(Ni,Al,Co,Cr) | 45 |
| Figure 12. Predicted average lattice parameters of $Cr_{123}C_6$ varying by temperature. The orange boxes use HEA10 potential, and the blue dots use (Fe–Cr–Mo–W–C) potential. | 46 |
| Figure 13. Melting point of single Phase $Fe_{23}C_6$ at two different heating rate using (a) HEA10 potential(Ni,Al,Co,Cr,Fe,C,Mo,W,Ti,Nb) (b) $Fe_{23}C_6$ potential | 47 |
| Figure 14. Melting point of single phase (a) Ni at 1900K (b) Al at 850K | 48 |
| Figure 15. Dislocation mobility in INCONEL 740H using Deep Potential (a) dislocation in the beginning (b) the model at the end (c) dislocation loop at the beginning (d) at the end (e) positions with time (f) velocity of the dislocation with varying stress | 49 |
| Figure 16. Energy and virial matching by Allegro of (a) HEA4 energy (b) HEA7 energy (c) HEA4 energy (d) HEA7 virial as predicted in y-axis vs training-DFT data(x-axis) | 51 |

| | |
|--|----|
| Figure 17. Force of HEA with 4 components (a) HEA4 predicted force by DeePMD (b) HEA4 predicted force by Allegro (c) HEA4 force RMSE Comparison (d) HEA4 force convergence with energy | 52 |
| Figure 18. Thermal stability of supercell models of Gamma phase and Gamma Gamma Prime phase of HEA 4 at different temperature using HEA 4 Allegro potential | 53 |
| Figure 19. RMSE plot of different training sets of HEA 4 force RMSE | 55 |
| Figure 20. Parallel coordinates of HEA 4 system | 56 |
| Figure 21. Error of force for individual species of HEA 4 (a) Ni (b) Al (c) Co (d) Cr | 58 |
| Figure 22. Lattice constant of HEA4 using DeePMD and MTP Potentials | 59 |
| Figure 23. Thermal stability of supercell models of Gamma phase of HEA 4 at different temperature using HEA 5 MTP Potential | 60 |
| Figure 24. Scheme of active learning iterations for training MTP | 62 |

INTRODUCTION

Literature Review

Nickel-based superalloys are a class of high-performance materials widely used in various applications due to their excellent mechanical properties, such as high strength, good corrosion resistance, and high-temperature stability. They are used in aerospace, power generation, and chemical processing industries. These superalloys typically contain nickel as the primary element, with additions of other elements such as cobalt, chromium, molybdenum, and tungsten. These additions provide the alloy with unique properties, and the specific composition can be tailored to meet the demands of different applications. One of the most well-known nickel-based superalloys is Inconel 718, which contains nickel (50-55%), chromium (17-21%), molybdenum (2.8-3.3%), niobium (4.75-5.5%), titanium (0.65-1%), aluminum (0.2-0.8%), and cobalt (1%). This alloy has a high strength-to-weight ratio and is used in aerospace components such as turbine blades and discs.

Another important nickel-based superalloy is Haynes 282, which is a cobalt-based superalloy with nickel (22-25%), cobalt (20-22%), chromium (14-16%), tungsten (6-8%), molybdenum (3-4%), and tantalum (1%). This alloy is known for its excellent high-temperature strength and resistance to oxidation and corrosion. One of the research areas that has attracted attention in recent years is the use of additive manufacturing to produce nickel-based superalloys. This technology offers the ability to produce complex geometries with improved mechanical properties and reduced weight. The specific composition can be tailored to meet the demands of different applications, and the use of additive manufacturing technology has attracted attention in recent years for producing complex geometries with improved properties.

High entropy alloys (HEAs) are a class of materials that contain multiple metallic elements in roughly equal proportions. This unique composition results in a high degree of randomness, or entropy, in the atomic arrangement, leading to improved mechanical properties such as high strength and toughness. The correlation between HEAs and nickel-based superalloys is that both types of alloys contain multiple metallic elements, but the key difference is that HEAs contain roughly equal proportions of the elements, while nickel-based superalloys have a high percentage of nickel and other elements in specific proportions to give it its specific properties. Both HEAs and nickel-based superalloys have the potential for high-temperature applications, but nickel-based superalloys have been more extensively studied and have a more established role in industries such as aerospace and power generation. In recent years, computational materials science has played an increasingly important role in the discovery and design of HEAs. The use of first-principles calculations, statistical mechanics, and machine learning methods has enabled the identification of new HEA compositions and the prediction of their properties. However, there is still much to be discovered and understood about these alloys, and ongoing research efforts are focused on further exploring the properties and behavior of HEAs.

Machine learning interatomic potentials (MLIP) are a type of computational model used to predict the behavior of materials at the atomic scale. These potentials are trained on large sets of data, such as quantum mechanical calculations or experimental measurements, and can be used to simulate the properties of a wide range of materials, including nickel-based superalloys. They can be used to simulate the properties of nickel-based superalloys in a computationally efficient manner. These potentials can be trained on data from quantum mechanical calculations or experiments and can be used to predict the behavior of the material under different conditions,

such as high temperatures or high stress. This can be useful for designing new alloys or optimizing existing ones. Overall, Machine learning interatomic potentials are a promising approach to simulate the properties of nickel-based superalloys. This can be used to design new alloys or optimize existing ones. Computational methods can be useful in materials research for developing predictive models of material properties in different conditions. When compared to other methods like density functional theory(DFT), molecular dynamics (MD) simulations offer a fine balance between accuracy and efficiency. The quality of the interatomic potentials (IAP), however, is crucial to MD's accuracy. In order to reliably reproduce the material's properties, conventional IAPs are parametric models, which are then fitted to experimental or computational data.

In this work, we developed the Deep Learning potentials for highly concentrated multi-component metallic systems by utilizing Deep Learning/Machine Learning code. The complex system has up to ten elements with three major phase constituents, namely Gamma, Gamma Prime, and Transition-metal rich Carbide. We systematically developed the Deep Learning potentials for 5, 7, and 10 component systems based on the complexity level of the phase mixtures. To optimize the hyperparameters, we used a series of machine learning (ML) algorithms to lower the RMSE of the force components. We then compare the accuracy of both the potentials developed using the two types of Deep Learning potentials through a variety of large-scale molecular dynamics (MD) simulations.

Nickel Based Superalloy

Nickel-base superalloys are high-temperature corrosion-resistant alloys that are typically used at temperatures above 500°C. They typically contain up to ten alloying elements, including

light elements such as boron or carbon, and heavy refractory elements such as tantalum, tungsten, or rhenium. Even at temperatures close to their melting points, superalloys exhibit excellent resistance to creep, sulfidation, and oxidation. Ni-based superalloys exhibit excellent mechanical behavior even at high temperatures because they have a two-phase γ/γ' matrix-precipitate microstructure. Aluminum and/or titanium are the essential solutes in nickel-based superalloys, with a total concentration of less than 10 atomic percent. This results in a two-phase equilibrium microstructure made up of gamma (γ) and gamma-prime (γ'). It is the γ' that is largely responsible for the material's elevated-temperature strength and incredible resistance to creep deformation (1). In Figure 1, we can see the gamma (γ) and gamma-prime (γ'), and carbides in the grain boundary. These phases make nickel-based superalloy strong and give excellent creep properties.

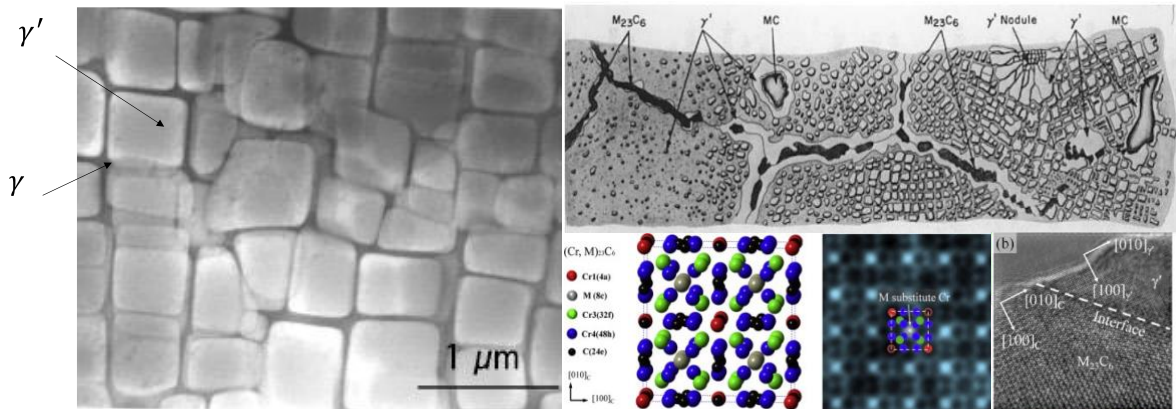


Figure 1. Nickel based superalloy microstructures

High Entropy Alloy

High entropy alloy (HEA) is a type of metallic alloy that is characterized by having a high degree of randomness or disorder in the distribution of its constituent elements. These alloys typically contain five or more elements in roughly equal atomic proportions. Because of their

high degree of disorder, HEAs exhibit unique mechanical, thermal, and physical properties that differ from those of traditional alloys made from just one or a few elements. One of the key benefits of high entropy alloys is their ability to maintain strength and other properties at high temperatures. The high degree of disorder in the atomic structure of these alloys helps to prevent the formation of defects and other structural weaknesses that can occur in traditional alloys at high temperatures. Additionally, the high degree of randomness in the distribution of elements in HEAs can lead to unique combinations of properties, such as high strength, high ductility, and high thermal stability, which are difficult to achieve with traditional alloys.

One of the key findings in HEA research has been the observation of solid solution strengthening, where the presence of multiple elements in the alloy leads to a strengthening effect beyond what would be expected from a single element. This has been attributed to the high degree of configurational entropy in the atomic arrangements. Another important aspect of HEA research has been the study of phase stability and microstructure. Many HEAs exhibit a single-phase solid solution microstructure, while others may exhibit multiple phases or complex microstructures. These microstructures have been found to play a critical role in determining the mechanical properties of the alloys. HEAs have also been found to exhibit high corrosion resistance, thermal stability, and exceptional wear resistance.

There are many different types of HEAs that have been studied, including those based on transition metal elements, refractory metals, and metal-nonmetal combinations. These alloys have been found to have potential applications in a wide range of fields, including aerospace, defense, transportation, biomedical, and energy. Some examples of high entropy alloys include AlCoCrCuFeNi and AlNiCo, which are known for their high strength and corrosion resistance. Another example is the AlCrFeCoNi alloy, which is known for its high strength and high thermal

stability, making it a candidate for high temperature applications such as jet engine components. HEA's are still a relatively new area of research, and scientists are still discovering new ways to use these alloys, but they show great potential in aerospace, automotive, and biomedical applications.

To create new materials known as high-entropy alloys, a new alloying technique that combines several principal elements in high concentrations has gained popularity over the past 15 years. Although only a small portion of the multi-dimensional compositional space has been studied thus far, it is practically infinite. An additional intriguing justification for studying these alloys was offered by Jien-Wei Yeh and colleagues (2). They suggested that the presence of several (five or more) elements in nearly equal atomic proportions would increase the configurational entropy of mixing by an amount sufficient to outweigh the enthalpies of compound formation, preventing the formation of potentially hazardous intermetallics. It was counterintuitive to think that the more elements there were in concentrated alloys, the more likely it was that some of them would react to form compounds. This belief was probably based on binary phase diagrams, in which solid solutions are typically found at the ends and compounds near the centers. They coined a catchy new name for this High-entropy alloys (HEAs) (3). Alloying has long been used to impart desirable properties to materials. Typically, it entails the addition of small amounts of secondary elements to a primary element. However, for the past decade and a half, a new alloying strategy that involves the combination of multiple principal elements in high concentrations to create new materials known as high-entropy alloys has been popular (4).

Computational Methods for Materials Research

In addition to experimental research, computational methods can be used to investigate the structure and properties of materials (5). Experimental results always contain sources of uncertainty, and computational methods rely on varying degrees of approximation due to the extreme computational requirements for solving the physical equations precisely. Different computational methods can range from fully quantum mechanical calculations that explicitly account for the electronic structure to classical molecular dynamics simulations in which the interaction between individual atoms is described by approximated IAP. A profound understanding can be of great assistance in the design and development of materials, but it requires experimentally characterized material properties and a theoretical framework for the observed qualities. Simulations of materials physics can aid in the comprehension of the microscopic phenomena and characteristics observed in experimental work, and in many instances serve as a bridge between experiment and theory.

Review of the Models of Interatomic Interaction

This section will provide an overview of various interatomic interaction models. The term force field or interatomic potential is frequently used in place of interatomic interaction model. Interatomic potential provides a functional dependence of the energy of the atomistic system on the positions and types of atoms constituting the system:

$$E = E(x)$$

From here x will be used to denote the configuration, meaning $x = \{r_i, z_i\}$, where r_i and z_i denote the position vector and the atomic number of the i th atom in configuration.

Empirical Potentials. In order to avoid solving the governing quantum-mechanical equations, empirical potentials are employed. In place of this, they offer an explicit analytical form with coefficients derived from fitting experimental or quantum-mechanical data. Majority of the empirical potentials are local, i.e., they account for interaction of each atom only with its nearest neighborhood, usually within some cutoff radius. Non-local potentials are typically applied to charged systems where the contribution of Coulomb forces at great distances is crucial. For local potentials, E is subdivided into the contributions V of individual atomic neighborhoods. The function V is referred to as an interatomic potential. To define the neighborhood of the i th atom n_i , we define r_{ij} as the position of the j th atom relative to the i th atom (r_{ij} is a vector) and z_j as the type of the j th atom. The i th atom's neighborhood is the collection of r_{ij} and z_j , and the previous equation can be rewritten as follows:

$$E(x) = \sum_i V(n_i)$$

Locality of the potential is expressed by the stipulation that $V(n_i)$ only depends on atoms that are closer to the atom i than some cutoff distance R_{cut} , which is typically around 5 Å. In metallic systems this typically means the closest neighbors up to some order are included in the atomic environment[6]. Examples and definition of potentials are given below:

Pair Potentials. Due to the fact that pair potentials are only two-body interactions, the total energy of a configuration can be expressed as the sum of all contributions made by two bodies. Examples of pair potentials are Lennard-Jones potential and Morse potential.

Three-Body Potentials. Tersoff and Stillinger-Weber potentials are among these. While two-body terms are often connected with bond lengths, three-body terms are typically related with bond angles.

Many-Body Potentials. The embedded atom approach, often known as the EAM potential, is the type of empirical potential that is used the most frequently for metals (6). It stems from the concept of imbedding each atom in an electronic atmosphere of a certain density, which is influenced by the density of the surrounding atoms.

Quantum-Mechanical Models and Density Functional Theory. Different approximations of the time-independent Schroedinger equation serve as the foundation for quantum-mechanical models. As the solution is generally impossible, the Born-Oppenheimer approximation is frequently employed. Due to a three-orders-of-magnitude difference in mass and comparable electrostatic forces, it locks up the nuclei, which move much more slowly than electrons. The Born-Oppenheimer approximation still can't be solved in general, and more simplifications, like density functional theory, are needed. Hybrid functionals in DFT also use predictions from Hartree-Fock theory (7). They can calculate many chemical systems with high accuracy by making changes to the exchange-correlation functionals. Aside from heavy methods like DFT hybrid functionals or quantum Monte Carlo (which uses an exact many-body wavefunction and deals directly with quantum effects), some QM models are semi-empirical and designed for fast calculations. They can make close but often accurate predictions for large systems, whose evaluation would take too long with heavier methods, like linear scaling DFT, which includes some screening of interatomic interactions.

One of the most common approaches is density functional theory (DFT), which is particularly useful for metals and alloys because it provides an appealing trade-off between calculation accuracy and computational speed. This method was used in my research to calculate ab initio properties of alloys, and it deserves special attention and a more detailed description because of the aforementioned. The cost of conventional DFT scales as $O(N^3)$ with the number

of atoms, whereas there is also a "linear scaling DFT" implementation with $O(N)$ scaling, which is less accurate but still applicable, particularly for large systems.

Density functional theory comes from the Thomas-Fermi model, in which the ground state energies of atomic systems are calculated as an approximation of a function of electron density. This is also the main idea behind modern DFT: instead of solving for the many-body wavefunction of the electrons, find the electron density and study the properties of the system as functions of this density. Pierre Hohenberg and Walter Kohn laid the groundwork for the current DFT by giving rigorous proofs to two important theorems, now called Hohenberg-Kohn theorems, which state that the external potential is a unique function of the electron density of the system and the function that gives the ground-state energy of the system reaches the minimum value with respect to possible electron densities if and only if the input density is the true ground-state density. The properties defined by the external potential are uniquely determined by the ground-state density, according to the first theorem. As is customary in multi-atom quantum mechanical computations, the Born-Oppenheimer approximation is assumed in DFT (8).

Due to the interaction between the electrons and the imprecision of the kinetic energy term, the problem remained intractable until W. Kohn and L. J. Sham devised a solution: instead of solving the system of interacting electrons, replace it with non-interacting particles moving in a fictitious effective potential so that the density of the particles remains the same. The equations (known as KohnSham equations) describing this fictitious system are KohnSham equations:

$$\left(\frac{-\hbar^2}{2m_i} \nabla_i^2 + V_{ks}(r) \right) \psi_i(r) = \epsilon_i \psi_i(r), i \leq N_e$$

$$n(r) = \sum_{i=1}^N |\psi_i(r)|^2$$

where ε_i is the orbital energy of the Kohn-Sham orbital ψ_i and n is the density of the N-particle system. The energy functional becomes:

$$E[n(r)] = T[n(r)] + \frac{1}{2} \iint \frac{n(r)n(r')}{4\pi\varepsilon_0|r-r'|} drdr' + \int v_{ext}(r)n(r)dr + E_{xc}[n(r)]$$

Here $T[n(r)]$ is the kinetic energy of the system of non-interacting particles, the second term is the electronic coulomb energy, v_{ext} is the external coulomb potential due to the nuclei, and E_{xc} is the exchange-correlation functional. Unfortunately, the exact form of the exchange-correlation functional is unknown, but in recent decades a large number of approximate functionals have been developed. The electron density and ground state energies are determined by solving the Kohn-Sham equations iteratively. First, an initial estimate of the electron density is provided, followed by the computation of the electron density based on the obtained potential. This cycle is repeated until self-consistency is achieved. The VASP-software (9) used in the thesis work uses plane-wave basis sets:

$$\psi_{m,k}(r) = \sum_{|G| < G_{max}} c_m(k+G)e^{i(k+G).r}$$

where $\psi_{m,k}$ is the component of the wave function of an electron at band m , k is the wave vector and G are reciprocal lattice vectors. Since the basis set is infinite in terms of vectors G , a cutoff value G_{max} is used. There are many benefits to using the plane-wave basis, but due to the rapid oscillations of the wavefunctions, the orbitals close to the atom cores need to be simplified. Since valence electrons are almost never involved in chemical bonds, pseudopotentials are used in the core region to keep the wavefunctions unchanged and, by extension, the system's dynamics unchanged.

Classical Molecular Dynamics. Quantum mechanical calculations are computationally expensive, so classical Molecular Dynamics (MD) is used in materials simulations. Though less accurate, Molecular Dynamics simulations can study larger systems with longer simulation

times, making them useful in materials research. MD accuracy depends on interatomic potentials. Classical Molecular Dynamics simulations treat atoms and molecules as classical particles interacting artificially and numerically solve Newton's equations of motion to study their movement.

$$m_i \frac{d^2 r_i}{dt^2} = f_i$$

$$f_i = -\nabla_i V$$

where r_i , m_i and f_i are the position, mass and force acting on particle i respectively and V is the potential energy. The Born-Oppenheimer approximation, electron eigenstates, and nuclei motion under the potential are assumed again. Interatomic potential includes atom bonds. The timestep should be small enough for accuracy but not at the expense of efficiency because the equations of motion are integrated by adding small increments to positions and velocities depending on the forces acting on the particles. The simple Euler algorithm can integrate Newton's equations of motion, but the Velocity Verlet and Predictor-Corrector algorithms provide more accurate calculations with less computation time and memory usage and allow for larger timestep values. Many MD interatomic potentials only affect nearby particles, so it is unnecessary to compute all particle interactions with every step. Thus, interactions are limited to a cut-off distance. Periodic boundary conditions in the simulation cell can model larger systems since even the smallest experimental systems usually have more atoms than can be simulated. Most of the time, it's helpful to look at the system as a whole and make comparisons to what's been discovered through experimentation. Since the ergodic hypothesis is satisfied and averages over time can be used to investigate macroscopic properties, MD simulations are frequently used for this purpose. Because experiments are typically conducted in a controlled environment with constant temperature and pressure, it is important to regulate these thermodynamic properties of

the system in a way that limits the number of artifacts that can arise in computer simulations. This is achieved by fine-tuning the dynamics to ensure that the simulated thermodynamic process is accurate. There are a variety of temperature and pressure regulators available for this purpose, including the Berendsen and Nosé-Hoover thermostats and the Andersen and Parrinello-Rahman barostats.

Machine Learning Interatomic Potential

Types of MLIP's. Machine Learning Potential (MLP) is an interatomic potential that is derived with or employs any kind or variety of machine learning technologies. The Gaussian Approximation Potentials (GAP) and the Neural Network Potentials (NNP) are two of the most common approaches. Bartok et al. were the ones who developed the GAP method, and it has been put to good use in the production of a GAP for graphene. In the meantime, neural networks as a type of machine learning and as a potential for machine learning have been present for some time. However, the works of Behler and Parrinello, which will be mentioned in a moment, made significant improvements to neural networks' machine learning potential. In the context of our study, we shall continue to concentrate on NNPs.

Neural Networks. Neural networks are a type of supervised machine learning in which a computer attempts to learn and predict from data. Neural networks do this by constructing a network of 'neurons' composed of layers of linked nodes designed to simulate biological cognitive processes. Figure 2 depicts a simple neural network design. Each input layer sends data to each corresponding output layer, forming a network. The hidden layers in the figure, for example, each get data from the inputs multiplied by a specific weight. The weights are used to lend significance to the data. For example, if this was a classification neural network, data that

better captured distinctions and aided in classification would be given more weight. At the receiver node, this data is subsequently transmitted through an activation function. The goal is to eliminate network linearity and regulate the flow of information. A neural network is a set of linear equations without an activation function, and hence merely gives a linear regression model. As the model learns to detect traits that contribute to classification, the weights and bias values are adjusted (10).

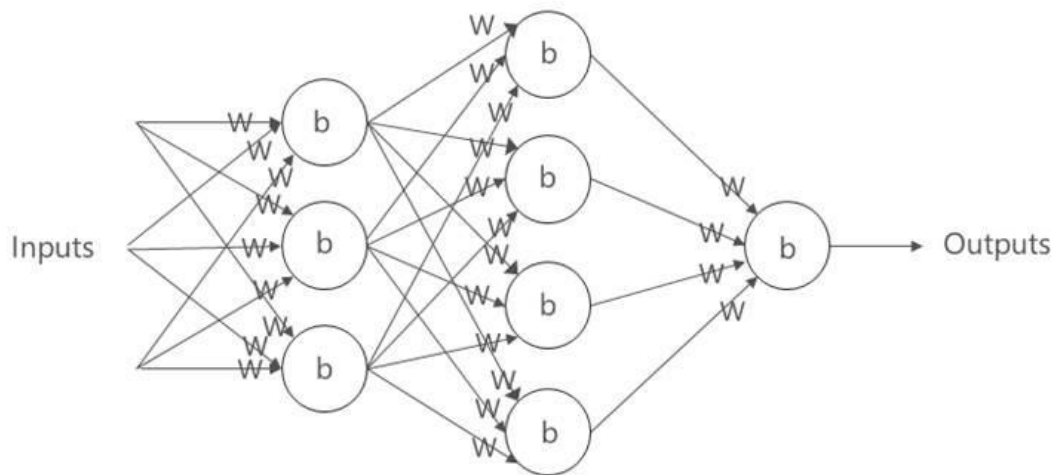


Figure 2. General representation of a neural network

Neural Network Potential Implementation. A neural network can be used for potential development by training it on a dataset of inputs and outputs that are relevant to the task at hand. Once the network has been trained, it can be used to make predictions on new inputs it has never seen before. This process can be used for a variety of tasks such as image recognition, natural language processing, and prediction. To use a neural network, one will typically need to use a software library, such as TensorFlow or PyTorch, that provides an implementation of the network architecture and training algorithm of need. After that, we will need to prepare data, choosing the right architecture and fine-tune the hyperparameters, then evaluate the performance

of the model, and iterate until we achieve the desired performance. Behler and Parrinello proposed a neural network architecture called "Behler-Parrinello Neural Network" (BPNN) which is a type of artificial neural network that is designed to model the properties of materials. The BPNN architecture consists of multiple layers of artificial neurons, which are connected by weights and biases. The network is trained using a set of input-output pairs, where the input is a set of structural features of a material, and the output is a set of properties of that material. The BPNN is highly effective in modeling the properties of complex materials, such as those found in the field of computational materials science. The BPNN has been widely used in various applications such as prediction of crystal structures, phase transitions, and thermodynamic properties of materials.

Utilizing the neural network structure shown above, this creates several problems. Atomic locations are used as the NNP's input. This mistake results from a type of "double dipping." For instance, the system i-j-k will return a specific energy value if it consists of three identical atoms, such as atoms i, j, and k. Now, if we take the exact same system but move two atoms around, let's say we have j-i-k, we still have the same system. Sadly, the NNP won't view it that way; instead, it will consider this to be a completely new system, and it's possible that it may receive a different energy value than it did earlier. Behler suggested a set of atom-centered symmetry functions to get around this issue by defining local atomic surroundings rather than precise atom placements. These functions are designed to eliminate the variation in system energy that would occur if the system's atom coordinates were exchanged, or if the system were rotated or translated. Equations 8 and 9 are examples of radial and angular symmetry functions:

$$G_i = \sum_j \cos(kR_{ij}) \cdot f_c(R_{ij}) \quad \text{where } f_c(R_{ij}) = \begin{cases} 0.5 \left[\cos\left(\frac{\pi R_{ij}}{R_c}\right) + 1 \right] & \text{for } R_{ij} \leq R_c \\ 0 & \text{for } R_{ij} > R_c \end{cases}$$

$$G_i = 2^{1-\zeta} \sum_{j,k \neq i}^{all} (1 + \lambda \cos^\zeta(\theta_{ijk})) \cdot e^{-\eta(R_{ij}^2 + R_{ik}^2)} \cdot f_c(R_{ij}) \cdot f_c(R_{ik})$$

In addition to their recommendation that we use symmetry functions to transform the inputs, Behler and Parrinello also offered a slightly modified form for the neural network. Figure 3 depicts this network, which first converts Cartesian coordinates to symmetry functions (11), then uses separate neural networks for each atom to determine that atom's contribution to the system's energy, and finally adds up all of the atoms' contributions. Although the aforementioned techniques have garnered widespread acclaim and are used in the present work, it is important to remember that NNPs' inputs may be tailored to suit a variety of purposes.

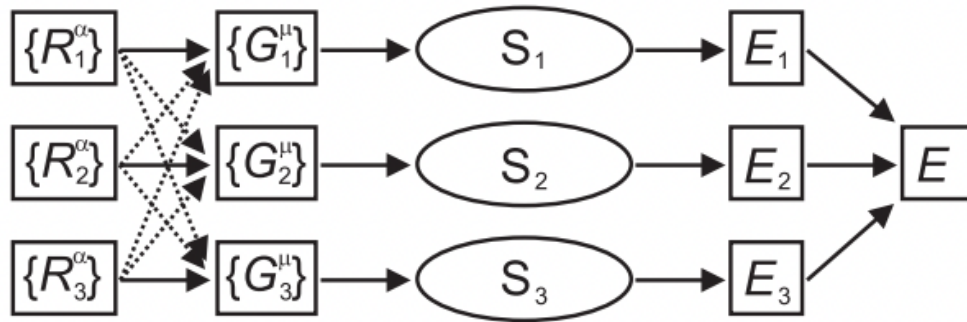


Figure 3. NN suggested by Behler and Parrinello

For my thesis, I have used invariant and equivariant NNs. Invariant neural networks are designed to be robust to certain types of transformations or changes in the input data. For example, a neural network that is invariant to translations in an image would be able to recognize an object regardless of its location within the image. This can be achieved by using pooling layers or other types of down-sampling in the network architecture. Equivariant neural networks, on the other hand, are designed to maintain certain properties of the input data after it has been processed by the network. For example, a neural network that is equivariant to rotations in an

image would be able to recognize an object regardless of its orientation. This can be achieved by using equivariant layers such as Steerable CNNs or group convolution.

Invariant and equivariant neural networks have been used in a variety of applications, including image recognition, object detection, and natural language processing. For example, in image recognition, invariant networks have been used to improve the robustness of the network to translation and rotation of the image. Equivariant networks have been used to improve the ability of the network to recognize objects in the image regardless of their orientation. In natural language processing, equivariant networks have been used to improve the ability of the network to understand and generate text despite the variations in word order. Invariant and equivariant neural networks have been widely studied as a way to improve the robustness and generalization of deep learning models. They have been used in a variety of applications, and have shown promising results in many cases.

DeePMD. I have used DeePMD-kit as an invariant NN for the thesis. DeePMD (Deep Potential Molecular Dynamics) is a deep learning based method for simulating the dynamics of molecular systems. It has been used in a variety of research studies to investigate the properties of different materials, including liquids, solids, and biomolecules. DeePMD-kit is a software package for deep learning potential energy surface (DL-PES) fitting and molecular dynamics (MD) simulation (12). It is designed to work with the Atomistic Machine Learning (AML) framework, which utilizes neural networks to represent the potential energy of a system of atoms. The software package is developed with the goal of providing a simple and user-friendly interface for performing DL-PES fitting and MD simulation. It includes various tools for data preprocessing, neural network training, and MD simulation, as well as a variety of utility functions for analyzing and visualizing the results. One of the key features of DeePMD-kit is its

ability to handle large datasets. It can handle datasets with millions of data points and is capable of training neural networks with thousands of parameters. Additionally, it includes built-in support for parallel computing, which allows for efficient training and simulation on high-performance computing clusters.

DeePMD-kit has been used in a variety of studies, including the prediction of potential energy surfaces for various materials, such as metals and semiconductors, as well as the simulation of complex chemical reactions and biological systems. One of the first studies to use DeePMD was published in 2017 by Han et al., who used the method to investigate the properties of liquid water (13). The study found that DeePMD was able to accurately predict the structural and dynamical properties of water, including the radial distribution function and the diffusion coefficient. Since then, DeePMD has been used in a number of studies to investigate the properties of different materials. In a study by Lu et al. (14), the DeePMD-kit was used to train a DL-PES for various systems. The results showed that the DL-PES was able to accurately predict the potential energy of the system, with a mean absolute error of less than 1 meV/atom. Additionally, the study showed that the DL-PES was able to accurately predict the properties of the system, such as the phonon dispersion and thermal conductivity. In another study (15), the authors describe utilizing a DeePMD-kit-generated deep learning potential to predict the thermal and mechanical properties of a high-entropy ceramic material. The scientists trained the deep learning potential on first-principles calculations to estimate the material's thermal conductivity, thermal expansion coefficient, specific heat capacity, Young's modulus, and Poisson's ratio. The authors claim that the deep learning potential methodology can efficiently and accurately anticipate the properties of high-entropy materials, which are difficult to examine using typical computational methods.

The procedure for using DeePMD involves some steps. We need to collect a dataset of molecular configurations and energies. This dataset is used to train the deep neural network. After that, a deep neural has to be trained to predict the potential energy of the system. The network is trained using the dataset collected in first step, and is used to predict the potential energy of new configurations of the system. Using the trained neural network to perform molecular dynamics simulations. The network is used to predict the potential energy of the system at each time step, and the equations of motion are solved using this potential energy. The key formula used in DeePMD is the equation of motion, which describes the motion of the particles in the system. The force is calculated using the potential energy predicted by the deep neural network, and the equations of motion are solved using this force. Atomic environment descriptors are used by DeePMD-kit as input for training a neural network. By establishing a cut-off radius and using the interactions of the atoms within it, the atomic environment is produced. DeePMD-kit preserves the system's rotational, translational, and permutational symmetry by generating descriptors through the atomic environment. For each atom, it records local coordinates by constructing local frames. The descriptor can be expressed as:

$$D_{ij}^{\alpha} = \left\{ \frac{1}{R_{ij}}, \frac{x_{ij}}{R_{ij}}, \frac{y_{ij}}{R_{ij}}, \frac{z_{ij}}{R_{ij}} \right\} (\text{complete information}).$$

$$D_{ij} = \frac{1}{R_{ij}} (\text{radial information})$$

Only radial information about the atoms is available when α is 0. While $\alpha = \{1, 2, 3, \text{full}\}$ offers radial in addition to angular information on the atoms. R_{ij} represents the relative location of atom i with respect to atom j , while (x_{ij}, y_{ij}, z_{ij}) are the coordinates of the atoms' relative positions. The total coordinate information describes the bound interactions between atoms, whereas the radial coordinate information describes the non-bonded interactions. The overall energy of the system is calculated by adding the atomic energies of all the atoms. The NN

trained on the descriptors predicts the atomic energies of the coordinates. DeePMD-kit seeks to minimize the loss function in order to maximize accuracy:

$$L(p_\varepsilon, p_f, p_\xi) = \frac{p_\xi}{N} \Delta E^2 + \frac{p_f}{3N} \sum_i |\Delta F_i|^2 + \frac{p_\varepsilon}{9N} |\Delta \Xi|^2$$

here ΔE , ΔF_i , and $\Delta \Xi$ are root mean square (RMS) errors of energy, force, and virial respectively. $p_\varepsilon, p_f, p_\xi$ are the perfectors which alter during the optimization process in training and obtained from the learning rate. TensorFlow's Adam stochastic gradient descent approach is used to optimize the model.

It's important to mention that DeePMD is a combination of two different models, one for the neural network and another for the MD simulations. The neural network model is a fully-connected feedforward neural network that receives the coordinates of the atoms of the system as an input, and outputs the potential energy of the system. The MD simulation model is a classical MD model, where the force is calculated from the potential energy and then the equations of motion are solved using an integration algorithm (such as Verlet algorithm).

The ML model is trained by transforming the generated training data to frames. Each coordinate frame is labeled with the energy and force associated with it. Along with coordinates, energies, and forces, DeePMD-kit requires the dimensions of the system's box and a file containing the system's atom kinds. After converting the training data to the right format, the NN is trained using input scripts that specify the training hyperparameters. The cut-off radius for the atomic environment, the number of hidden layers, and other training parameters are specified here.

E(3)-equivariant Graph Neural Network. Neural Equivariant Interatomic Potentials(NeQUIP) is a highly data-efficient deep learning technique for learning interatomic potentials from reference first-principles computations. This proposed method is claimed to

achieve greater precision than existing ML-IP methods for a wide range of systems, including tiny molecules, water in different phases, an amorphous solid, a reaction at a solid/gas interface, and a Lithium superionic conductor. Furthermore, NequIP is proven to demonstrate outstanding data efficiency, permitting the creation of accurate interatomic potentials from data sets including less than 1000 or as low as 100 reference ab-initio computations, whereas other methods require orders of magnitude more. Notably, on small molecular data sets, NequIP beats not just other neural networks, but also kernel-based techniques, which normally achieve higher predictive accuracy on short data sets than NN-IPs (although at significant additional cost scaling in training and prediction). From reference (16), a training set of molecular data obtained at the quantum chemical coupled-cluster level of theory is utilized to exhibit excellent data efficiency and accuracy. They were able to validate the method through a series of simulations and demonstrate that we can accurately duplicate the structural and kinetic parameters estimated from NequIP simulations in comparison to ab-initio molecular dynamics simulations (AIMD). They proved that the performance gains are a result of the new NequIP model's E(3)-equivariant convolution architecture.

Machine learning in atomistic systems uses equivariance because physical properties have well-defined transformation qualities under translation, reflection, and rotation of atoms. An equivariant transformation rotates a molecule's atomic dipoles or forces when rotated in space. Equivariant neural networks can better reflect physical system tensor characteristics and operations (e.g. vector addition, dot products, and cross products). Because they are expressly constructed from equivariant processes, equivariant neural networks ensure the transformation features of physical systems under coordinate changes. A function $f: X \rightarrow Y$ is equivariant with regard to the group G acting on X and Y if and only if:

$$D_y[g]f(x) = f(D_x[g]x) \quad \forall g \in G, \forall x \in X$$

Here where $D_x[g]$ and $D_y[g]$ are the group element representations in vector spaces X and Y , respectively. The goal is to determine the total potential energy E_{pot} and the forces acting on the atoms F_i , given a set of atoms. This total potential energy is calculated by adding the potential energies of individual atoms. The derivatives of this estimated total potential energy with respect to the atomic positions are then used to calculate the forces.

$$E_{pot} = \sum_{i \in N_{atomic}} E_{i,atomic}$$

$$\vec{F}_i = -\nabla_i E_{pot}$$

The local atomic energies The graph neural network predicts the scalar node characteristics $E_{i,atomic}$. Even while NequIP's output, the anticipated potential energy E_{pot} , is invariant under translations, reflections, and rotations, the network's core characteristics are geometric tensors that are equivariant to rotation and reflection. This is the fundamental distinction between NequIP and existing scalar-valued invariant graph neural network interatomic potentials(GNN-IPs). Each atom in NequIP is associated with characteristics consisting of scalars, vectors, and tensors of various orders. Formally, the feature vectors are geometric objects consisting of the direct sum of irreducible representations of the $O(3)$ symmetry group. The feature vectors $V_{acm}^{l,p}$ are indexed by keys l, p , where “rotation order” $l=0,1,2,\dots$ is a non-negative integer and parity is one of $p \in (1, -1)$ which together designate the irreducible representations of $O(3)$. The indices a, c , and m correspond to the atoms, the channels (elements of the feature vector), and the representation index $m \in [1, l]$ accordingly. The convolutions that work on these geometric objects are equivariant functions rather than invariant ones, meaning that if a feature at layer k is modified by a rotation or parity transformation, the output of the convolution at layer $k \rightarrow k+1$ is also transformed. Since their filters act on relative

interatomic distance vectors, convolution processes are intrinsically translation invariant. In addition, they are permutation invariant, as the sum of contributions from various atoms is invariant under permutations of those atoms. Notably, whereas atomic properties are equivariant to permutation of atom indices, the total potential energy of the system is permutation-invariant globally. To achieve rotation equivariance, the convolution filters $S_m^l(\vec{r}_{ij})$ are limited to be products of SO(3) equivariant learnable radial functions and spherical harmonics.

$$S_m^l(\vec{r}_{ij}) = R(r_{ij})Y_m^l(\hat{r}_{ij})$$

where if (\vec{r}_{ij}) represents the relative position from atom position i to neighboring atom j , \hat{r}_{ij} and r_{ij} are the associated unit vector and interatomic distance, respectively, and $S_m^l(\vec{r}_{ij})$ denotes the corresponding convolutional filter.

Most MLIPs use just invariant inputs to provide energy invariance. In invariant, atom-centered message-passing interatomic potentials, each atom's hidden latent space is a feature vector of only invariant scalars. Recently, equivariant neural networks have been created to directly respond on non-invariant geometric inputs such displacement vectors in a symmetry-respecting manner. Only E(3)-equivariant operations create a model that is equivariant with regard to the 3D Euclidean group. Interatomic potential models have been developed using equivariant topologies. The NequIP model, followed by numerous other equivariant implementations, showed unprecedentedly low error on a wide range of molecular and materials systems, accurately characterizes structural and kinetic features of complex materials, and has outstanding sample efficiency. In NequIP, the representation $D_X[g]$ of an operation $g \in O(3)$ on an internal feature space X is a direct sum of irreducible representations (irreps) of $O(3)$. The feature vectors are composed of geometric tensors corresponding to irreps that explain how they transform under symmetry operations. A tensor inhabits an irrep l, p if it transforms according to

its rotation order and parity. In many circumstances, one can skip the parity index and design features that are merely SE(3)-equivariant (translation and rotation), simplifying network construction and reducing memory needs.

An essential operation in such equivariant networks is the tensor product of representations, an equivariant operation that combines two tensors x and y with irreps ℓ_1, p_1 and ℓ_2, p_2 to produce an output residing in an irrep ℓ_{out}, p_{out} that satisfies $|\ell_1 - \ell_2| \leq \ell_{out} \leq |\ell_1 + \ell_2|$ and $p_{out} = p_1 p_2$:

$$(x \otimes y)_{\ell_{out}, m_{out}} = \sum_{m_1, m_2} \begin{pmatrix} \ell_1 & \ell_2 & \ell_{out} \\ m_1 & m_2 & m_3 \end{pmatrix} x_{\ell_1, m_1} y_{\ell_2, m_2}$$

here the products under multiplication is Wigner 3j symbol. This tensor product is bilinear (linear in both x and y) and combines tensors from separate irreps in a symmetrically valid manner.

Allegro. For my thesis, I mostly used Allegro. The original equivariant message-passing graph neural network model for force fields is NequIP (Neural Equivariant Interatomic Potentials). Allegro (17) is a newer equivariant model that is strictly local and thus massively parallelizable, allowing for quick and precise simulations. Both models are PyTorch-based and incorporate ASE and the LAMMPS (18) molecular dynamics code. For Allegro, the potential energy of a system is decomposed into per atom energies E_i .

$$E_{system} = \sum_i^N \sigma_{z_i} E_i + \mu_{z_i}$$

where σ_{z_i} and μ_{z_i} are perspecies scale and shift parameters. Allegro decompose the per atom energy into a summation of energy pairwise and it is indexed by the central atom and their local neighbors.

$$E_i = \sum_{j \in N(i)} \sigma_{z_i, z_j} E_{ij}$$

where j is the range of atom i 's neighbors, and again, a per-species-pair scaling factor σ_{Z_i, Z_j} may be applied. While these pairwise energies are indexed by the atom i and its neighbor j , they may be affected by all nearby atoms k in the immediate environment $N(i)$. E_{ij} and E_{ji} contribute to separate site energies, E_i and E_j , we chose to limit them to the surroundings of the corresponding central atoms. As a result of this and by design, $E_{ij} \neq E_{ji}$. Finally, the force acting on atom i , \vec{F}_i , is estimated using autodifferentiation as the negative gradient of total energy with respect to atom i 's position and this gives an energy-conserving force field:

$$\vec{F}_i = -\nabla_i E_{system}$$

Figure 4 depicts the Allegro architecture, which is an arbitrarily deep equivariant neural network with $N_{layer} \geq 1$ layers. The architecture discovers representations for ordered pairs of nearby atoms using two latent spaces : an invariant latent space with scalar ($\ell=0$) features and an equivariant latent space with tensors of arbitrary rank $\ell \geq 0$. At each layer, the two latent spaces interact with one another. A multi-layer perceptron (MLP) acting on the final layer's scalar features then computes the final pair energy E_{ij} . The notations that are used here is the following: \vec{r}_i is the position of the i th atom within the system. \vec{r}_{ij} is the relative displacement vector $\vec{r}_i - \vec{r}_j$ from i to j . r_{ij} is corresponding interatomic distance. $\vec{Y}_{\ell, p}^{ij}$ is the projection of r_{ij} onto the ℓ -th real spherical harmonic whose parity is $p = (-1)^\ell$. We omit the $m = -\ell, \dots, 0, \dots, \ell$ index from the representation of the compactness notation. Z_i is chemical species of atom i . $\text{MLP}(\dots)$ is multi-layer perceptron—a fully connected, possibly nonlinear, scalar neural network. $\mathbf{x}^{ij, L}$ is invariant scalar latent features of the ordered pair of atoms ij in layer L . $V_{n, \ell, p}^{ij, L}$ is equivariant latent features of the ordered pair of atoms ij at layer L . These transform according to a direct sum of irreps indexed by the rotation order $\ell \in 0, 1, \dots, \ell_{max}$ and parity $p \in -1, 1$; therefore, they consist of both scalars ($\ell = 0$) and

higher-order tensors ($\ell > 0$). The ℓ_{max} hyperparameter determines the maximum rotation order to which network features are truncated. In Allegro, n represents the channel index that ranges from 0 to $n_{equivariant} - 1$

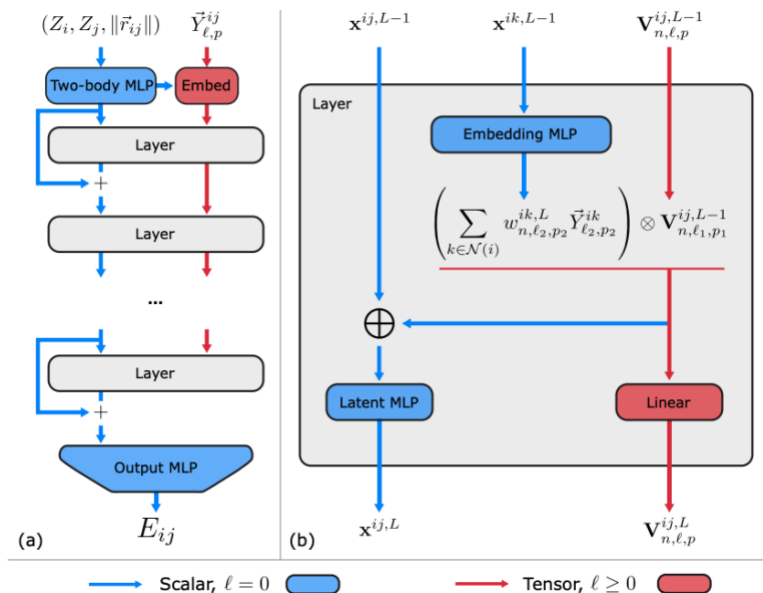


Figure 4. Allegro Network (a) Model architecture (b) Details of a tensor product layer

It omits the m index from the compactness notation for each irreducible representation. Each Allegro tensor product layer consists of the following four elements: An MLP that generates weights to embed the environment of the center atom, a tensor product using those weights that is equivariant, an MLP that updates the scalar latent space with scalar information derived from the tensor product and a linear equivariant layer that combines channels in the equivariant latent space.

The motive behind tensor product was to add interactions between the current equivariant state of the center-neighbor pair and other neighbors in the environment, and the tensor product is the most logical way to interact equivariant features. We therefore define the updated

equivariant features on the pair ij as a weighted sum of the tensor products of the current features with the geometry of the various other neighbor pairs ik in i 's local environment.

In Allegro, they use the advantage of the bilinearity of the tensor product to express the update in terms of a single tensor product, rather than one for each neighbor k , which significantly reduces the computational effort required. This is a variant of the "density trick."

$$\begin{aligned}
V_{n,(\ell_1,p_1,\ell_2,p_2)\rightarrow(\ell_{out},p_{out})}^{ij,L} &= \sum_{k \in \mathcal{N}(i)} w_{n,\ell_2,p_2}^{ik,L} (V_{n,\ell_1,p_1}^{ij,L-1} \otimes \vec{Y}_{\ell_2,p_2}^{ik}) \\
&= \sum_{k \in \mathcal{N}(i)} V_{n,\ell_1,p_1}^{ij,L-1} \otimes ((w_{n,\ell_2,p_2}^{ik,L} \vec{Y}_{\ell_2,p_2}^{ik})) \\
&= V_{n,\ell_1,p_1}^{ij,L-1} \left(\sum_{k \in \mathcal{N}(i)} w_{n,\ell_2,p_2}^{ik,L} \vec{Y}_{\ell_2,p_2}^{ik} \right)
\end{aligned}$$

Tensor product's second argument, $\sum_{k \in \mathcal{N}(i)} w_{n,\ell_2,p_2}^{ik,L} \vec{Y}_{\ell_2,p_2}^{ik}$ is a weighted sum of the spherical harmonic projections of the various neighbor atoms in the local environment. The atomic density is projected onto a weighted spherical harmonic basis, analogous to how ACE and SOAP project onto a spherical-radial basis. $\sum_{k \in \mathcal{N}(i)} w_{n,\ell_2,p_2}^{ik,L} \vec{Y}_{\ell_2,p_2}^{ik}$ is therefore considered to be the "embedded environment" of atom i . The scalar products of the tensor product are reintroduced into the scalar where \parallel denotes concatenation and \oplus denotes concatenation over all tensor product paths whose outputs are scalars ($\ell_{out} = 0, p_{out} = 1$), each of which contributes *nequivariant* scalars.

$$\mathbf{x}^{ij,L} = \text{MLP}_{latent}^L \left(\mathbf{x}^{ij,L} \parallel \bigoplus_{(\ell_1,p_1,\ell_2,p_2)} V_{n,(\ell_1,p_1,\ell_2,p_2)\rightarrow(\ell_{out}=0,p_{out}=1)}^{ij,L-1} \right) \cdot u(r_{ij})$$

The smooth cutoff envelope from equation is repeated as the function $u(r_{ij})$. The latent MLP's goal is to integrate and compress data from the tensor product of arbitrary dimensions into a latent space with a fixed dimension. Since the scalars extracted from the tensor product include

information about non-scalars that was previously only accessible to the equivariant latent space, this operation completes the coupling of the scalar and equivariant latent spaces.

After that, equivariant features are generated by linearly mixing the results from different paths of tensor product computations that share the same irrep (ℓ_{out}, p_{out}) . $V_{n,\ell,p}^{ij,L}$ with n channels, where n is the number of channels in the input features:

$$V_{n,\ell,p}^{ij,L} = \sum_{\substack{n' \\ (\ell_1, p_1, \ell_2, p_2)}} w_{n,\hat{n},((\ell_1, p_1, \ell_2, p_2) \rightarrow (\ell, p))}^L V_{\hat{n},(\ell_1, p_1, \ell_2, p_2) \rightarrow (\ell, p)}^{ij, L-1}$$

The weights, $w_{n,\hat{n},((\ell_1, p_1, \ell_2, p_2) \rightarrow (\ell, p))}^L$ are memorized. This operation, regardless of the number of paths, compresses the equivariant information from various paths with the same output irrep (ℓ, p) into a single output space. Finally, a SE(3)-equivariant version of Allegro, which is sometimes useful for computational efficiency, can be built in the same way as the $E(3)$ -equivariant model described here by simply omitting all parity subscripts p . Allegro employs a residual update in the scalar latent space after each layer, which updates the previous scalar features from layer $L-1$ by adding the new features. The residual update enables the network to easily propagate scalar information forward from earlier layers. A fully connected neural network is applied with output dimension 1 to the latent features output by the final layer to predict the pair energy E_{ij} :

$$E_{ij} = \text{MLP}_{\text{output}}(x^{ij, L=N_{\text{layer}}})$$

Moment Tensor Potentials. Moment Tensor Potentials are a class of systematically-improvable interatomic potentials invented at Skoltech by Shapeev et al. and implemented in the MLIP software package (19). MTPs are capable of actively selecting setups and configuring the potential while running. Demonstrably, MTPs accurately duplicate ab-initio-calculated energy, forces, and stresses (20). This section is largely influenced by Novoselov et al's outline.

As is the case with the vast majority of interatomic potentials, it is assumed that the overall interaction energy of a configuration may be represented as the sum of atomic contributions (21). The contribution from a single atom i is defined as $V(r_i)$, where V is the interatomic potential and $r_i = (r_{i,1}, \dots, r_{i,n})$ is a collection of vectors pointing from atom i to its neighbours within the potential cut-off. MTP then postulates linear representations for every atomic contribution $V(r_i)$.

$$V(r_i) = \sum_{j=1}^m \theta_j B_j(r_i)$$

Here θ_j can be adjusted, B_j defines as basis function and m is number of functions in that basis. The total energy of the system can be written as the sum of $V(r_i)$ over all atoms.

$$E(x) = \sum_{i=1}^N \sum_{j=1}^m \theta_j B_j(r_i)$$

N is the atom number in x configurations. The force can be determined as a derivative of $E(x)$ with respect to atom position and virial with respect to lattice vectors L .

$$f_j(x) = -\nabla E(x)$$

$$\sigma(x) = \frac{1}{|\det(L)|} (\nabla_L E(x)) L^T$$

According to equations above, the energy, forces, and stresses for a particular configuration are defined by the set of basis functions B_j and the values of the adjustable parameters θ_j . These parameters can be determined by fitting an unfitted MTP to the DFT calculation data. Assume that DFT computations were performed for some number of time steps on the training set, which is a collection of configurations X_{TS} . For every configuration x_i in the set X_{TS} , the "precise" energy ($E^{DFT}(x_i)$), per-atom forces ($f_j^{DFT}(x_i)$), and stress tensor components ($\sigma_j^{DFT}(x_i)$) are known. Hence, the MTP energy error for configuration x_i is

$$\Delta E(x_i) = |E(x_i) - E^{DFT}(x_i)|$$

We can minimize the functional to get the value of θ_j .

$$\sum_{x_i \in X_{TS}} \left[C_E^2 \Delta E(x_i)^2 + C_f^2 \sum_{j=1}^{N_i} \Delta f_j(x_i)^2 + C_s^2 \nabla \sigma(x_i)^2 \right]$$

C_E , C_f , and C_s weight energy, forces, and stress. These weights are modified during fitting to reduce inaccuracy. Fitting weights indicate property relevance during fitting. MTPs choose training set configurations based on per-atom energy ($V(r_i)$), configuration energy ($E(x_k)$), forces ($f_i(x_k)$), and stresses ($j(x_k)$). These traits are selected by neighbours, energy, forces, or stresses. Like the fitting process in eq., the strategies might be utilised simultaneously. Weights establish the relative importance of each strategy in the selection process. A threshold value controls the number of selected configurations for a given set of weights, limiting extrapolation. If the threshold is exceeded, the configuration is added to the optimised training set, hence decreasing the threshold increases the number of picked configurations.

Conventional machine learning is passive learning, in which an algorithm or machine learning model is fed data until convergence. In ambiguous regions, the model cannot discard points or seek new data. Active Learning (AL) tackles this problem by providing an unlabeled data set from which the model can query important points. This strategy is effective because all learners, whether algorithms or living organisms, gain from the capacity to explore, pose questions, and collect new information (22). The AL model requires an oracle, or instructor, to accurately identify the newly selected data points and quantify the uncertainty in its projection in order to maximise the data set's potential for extension and improvement. By accumulating more data, the pre-trained model is improved iteratively and by reading oracles provide data labels that may improve model performance (23).

AL's iterative method begins with a model trained on low-accuracy data or a data set that is too small to generalise the data trend. This model can be initialised at random or left empty to

avoid initial training data. This model predicts data without labels without training. In this phase, the model must be capable of estimating its prediction errors or extrapolation grade. These values can then sort the prediction data. The growth of training set. Then, a selection of these data points is chosen and the oracle is asked for the actual labels. After expanding the training sets with fresh tagged data, the model can be retrained to improve its performance when predicting near the spots. The parameter space for the model's projected data is scanned by repeatedly labelling high error data points and expanding the training set. The model will accurately reflect data patterns without major extrapolation or error.

Hyperparameter Optimization in Machine Learning Models. Machine learning entails predicting and classifying data and employing various machine learning models based on the dataset. Machine learning models are parameterized in order to be tuned for a specific problem. These models can have a large number of parameters, and determining the best combination of parameters can be viewed as a search problem. A model hyperparameter is an external configuration of the model whose value cannot be estimated from data. They are frequently used to estimate model parameter values. They are frequently optimized for a particular predictive modeling issue. One cannot determine the optimal model hyperparameter value for a given problem. We can use rules of thumb, copy values from other issues, or use trial and error to determine the optimal value. When a machine learning algorithm is tuned for a specific problem, you are essentially tuning the model's hyperparameters to determine which parameters produce the most accurate predictions. A popular book titled "Applied Predictive Modelling" states: "Many models have crucial parameters that cannot be directly estimated from the data. In the K-nearest neighbor classification model, for instance. This type of model parameter is known as a tuning parameter because there is no analytical formula to determine an

appropriate value." Frequently, model hyperparameters are referred to as model parameters, which can be confusing. The following is a good rule of thumb for avoiding this confusion: If a model parameter requires manual specification, it is likely a model hyperparameter (24).

Number of Hidden Layers. Hidden layers are those that exist between the input and output layers. A large number of hidden units within a layer using regularization techniques can improve accuracy. A lower number of units may result in underfitting.

Dropout. To boost its generalization ability, dropout is a regularization technique that reduces the likelihood of overfitting (improves validation accuracy). In most cases, a small dropout value of 20%-50% of neurons is appropriate, with 20% being a good baseline. If you set the probability too low, it won't have much of an impact, but if you set it too high, the network won't learn enough. When applied to a more extensive network, dropout increases the model's chances of acquiring accurate results by allowing it to form more distinct representations.

Network Weight Initialization. It's possible that different weight initialization schemes would be optimal for each layer's activation function. Typically, a fairly even distribution is employed.

Activation function. Activation functions are used to introduce nonlinearity into models, enabling deep learning models to discover nonlinear prediction boundaries. In general, the most popular function is rectifier activation. In the output layer, sigmoid is used when making binary predictions. Softmax is utilized in the output layer when making predictions for multiple classes.

Learning Rate. The learning rate specifies the rate at which a network modifies its parameters. The learning process is slowed by a low learning rate, but it converges smoothly. A higher learning rate accelerates the learning process, but it may not converge. Typically, a degrading Learning rate is favored.

Number of Epochs. The number of epochs represents the number of times the entire training set is presented to the network during training. Increase the number of epochs until validation accuracy begins to decline even as training accuracy rises (overfitting).

Batch size. Mini batch size is the number of subsamples provided to a network before a parameter update occurs. The default batch size could be 32 items. Additionally, one can try 32, 64, 128, 256, etc.

METHODS

High Entropy Alloy Structures for Ab-initio Simulation

We need to generate structures for computing the DFT and ab-initio calculation. It is important to have all the possible combinations of samples. The basic strategy we adopted in this study shown in Figure 5, which we dubbed the "high-entropy strategy" is a sampling technique that aims to maximise the statistical variance in volumetric/pressure and energy environments of the phase of interest (25). We started with HEA because we can easily exchange information to other binary rich or pure rich composition. Such a method may also be extended to higher-order systems by sampling the high-entropy alloy (HEA) of different phases, which include a relatively equal proportion of transition metals.

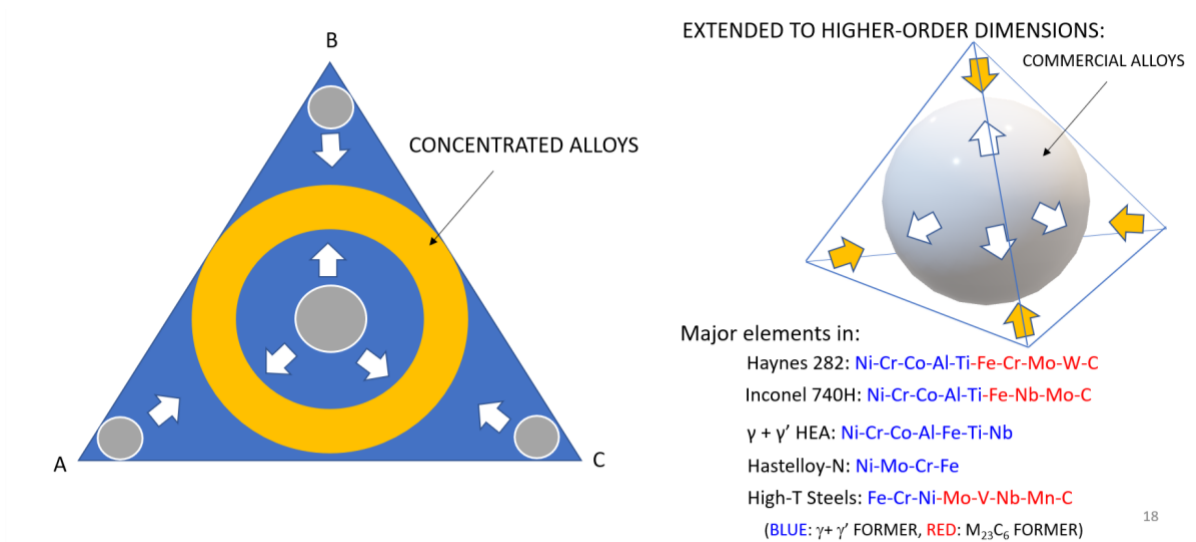


Figure 5. High entropy alloy strategy

The complete phase space is sampled in a different manner by expanding a large range of simulated temperatures and volumes at a given temperature. I studied the various temperature–

volume domains by randomising the transition metal in order to discover more comprehensive compositional changes. I varied the Vienna Abinitio Simulation Program(VASP) input file, POSCAR which contains lattice geometry and ionic positions and changed the universal lattice scaling vector from 0.95 to 1.2. Different volumes were made either by scaling up or down by increasing and decreasing the relaxed structure.

In each sampled set of trajectories, the early frames are purposefully trimmed and not used for training or testing due to erroneous energy considerations caused by the normally randomly assigned initial velocity, hence enhancing the quality of the data as a whole. In my thesis, I am going to discuss potentials of 4 to 10 components HEA including Ni, Cr, Co, Al, Fe, C, Mo, W, Ti, Nb. The DFT calculation for these composition were done at various temperature and volumes. The samples also had pure rich compositions of HEA including FCC, BCC, HCP and random phases. Starting configurations for first-principle molecular dynamics (FPMD) are derived from Materials Project crystallographic data in cif format for FCC, BCC and HCP phases, shown in Figure 6.

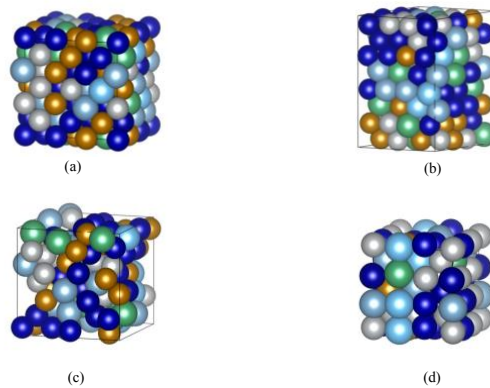


Figure 6. Input structures for the DFT calculation. (a) FCC (b) HCP (c) Random/melt (d) BCC

These unit cells for *AB-INITIO* MD simulations have 50~120 atoms per unit cells. For each HEA composition, training data are created in accordance with the VASP documentation for canonical ensemble (NVT) FPMD sampling, including recommended GPAW pseudopotentials(Table1). To ensure reliable sampling, and given that explicit relaxation data are not sampled, a sufficient kinetic energy cutoff of 385~520 eV is determined for different compositions, with the greatest of each species' pseudopotentials being chosen. The self-consistent field(SCF) iteration threshold is set at atleast 1E-05 eV, and the precision was accurate. Given a huge cell size of atleast 108 atoms, real-space projectors and distributions are enabled using mostly 1x1x1 and 2x2x2 KPOINTS grid for efficient data creation and some pure rich configurations with 3x3x3 KPOINTS.

Table 1. Selection of ab initio data used for model training of HEA 4, 5, 7

| System | Temperature(K) | Frames |
|--------|----------------|--------|
| HEA 4 | 1000~5000 | 148000 |
| HEA 5 | 1000~2500 | 2110 |
| HEA 7 | 1000~3000 | 25764 |
| HEA 10 | 1000~5000 | 200857 |

After sampling different volumes in different temperature, the samples needs to process for obtaining the behaviour of these structures using ab-initio molecular dynamics with VASP. The goal of performing ab-initio molecular dynamics on the crystal structure is to provide information on energy, force, and strain at high temperatures. VASP supports many thermostat types in order to accomplish the NVT ensemble. The kinetic energy cutoff was set higher than the maximum energy cutoff for the species they had. This energy setting was used for all NVT

ensemble abinitio molecular dynamics (AIMD) calculations. The maximum number of self-consistent field (SCF) iterations to perform to converge the electronic structure was kept at least 40. The field convergence of $1.0\text{E-}6$ eV was employed. Similarly, for partial occupancies of a set of orbitals, Gaussian smearing with a smearing width of 0.05 eV was set. There are multiple methods for obtaining samples for various configurational spaces inside the structure and canonical NVT ensemble temperatures that would address the harmonic and anharmonic vibrations in the crystal structure. Therefore, the typical technique here is to first match the thermal fluctuation caused by harmonic vibration and then go to a higher temperature for sampling the anharmonic contribution for the specified configurational space. I use this method and run NVT ensemble simulations at temperatures of 1000K to 4000K for all given structures with volume scaling and deformations.

Machine Learning Interatomic Potentials Training

After ab-initio calculation of the HEA samples, the energy, force, and stress data must be fitted. For each ab-initio VASP computation, a vasprun.xml file containing energy, force, and stress information from molecular dynamics run is generated. NVT ensemble samples of hundreds of trajectories were produced for different temperature. The vaprun.xml file contained volume scaling for these trajectories. Each structure's trajectory was formed by altering lattice settings and randomly dislodging atoms. I will discuss results including three different NN build potentials. First, I used Deep Potential—Smooth Edition (DeepPot-SE) neural networks with DeePMD package. To improve the predictability of elastic characteristics, radial and angular descriptors from input frames are considered. Afterwards, further training parameters are determined based on prior literature and potential applications. Hence, according to (27), three-

layer embedding and fitting networks consisting of [25, 50, 100] and [240, 240, 240] neurons are used. A cutoff radius of 5.8-6 Å guarantees significant neighbor interactions for up to the third closest neighbor in each training dataset. I have chosen three cut off radius like 2, 3 and 5 Å to see how well they predict the data. In Table 2, I am showing the RMSE error evaluation. I got better results with 6 for most of our systems. The choice of cut-off radius in neural network potential models specifies the maximum distance between atoms that will be considered when calculating atomic interactions. The choice of cut-off radius must strike a compromise between including enough surrounding atoms to capture all significant interactions and include too many atoms to render the computation impractical. A cut-off radius of 6 Å likely produced better findings than a cut-off radius of 3 or 2 Å because it includes more nearby atoms in the energy calculation. A bigger cut-off radius permits interactions between a greater number of atoms, which can result in a more precise energy computation. However, expanding the cut-off radius excessively may also include irrelevant interactions (8 Å), leading to overfitting or slower calculation. Consequently, the choice of cut-off radius must be meticulously optimized based on the particular system and the available computational resources. Virial perfactors that are not zero are utilized to explicitly train the elastic data contained inside the datasets. The starting and end energy, force, and virial perfactors for one to five million training steps are [0.02, 1], [1000, 1], and [100, 1], respectively. For Nequip or Allegro, the network architecture parameter was chosen in a similar way. However, I varied the cut off radius from 4 to 6 Å for this NN so that I can visualize how hyperparameters' values affect the results. I have used more than 4 number of layers for both of these NN codes. In the context of NequIP, I've selected "l_max: 2", which refers to the maximum quantum number of angular momentum included in the basis functions used to characterize the atomic environment.

Table 2. Cut off radius and error

| Cut off radius(\AA) | Energy RMSE/Natoms(eV/A) |
|--------------------------------|--------------------------|
| 8 | 6.459E-03 |
| 6 | 6.000E-03 |
| 3 | 1.087E-02 |
| 2 | 2.370E-01 |

This expansion is utilized to construct the neural network's input features, which are then used to estimate the potential energy surface. I look through the examples of training data input and scripts from (17) and choose parameters that gave me lowest error. The codes mentioned above had some type of inputs for all the compositions. For MTP, I had worked with OUTCAR which is a detailed output of a VASP run, including: a summary of the used input parameters, information about the electronic steps, KS-eigenvalues, stress tensors. forces on the atoms. I eliminated configurations where the smallest distance between atoms is less than 1.5. (19) has 28 untrained MTP potential input files to choose from and I select the level 10 functional form of MTP, eight radial basis functions, and maximum cut off radius of 5 \AA . After training data with these codes, I decided to verify and validate these potentials. For each of these MLIPs, I have validated them by evaluating how good they predict the input data such as energy, force and stress of the systems. I have used larger scale MD simulations to verify how well these MLIPs predict mechanical and thermal properties of the given systems.

RESULTS AND DISCUSSION

DeePMD Model Accuracy

The model accuracy was evaluated on how good the potentials were able to predict the input data. I am going to discuss results of three different number composition-based HEA. HEA 4 is made of equal portion of Ni, Co, Cr and Al. HEA 7 is with Ni, Co, Cr, Fe, Al, Ti, Nb and HEA 10 which means 10 elements is made of Cr-W-Mo-C-Ni-Co-Fe-Al-Ti-Nb. Average RMSE training values for HEA4 from energy per atom, force, and virial per atom are $6.000\text{E}-03$ eV, $4.759\text{E}-1$ eV/Å, and $2.414\text{E}-2$ eV respectively, after 1750,000 training steps which is shown in Figure 7.

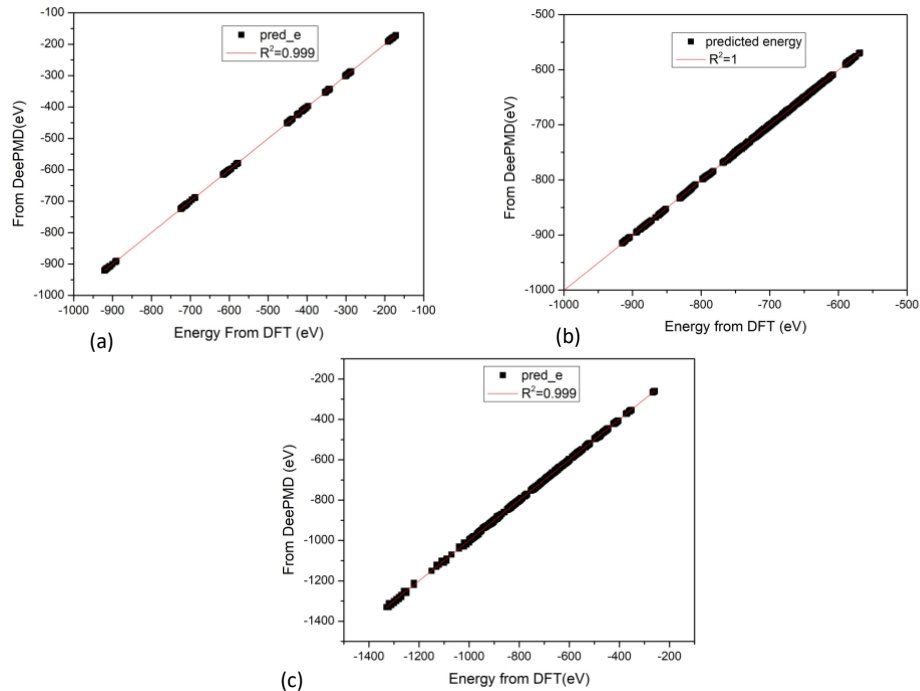


Figure 7. Energy matching by DeePMD of (a) HEA4 (b) HEA7 (c) HEA10 as predicted by DeePMD potentials in y axis vs training-DFT data(x-axis)

These training errors are within the same order of magnitude of other published work (26). Additionally, given inclusion of virial prefactors in this computation, it is reasonable that the total RMSE error is bigger than reported in other literature, where just force and energy are employed for training. Energy and virial matching in Figure 8 are linear in character, with a correlational coefficient of 0.999 and 0.999, respectively. I will discuss the force RMSE along with Allegro result as we found out better force convergence in equivariant NN. The energy and virial figures with these 3 different composition data with such near-linear connections show that the model well matches learned data. Numerical RMSE findings reveal that the model is typical of the trained ab-initio datasets.

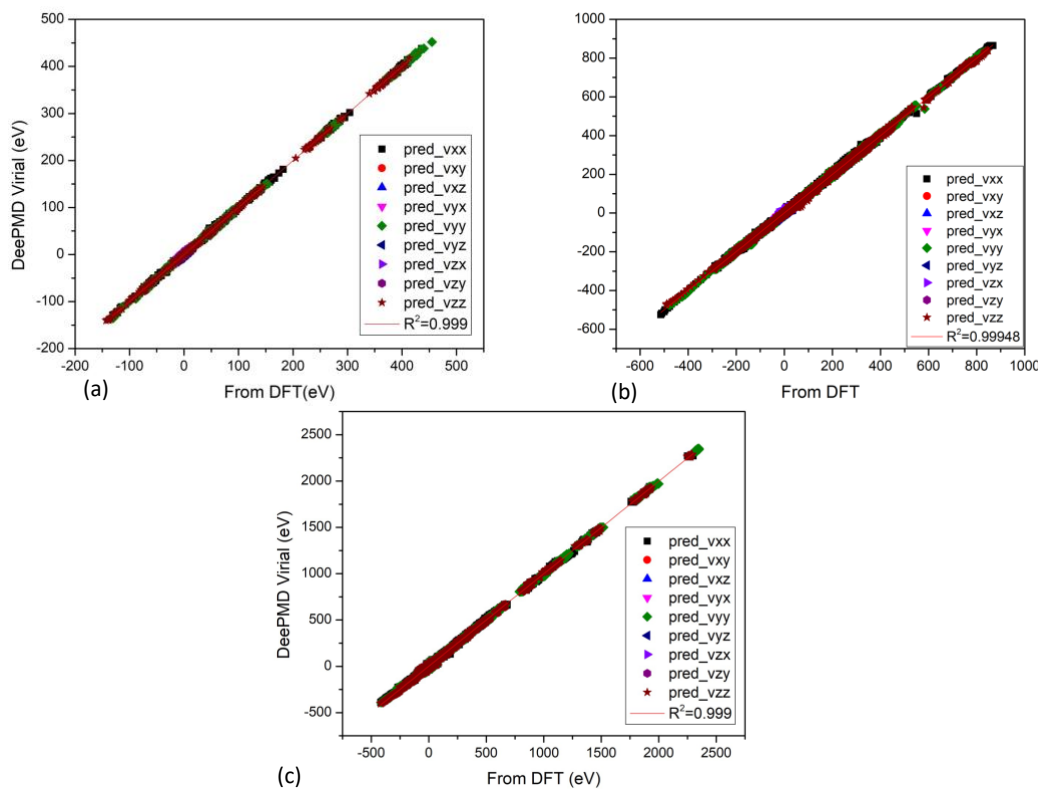


Figure 8. Virial matching by DeePMD potentials of (a) HEA4 (b) HEA7 (c) HEA10 as predicted in y axis vs training-DFT data(x-axis)

The model accuracy for all of these 3 different elements systems are shown in Table 3. We can see that we got the most accuracy in prediction of energy and lower accuracy in terms of force prediction. Due to this reason, we opted to use new equivariant NN code which gave good accuracy to all of these three components. Additionally, DeePMD potential robustness is reflected in molecular dynamic simulations which sample untrained settings and properties- serving to additionally check overfitting has not happened.

Table 3. Error Analysis of three Models of DeePMD

| System Name | Energy RMSE (eV/atoms) | Force RMSE (eV/atoms) | Virial RMSE (eV/atoms) |
|-------------|---------------------------|--------------------------|---------------------------|
| HEA04 | 6.000E-03 | 4.759E-01 | 2.414E-02 |
| HEA07 | 8.044E-03 | 5.867E-01 | 3.648E-02 |
| HEA10 | 1.542E-02 | 4.894E-01 | 3.928E-02 |

DeePMD Model Verification Using MD Simulations

The microstructure of Nickel-based alloys consists of a primary matrix with embedded secondary precipitates, the stress-strain as well as the creep deformation behavior at different physical conditions and environments depend on the microstructural characteristics. The mechanical behavior of these alloys is dependent on the volume fraction, shape and size of the precipitates, defect kinds and quantity, and their dislocation motion resistance. In this study, compression experiments were conducted to examine the dislocation dynamics in relation to the creep deformation of nickel-based superalloys with varied parameters. Figure 9 illustrates the uniaxial compression test at strain rate of 10^{10} /s at 4 different temperatures using DeePMD

potential of 10 component systems. We have a replica simulation cell of almost half a million atoms of gamma phase like Haynes 282 alloy composition. I performed the constant pressure test (NPT) at 4 temperatures with same strain rate. If we compare with experimental results (27), we can see that in Figure 9 (b) that the ultimate tensile stress of the curves is decreasing with increase of temperature. Also, 9(c) includes the partial dislocation along (111) slip planes and loops that forms at the end of the simulation. This indicates that the potential was able to predict the mechanical behavior under compression test for one phase system.

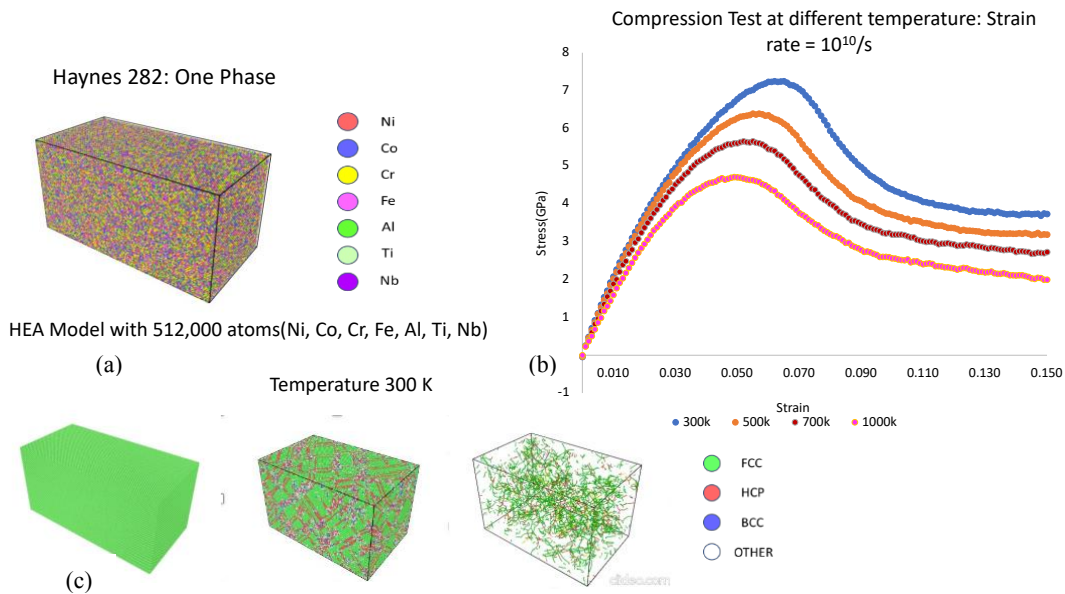


Figure 9. Compression test under various temperature using HEA10 potential (a) the model (b) stress vs strain graph (c) the dislocation band generation

The next simulation cell that I used under the uniaxial compression test at 300K temperature with strain rate of $5 \times 10^9/s$ was multilayer of Chromium Carbide ($Cr_{23}C_6$) with Gamma-Gamma prime in between. We have Ni as the FCC phase and Ni_3Al as the gamma prime. In figure 10(a) and 10(b) showing the simulation cells with atoms whereas the 10(b) is the

condition at the end of the simulation. The rest of the figures are same, but they are without atoms for FCC phase. As FCC is the softest phase here, we can only see dislocation forming in this phase. The carbide multilayer is harder than the FCC phase and it's difficult for the dislocation to pass in this layer. The DeePMD potentials with 10 components were able to predict this phenomenon.

I will discuss thermal stability of the DeePMD potentials. During NPT simulations, thermal expansion is examined by alternating stages of heating and equilibration. Figure 11 demonstrates the expected linear trends for the carbides in the low-temperature range (300–1500 K), as validated by high correlational coefficients in linear regression.

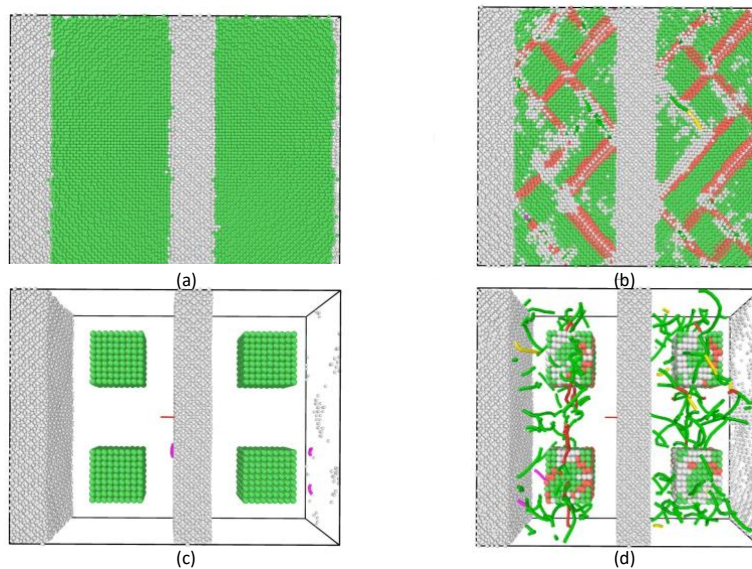


Figure 10. Compression test under various temperature using HEA10 potential (a) the model (b) Stress vs strain graph (c) the dislocation band generation

This qualitative behavior is anticipated in this temperature regime, given that the structure is stable and must exhibit low anharmonicity. Figure 11 shows the coefficient of thermal expansion (CTE) of HEA of 4 and 5 components (Ni, Al, Cr, Co, Fe). I used NPT

simulation from temperature 300K to 600 K with increment of 50K. The CTE from my simulation is similar to the reference which indicate this forcefield is simulating the material's underlying physics and interactions. I also computed same NPT test on single phase Cr₂₃C₆ from 100K to 1500K temperature using HEA10 potentials and compared with our previous potential of binary carbide. During NPT simulations, thermal expansion is examined by alternating stages of heating and equilibration. Figure 12 demonstrates the expected linear trends for the carbides in the low-temperature range (300–1500 K), as validated by high correlational coefficients in linear regression. This qualitative behavior is anticipated in this temperature regime, given that the structure is stable and must exhibit low anharmonicity.

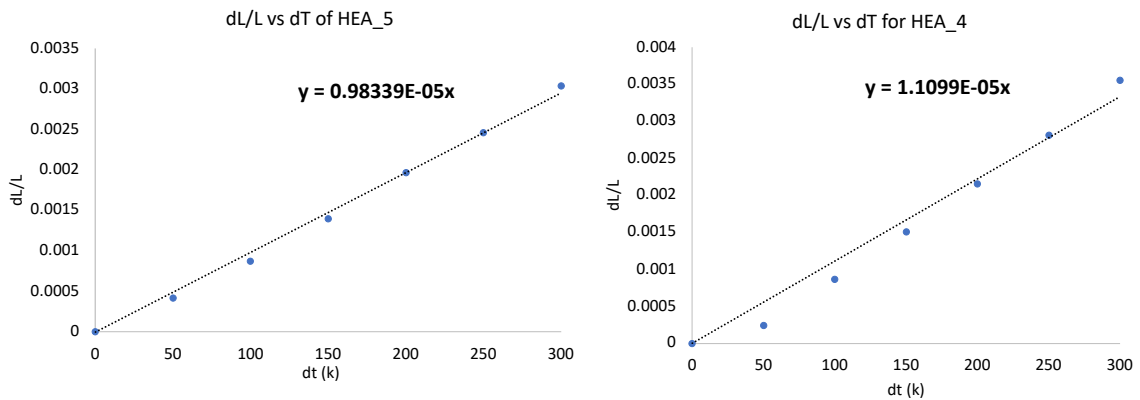


Figure 11. Coefficient of thermal expansion at 300K using HEA7 potential (a) HEA 5(Ni,Al,Co,Cr,Fe) (b) HEA 4(Ni,Al,Co,Cr)

The eventual modeling of creep deformation behavior requires an accurate prediction of elastic characteristics. Using LAMMPS, elastic constants and associated characteristics are calculated and compared to known literature for a near match. Developed potentials accurately reflect the following desirable elastic properties of Ni, Ni₃Al, Cr₂₃C₆, Fe₂₃C₆ in Table 4: ground-state elastic constants, Poisson Ratio, and Bulk Modulus. Interestingly, the elastic characteristics

are not explicitly sampled in the training dataset, and only NVT simulations are employed; despite this, correct predictions are made.

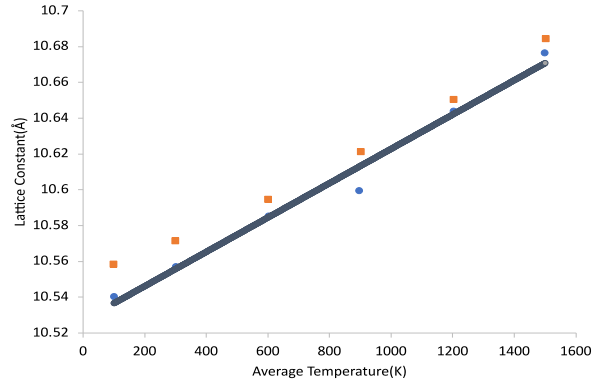


Figure 12. Predicted average lattice constants of Cr_{23}C_6 varying by temperature. The orange boxes are using HEA10 potential and blue dots are using (Fe–Cr–Mo–W–C) potential.

Table 4. Predicted CIJ

| Systems | C_{11} (GPa) | C_{12} (GPa) | C_{44} (GPa) | η | K (GPa) |
|----------------------------|----------------|----------------|----------------|------------|-------------|
| Cr_{23}C_6 | 437.52397 | 244.906 | 136.809 | 0.3588 | 309.112 |
| | 471.71(25) | 232.48(25) | 136.34(25) | 0.3301(25) | 312.221(25) |
| | 472.15(28) | 215.7(29) | 135.1(29) | 0.3141(28) | 301.01(29) |
| Fe_{23}C_6 | 405.561 | 285.1433 | 100.421 | 0.4107 | 315.5217 |
| | 466.84(25) | 301.42(25) | 68.79(25) | 0.3923(25) | 356.557(25) |
| | 490.7(29) | 255.9(29) | 133.8(29) | 0.34(29) | 334.2(29) |
| Ni | 235.5085 | 175.9411 | 121.617 | 0.4276 | 195.796 |
| | 276(30) | 159(30) | 132(30) | 0.29(30) | 198(30) |
| Ni_3Al | 202.679 | 145.101 | 107.134 | 0.4172 | 164.293 |
| | 239(30) | 150(30) | 129(30) | 0.30(30) | 180(30) |

The average percent error for all elastic are quite low, with near agreement with (15, 16). Thus, trained DeepPot-SE type potentials may forecast the ground- state elastic constants and bulk elastic characteristics of interested binary systems, pure system within ab initio ranges. The only exception is the C_{44} value of Al. This issue may be solved by further adding more adding training data or tweaking model architecture and hyperparameters. We tried our potentials to find out melting of different systems. Figure 13 illustrates this behaviour for the Fe_{23}C_6 system as determined by NPT simulation with $5\text{E-}4$ picosecond timesteps for an initial equilibration of 1000 steps and additional heating for 49,000 steps to 3500 K.

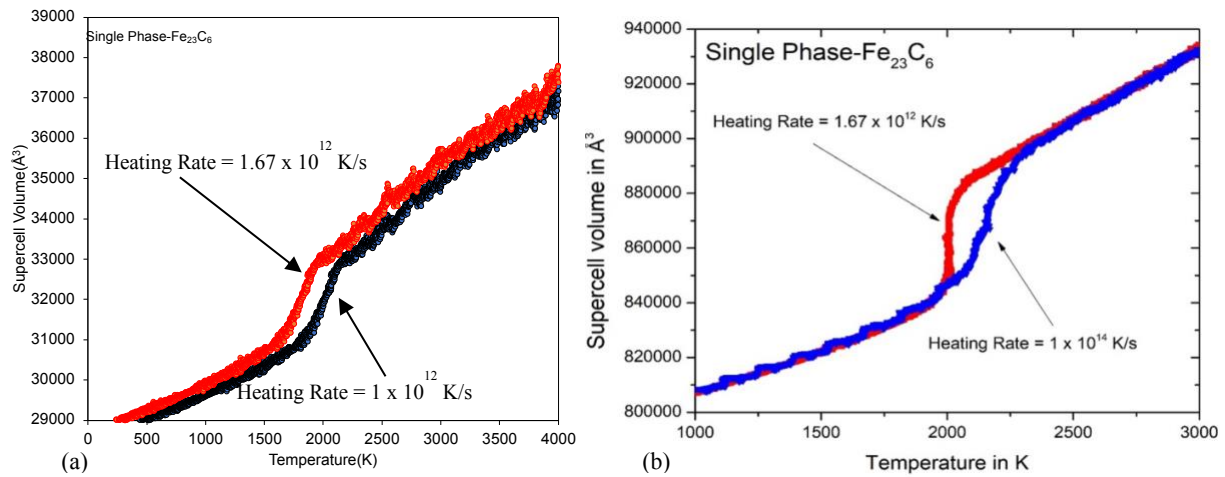


Figure 13. Melting point of single phase Fe_{23}C_6 at two different heating rate using (a) HEA10 potential(Ni,Al,Co,Cr,Fe,C,Mo,W,Ti,Nb) (b) Fe_{23}C_6 potential

Before and after the phase transition, the system exhibits linear thermal expansion, with a nearly discontinuous change in volume at the melting point. Thus, MD provides a signal at 2000 K for the melting point. We compared this with binary carbide components potentials (25) of two different heating rate and the results came out in similar temperature range. The same NPT melting test was done on pure systems like Ni, Al of 4000 atoms and in Figure 14, we can see the

melting point of Ni using HEA10 deep potential came out to be 1900K which a bit higher than (31). The Al melting point came out near 860K which is in near range to the melting point mentioned in (32)

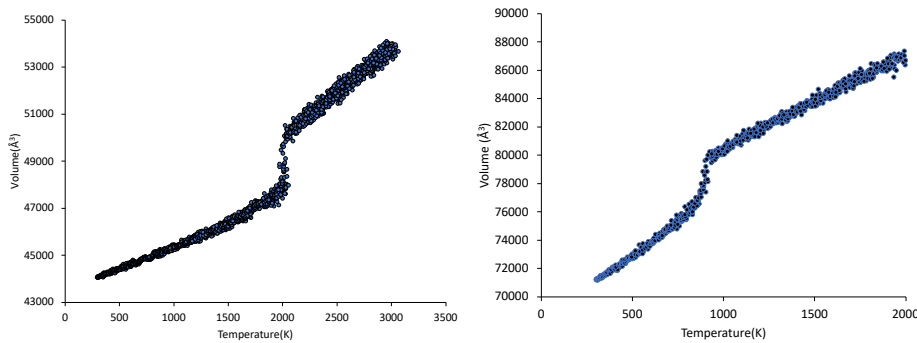


Figure 14. Melting point of single phase (a) Ni at 1900K (b) Al at 850K

We also investigate dislocation behavior under varying stress for a simulation cell of similar to INCONEL 740H. I created a one phase structure with prolonged length in the Y axis and dislocation in between this length which will move the dislocation to the X axis. The tutorial from LAMMPS official page (33) using OVITO (34) was followed and in Figure 15, we can see the dislocation position and velocity with varying stress. When an extra half-plane of atoms disrupts the crystal lattice, dislocations form. Dislocations travel within the crystal structure, creating plastic deformation when a material is stressed. Atoms move dislocations, which takes energy. Low tensions make dislocation movement harder since the energy required to move it is large. When material stress increases, the energy required to move the dislocation reduces, making it easier to move. With larger stresses, the dislocation exerts more force, making it simpler to overcome crystal structural energy barriers. Increased force makes the dislocation move quicker and farther before hitting a barrier. Under larger stresses, crystal structure atoms

are more stimulated and more likely to engage in dislocation motion. This speeds the dislocation across the crystal structure. That's why in figure 15 (b) we can see with increasing stress the velocity is increasing linearly.

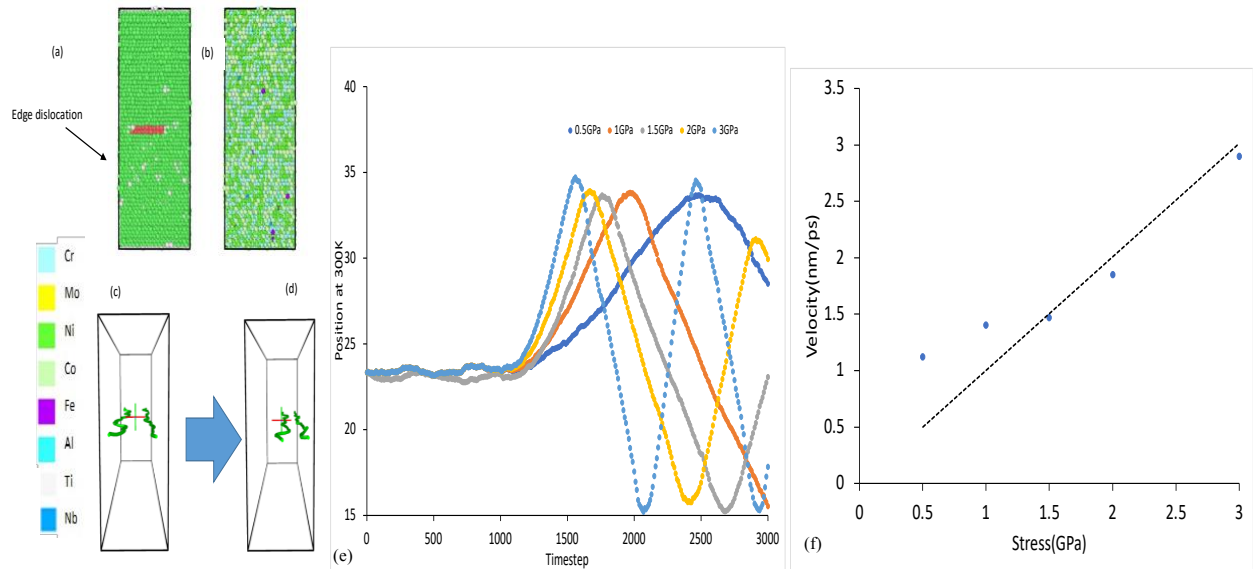


Figure 15. Dislocation mobility in INCONEL 740H using Deep Potential (a) dislocation in the beginning (b) the model at the end (c) dislocation loop at beginning (d) at end (e) positions with time (f) velocity of the dislocation with varying stress

DeePMD Model Accuracy with Higher K-Points Input

To evaluate how this NN perform with higher quality data, I have used HEA7 DFT data with 3x3x3 K-points, same configuration of the previous data. Due to higher K-points, I got trajectories near 6263 frames whereas the previous data with low K-points has 24154 frames. Though the comparison between this two data set model is not fair, I still trained the higher quality data with same configuration. In Table 5, the RMSE error are shown. Both of the RMSE are similar which indicate that with lower data of higher K points, we can achieve better results using DeePMD.

Table 5. Error analysis of HEA7

| System Name | Energy RMSE (eV/atoms) | Force RMSE (eV/atoms) | Virial RMSE (eV/atoms) |
|-----------------------|---------------------------|--------------------------|---------------------------|
| HEA07 higher K points | 6.400E-03 | 5.031E-01 | 3.593E-02 |
| HEA07 | 5.510E-03 | 5.891E-01 | 3.242E-02 |

Allegro Model Accuracy

Similar to DeePMD model, I evaluated Allegro by comparing the predicted energy, force, virial output data with DFT inputs. In Allegro, I choose equivariant NN. The choice of cut off radius from 4 to 6 Å with two different learning rate and batch size of 0.001, 0.005 and 1,3 respectively. The goodness of fit in this comparison graphs in Figure 16 shows good accuracy as the predicted values are with goodness of fit for the input data. In Table 6, we can see how the force RMSE errors are less than DeePMD force RMSE, mentioned in Table 3. The configuration of hyperparameters will be discussed later which were same for some of them compared to DeePMD’s hyperparameters. The energy prediction with Allegro is similar to DeePMD except for the 10-component system. RMSE is only one of many possible evaluation measures, and other metrics may provide different insights into model performance. In addition, the performance of Allegro model can be affected by a variety of variables, such as the method and hyperparameters employed, and the nature of the issue being solved. When it comes to comparing the performance of different models, it is essential to conduct a comprehensive evaluation to draw meaningful conclusions. A thorough evaluation typically involves testing the models on various datasets and using various evaluation metrics to assess their performance. In terms of RMSE, Allegro performed better than DeePMD in terms of force RMSE.

Table 6. Error analysis of three models of Allegro

| System Name | Energy RMSE (eV/atoms) | Force RMSE (eV/atoms) |
|-------------|------------------------|-----------------------|
| HEA04 | 7.000E-03 | 1.082E-01 |
| HEA07 | 5.914E-03 | 1.951E-01 |
| HEA10 | 7.920E-03 | 1.034E-01 |

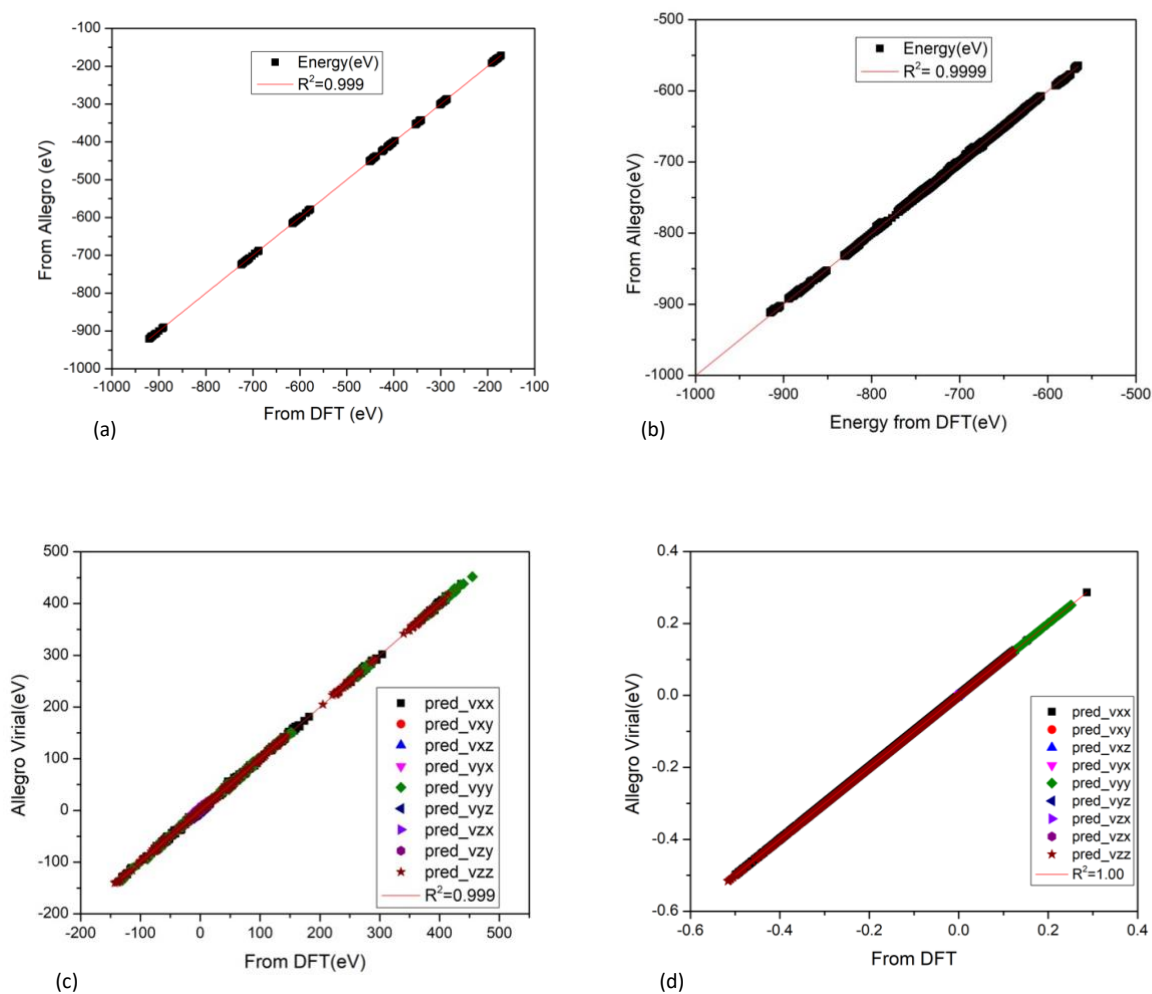


Figure 16. Energy and virial matching by Allegro of (a) HEA4 energy (b) HEA7 energy (c) HEA4 energy (d) HEA7 virial as predicted in y axis vs training-DFT data(x-axis)

We have been getting higher RMSE error in force with invariant neural networks like DeepMD. That was one of the reasons that we chose Allegro. In Figure 17, one can see that the goodness of fit of the graph is better in Allegro, R^2 value near 1 and for DeePMD, $R^2=0.87$. If we evaluate force RMSE for HEA of 4 components in Figure 17 (c), the error is less than 0.2 eV

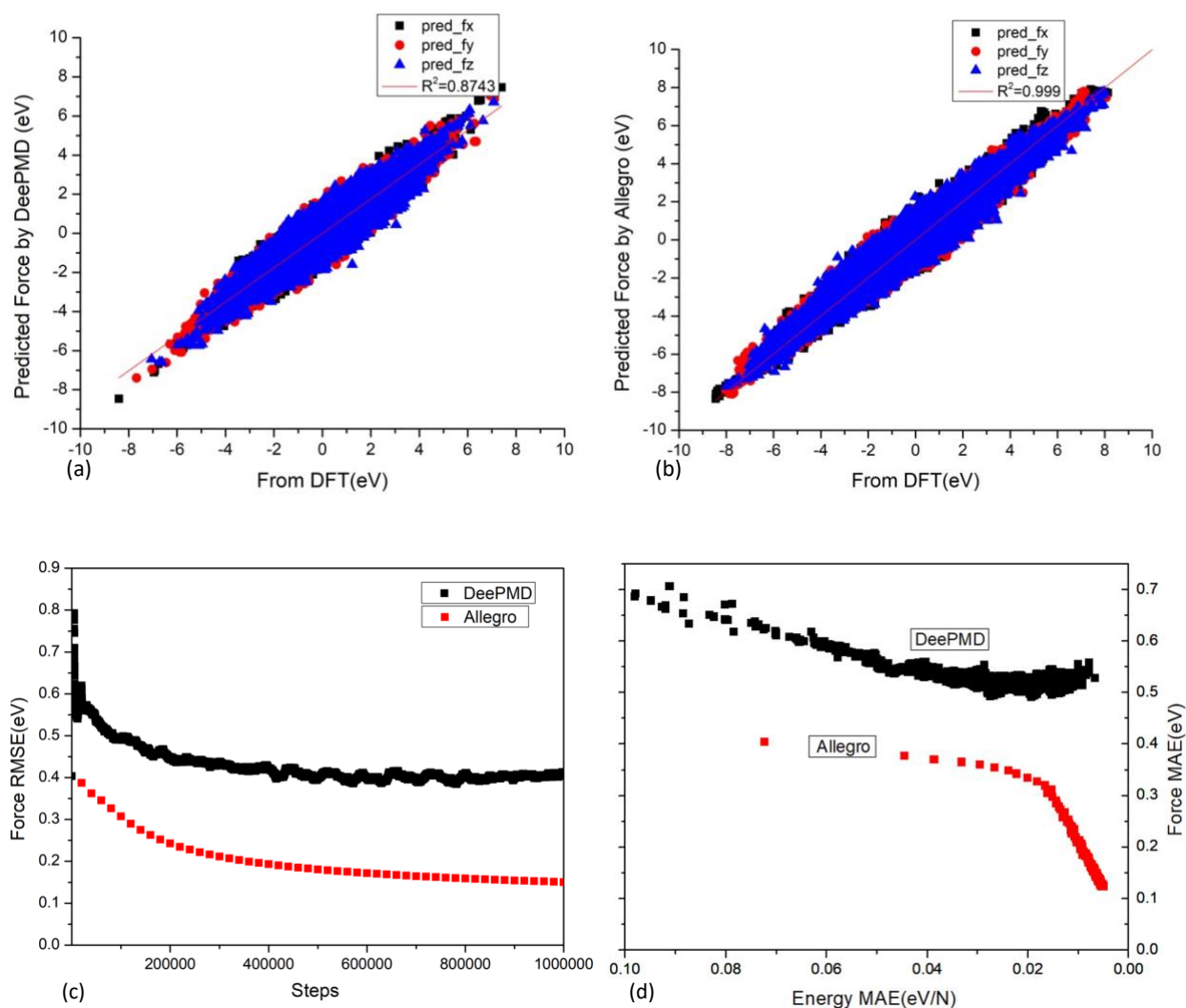


Figure 17. Force of HEA with 4 components (a) HEA4 Predicted force by DeePMD (b) HEA4 Predicted force by Allegro (c) HEA4 Force RMSE Comparison (d) HEA4 force convergence with energy

for Allegro. DeePMD show comparatively higher error than Allegro which indicates the effectiveness of equivariant NN in capturing the essence of input component, force. Also Allegro has potential to get lower RMSE with enough training and hyper parameters optimization.

Allegro Model Verification Using MD Simulations

I was able to try NVT test using Allegro Potentials and the results that I am going to show is mostly stability under 300K and 1000K of different structures and phases. In Figure 18, there's two different phases, the first one is gamma phase of HEA 4 near 7000 atoms and second structure represent gamma gamma prime phase. The figure shows their stable condition at different temperatures with radial distribution function(rdf) (35).

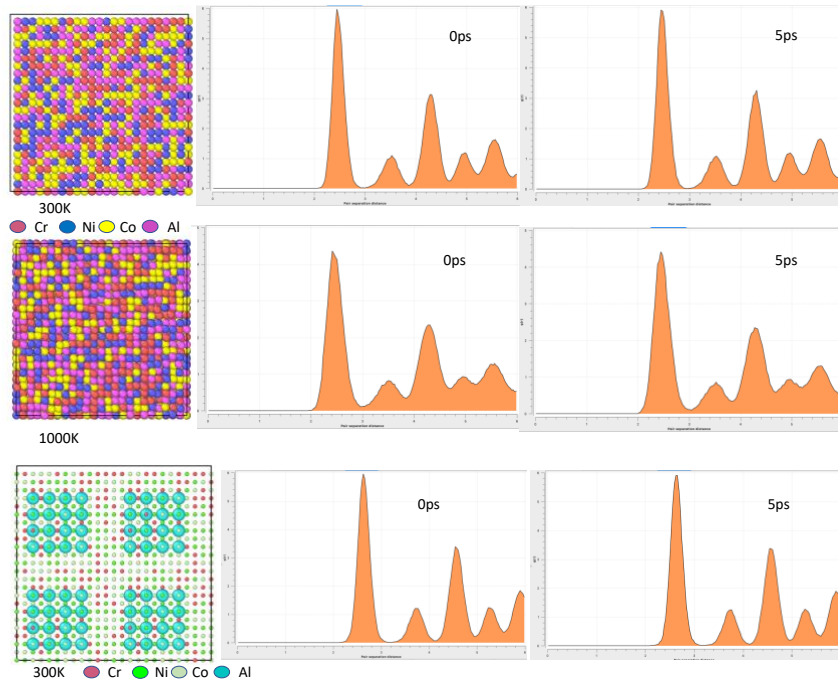


Figure 18. Thermal stability of supercell models of Gamma phase and Gamma Gamma Prime phase of HEA 4 at different temperature using HEA 4 Allegro Potential

It follows that the RDF can be used to monitor the melting process as for any given cluster size, the RDF will give an indication of stability and melting behavior. We can see that by coordination analysis, the first peak can be found and when these sharp peaks get neutralized; gets repetitive shape so the RDF plot can be assumed the characteristic shape of a typical fluid function. If $g(r)$ remains constant with pair distance for 5000 frames at a specific temperature, this indicates that the system is in equilibrium and has a stable structure that is maintained throughout time. In other words, the particles are ordered in a manner consistent with the system's circumstances (e.g., temperature, pressure, etc.), and this arrangement does not change over the observation time. The precise value of $g(r)$ at this temperature and for this system would depend on the nature of the particle interactions, the system's density, and other relevant considerations. Yet, the fact that $g(r)$ remains constant over 5000 frames shows that the particle arrangement or structure of the system does not change significantly over time.

Allegro Data Efficiency

Nequip/Allegro uses the symmetry and invariance of the physical systems being described to get the most out of the data. Some features of physical systems don't change when the system is rotated or moved. By making neural network architectures that follow these symmetries and invariances, they can learn from fewer training examples and work better with new ones. To prove this claim, I used HEA 4 input data and divided them in three different input set. The total frames are 8075 from 35 DFT calculation files. The next input sets were chosen by randomly picking 17 DFT calculation files from the 35 data and the total frames they gave around 3000. I created three different training sets from these data where one is with all data, number of training sets 6000 and 2000, the last one with smaller input data which only has 2000

training sets. The results for training force RMSE is shown in Figure 19. Besides the number of training sets per epoch, every other hyperparameter was kept the same. It was observed that each of them performed and showed results at the same level. At 600 steps, the RMSE error with largest training set with 6000 training set is 0.0947 eV/atoms and with the smallest is 0.101 eV/atoms. The dilemma here is that using large data we get same results like the small data gave us. It's conceivable that using a larger dataset didn't yield different results because the dataset was already large enough to capture the necessary information for the task at hand. In some cases, a certain quantity of data is sufficient to accurately model the system being studied, and adding more data beyond that point may not provide additional benefit. Additionally, there could be other factors that influenced the results beyond the magnitude of the dataset. For example, the quality and diversity of the data could be more essential than the quantity of the data. It's also conceivable that the hyperparameters of the model or the training process were not well-tuned, which can lead to similar results regardless of the dataset size.

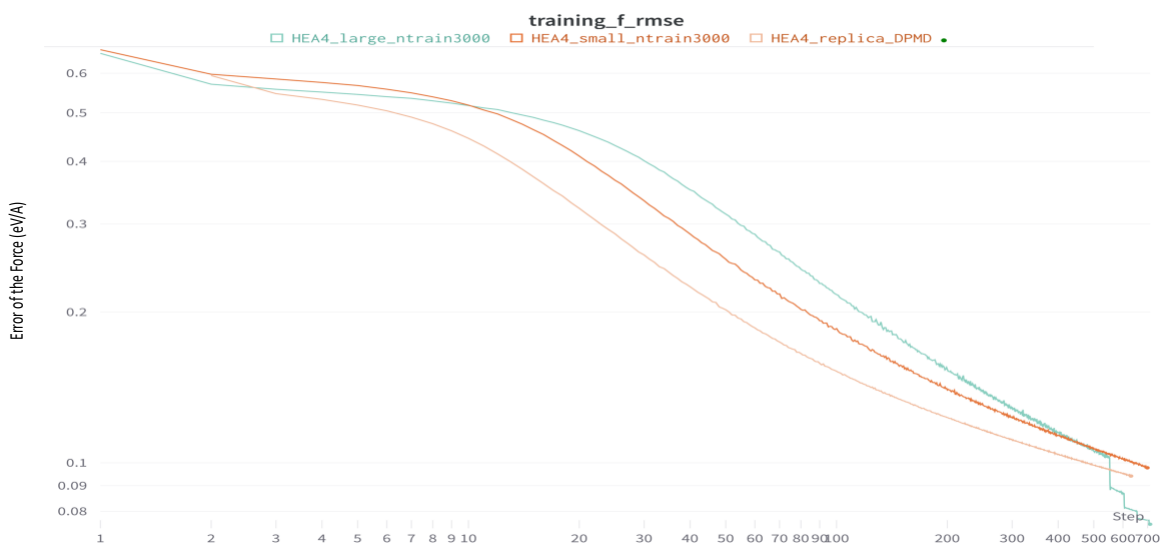


Figure 19. RMSE plot of different training sets of HEA 4 force RMSE

Hyperparameter Optimization

As I mentioned earlier that importance of hyperparameters in training MLIP, I will mention here the hyperparameters value that we used in Allegro. We used cut off radius from 4 to 6 Å and we get good results shown in Table 7 when we use 4~4.5Å for our HEA compositions. Learning rate was tweaked between 0.001 to 0.005, batch size 1. The Weights and Biases (wandb) (36) platform has a technique called "parallel coordinates" that lets users plot and explore high-dimensional data. In this way of showing data, each feature or variable is shown by a vertical axis, and all of the axes run in the same direction. Then, each data point is shown as a line that connects the values of each feature along the axis that goes with it(Figure 20).

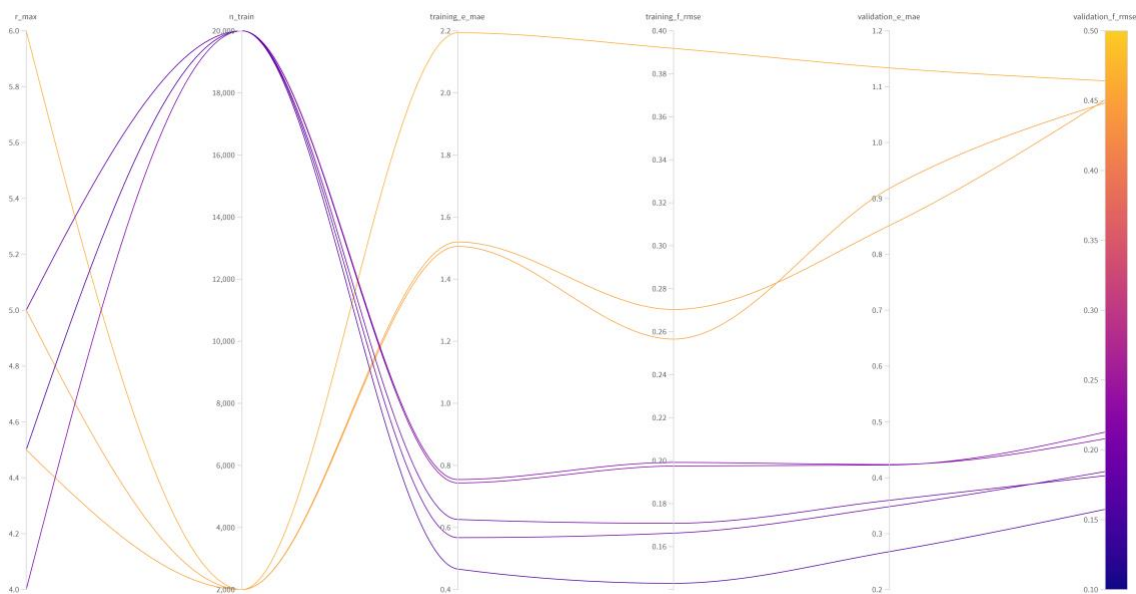


Figure 20. Parallel coordinates of HEA 4 system

The main benefit of parallel coordinates is that it makes it easy to find patterns and relationships between variables in data with a lot of dimensions. By looking at the lines, users can easily see which variables are strongly related to each other and how changes in one variable

affect other variables. This is especially helpful in machine learning, where models often work on high-dimensional data with many variables. Parallel coordinates in WandB also have interactive features, like filtering and highlighting, that let users explore and compare subsets of the data based on certain criteria. For example, users can filter the data to only show lines where a certain variable falls within a certain range or highlight lines where a certain variable has an extreme value. This makes it easier to find trends and patterns in the data that might not be obvious with other ways of displaying it. Another benefit of using parallel coordinates in WandB is that it works well with other features of the platform, such as the ability to track model performance and hyperparameters. This makes it easy for users to see how changes to model parameters or input data affect the way variables relate to each other.

Table 7. Hyperparameters and RMSE

| Hyperparameters | Energy RMSE (eV) | Force RMSE (eV/atoms) |
|---|------------------|-----------------------|
| Cut off radius 4 ~ 5 with training set 20,000 | 0.45-0.77 | 0.150 - 0.195 |
| Cut off radius 4 ~ 5 with training set 2,000 | 1.50 - 2.22 | 0.259 – 0.390 |

For most of Allegro potentials, we got lower for RMSE when we used higher number of training data per batch. When a model has more training data, it can perform better and have a lower RMSE because it has more examples to learn from and apply to new data. In other words, when there is more training data, the model can pick up on a wider range of patterns and changes in the data, making it more accurate and stable. When a deep learning model is trained on a small amount of data, it might not be able to find all of the patterns and differences in the data. This can lead to overfitting, where the model becomes too specific to the training data and doesn't work well with new data it hasn't seen before. But if you give the model more training data, it

can see a wider range of examples and learn more general patterns. Also, having more training data can help make outliers and noise in the data less noticeable. When a model is trained on a small amount of data, outliers and noise can have a big effect on how well it works. With more training data, the model can tell the difference between real patterns and random noise in the data better. Also using wandb visualization tool, we can also inspect each species of our system.

Figure 21 shows which species error of force with training steps for HEA4 and we can tell from this plot that Al has lowest RMSE. We can add more training data or improve the quality of the data of other elements to get a good potential.

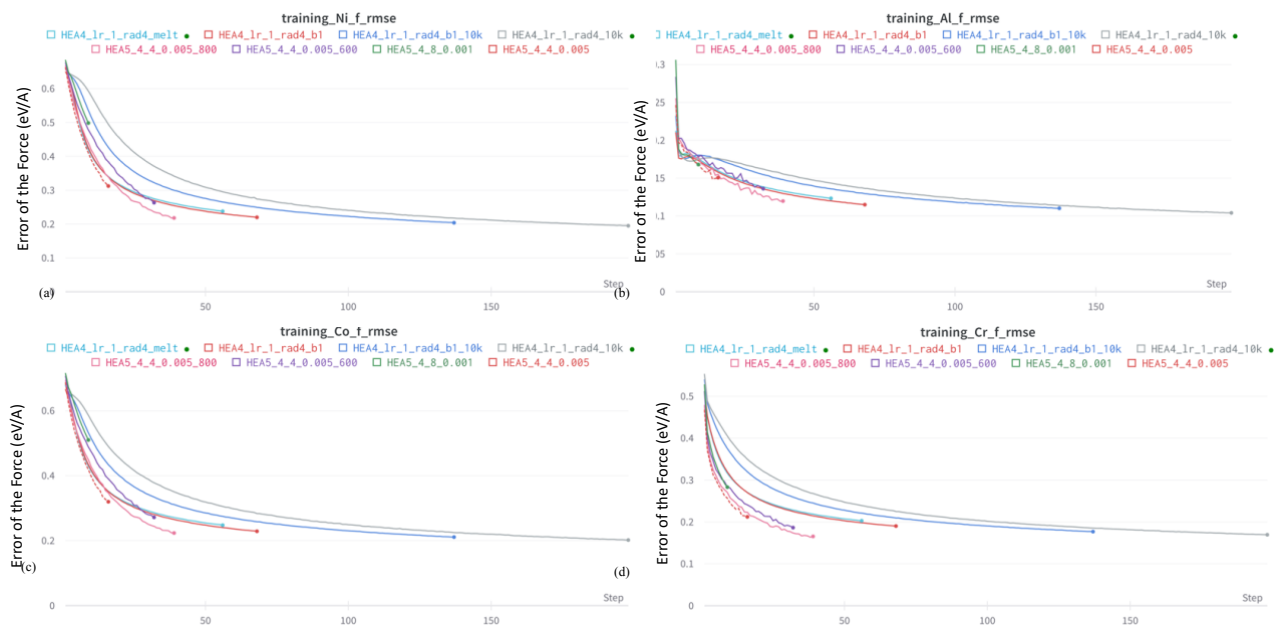


Figure 21. Error of force for individual species of HEA 4 (a) Ni (b) Al (c) Co (d) Cr

MTP Model Accuracy

We tried 4 and 5 elements HEA in MTP. With different training approach and potential type, we got average RMSE training values for HEA4 from energy per atom, force, and virial per atom 0.019 eV, 0.51 eV/Å, and 0.07 eV respectively. For HEA5, the values are lower than HEA4 for

energy and they are 0.0089 eV, 0.44 eV/Å and 0.089eV respectively. In these training, we used functional form of MTP of level 10. I also tried for HEA 5 with level 02, the energy rmse was 0.015eV and forces 0.62 eV/Å, level 06 was which is higher rmse than level 10.

MTP Model Verification Using MD Simulations

I am going to discuss the results from HEA 5 NPT test. In Figure 22, the lattice constant of HEA4(Ni, Al, Co, Cr) using MTP and DeePMD 5 component potential for 300-600K is shown. The lattice constants fall into the range of 3.55 to 3.59 Å which is closer to the reference (37). The MLIPs are generously capturing the temperature-dependent behavior of the materials. I also got CIJ of Ni using this MTP potentials. The values I got for C_{11} , C_{22} , C_{44} , η , K are 209.51 GPa, 122.567 GPa, 147.133 GPa, 0.37, 151.608 GPa respectively which is not so close for some elements to the values described in Table 2 but closer to the reference (38) .

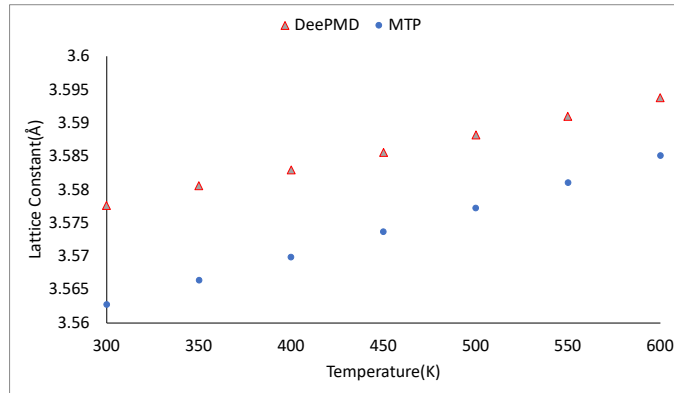


Figure 22. Lattice constant of HEA4 using DeePMD and MTP Potentials

Similar to Allegro, I was able to do an NVT test using MTP potentials of 5 elements, and the findings I'm going to provide are primarily stability between 300K and 1000K for various structures and phases. Figure 23 depicts one phase under two different temperatures of 300K and

1000K. The figure depicts their stable condition at various temperatures using a radial distribution function (rdf).

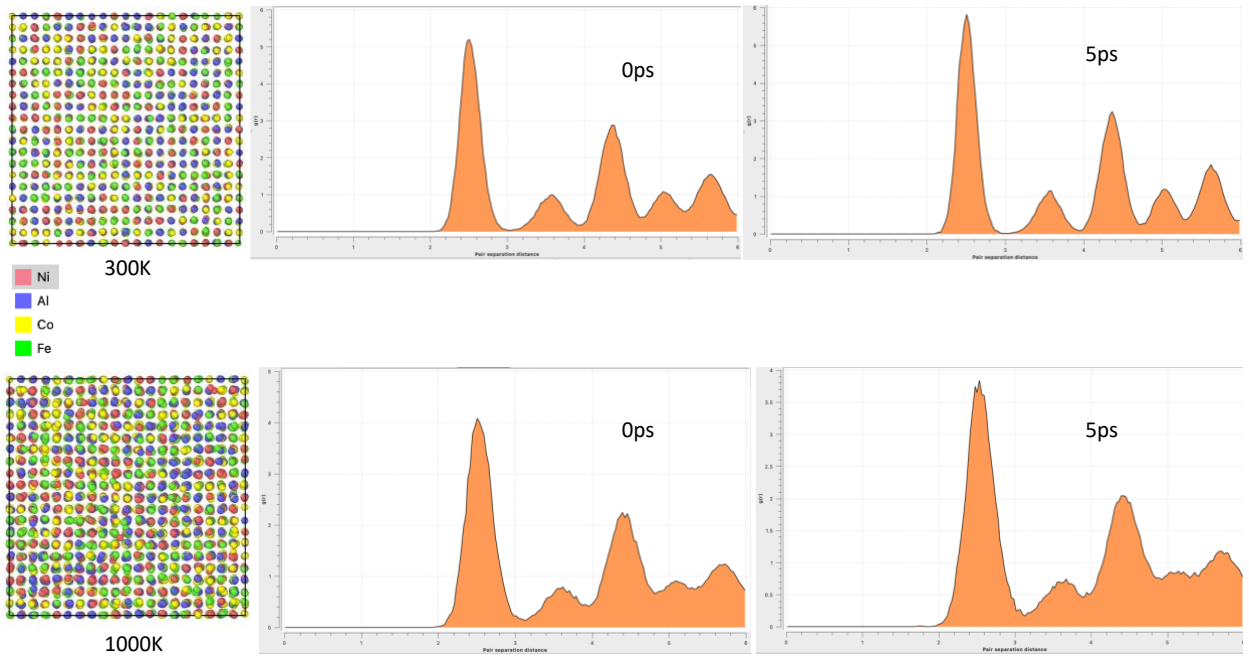


Figure 23. Thermal stability of supercell models of Gamma phase of HEA 4 at different temperature using HEA 5 MTP Potential

Since the RDF indicates stability and melting behavior for any cluster size, it can be used to monitor melting. Coordination analysis shows the first peak, and when these abrupt peaks are neutralized, the RDF plot becomes repeated, like a normal fluid function. If $g(r)$ remains constant with pair distance for 5000 frames at a certain temperature, the system is in equilibrium and has a stable structure that persists over time. Hence, the particles are organized according to the system's conditions (temperature, pressure, etc.) and do not vary over time. The system's density, particle interactions, and other factors determine $g(r)$ at this temperature and system. The

system's particle arrangement or structure does not change during 5000 frames since $g(r)$ remains constant.

MTP Active Learning and Future Work

The process of calculating the melting point of aluminum through active learning with an MTP is outlined below. Aluminum is chosen as the benchmark system due to the availability of accurate DFT results. The tuning of the parameters of the potential is accomplished actively during a molecular dynamics' simulation, and the training configurations are updated as needed by quantum mechanical calculations. During the MD simulation, the machine learning algorithm employs a "query strategy"—an algorithm that geometrically compares a given configuration with the training configurations—to assess whether the computed atomic configuration is sufficiently novel to be added to the training set. If it must be added, the program retrieves the quantum mechanical (QM) computations for this new configuration and modifies the MTP parameters in accordance with the derived atomic energies, forces, and stresses. In fact, the learning on the fly algorithm connects the LAMMPS-implemented molecular dynamics driver to a quantum mechanical model (VASP). Learning on the fly will continue until the MTP no longer detects new configurations during the MD. The training efficiency and accuracy of the resulting MTP are dictated by the extrapolation threshold, which is, in essence, the tolerance above which a configuration is deemed sufficiently novel. The active learning iterations are implemented in MLIP, following the steps described in Figure 24. The iteration begins by generating the state.als file based on the training set. Then, LAMMPS is used to run an MD simulation with active selection of configurations, and the extrapolative configurations are written to preselected.cfg. The select-add command is used to select representative configurations with maximal determinant, and

DFT calculations are performed on these configurations and written to computed.cfg. The computed.cfg is then appended to the previous training set, and the potential is re-fit on the expanded training set. The output of the iteration is an updated potential that can predict in a larger configurational space.

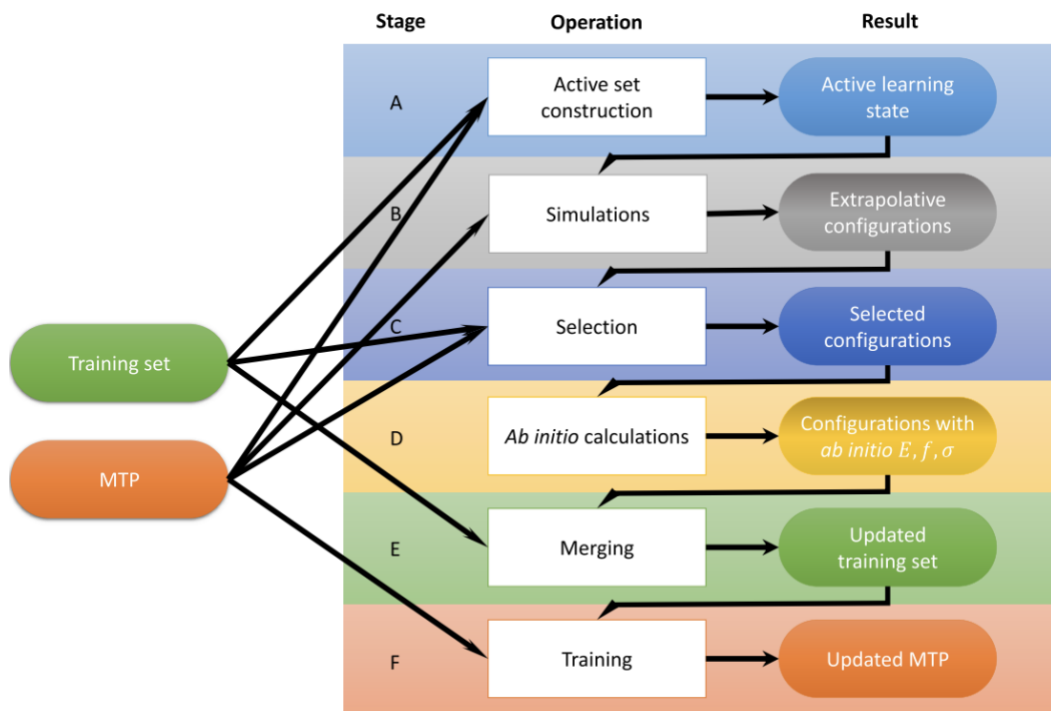


Figure 24. Scheme of active learning iterations for training MTP

Using active learning, the purpose of this study is to develop a valid DFT-trained potential for sampling Al solid and liquid configurations. In step 1, an initial training set is produced by conducting a 90-fs VASP MD trajectory, and a 108-atom FCC structure is used to execute an optimal NVT-MD to train the potential. Active learning iterations involve molecular dynamics and configuration collection, with the potential extrapolating in accordance with the procedure shown in figure 24. The study then advances to stage 1b, where active learning is

implemented via multi-scenario MD simulation. Parallel trajectories are done with lattice parameter a between 4.0 and 4.25 and temperatures, T between 700 and 800 K, and in the liquid scenario, 10,000 steps are executed with $a = 4.25$ and $T = 1500$ K, with the target temperature gradually decreasing a to the required value. To evaluate iterations of active learning, a validation set of 180 configurations is generated. After boosting data precision, this level's 186 liquid and solid variations are computed using high-precision DFT. Five MTPs with random initialization of parameters are fitted to this training data, and the MTP with the lowest error is selected. Active learning iterations recommence using the same 180-configuration validation set, but with new DFT parameters for error evaluation. Active training reduces the validation error of the first 10 configurations from 13.5% to 12%, as shown by the results. On the 75th iteration, the trained potential samples a total of 186 solid and liquid options. The conclusion of the paper discusses the possibility for future research into whether MD probability distributions influence the accuracy of the potential and the efficacy of noise averaging as a method for increasing potential accuracy.

This study emphasizes the possibility of active learning to generate accurate potentials for modelling atomic-level materials. The authors demonstrate that by combining simulations of molecular dynamics and configuration collection with DFT training, it is possible to develop a highly accurate potential that can consistently sample solid and liquid Al configurations. Further study in this area could aid in enhancing the precision of these potentials and enable simulations of materials and their properties that are more precise.

CONCLUSION

I used three different machine learning approaches and developed Deep Learning interatomic potentials to model a multi-phase and multi-components system of Ni-based Superalloys. I have found that these machines learning interatomic potentials can predict DFT input data very well. They show excellent similarities in predicting mechanical and thermal properties using MD simulations. But there is still some room for improvement. I have found that we should focus carefully on what type of data we are feeding to these algorithms. The overall strategy for building interatomic potential by first fitting energy and force data at relatively low temperature and then adding high temperature data to match the potential is effective. It is noticeable that some of my ML potentials which shows great validation by predicting input data well, they cannot directly translate to recreate stable MD trajectories. So, to simulate a complex system of 10 component systems, we will need to employ iterative or on the fly learning strategy to refine the original potential. For future project, we should focus on active learning using DP-GEN (39), Psiflow (40) which are promising codes to create more robust potentials for multicomponent systems. Simulating larger domains with these accurate models will serve to strengthen our predictions to materials of interest, given the promise of exascale computing power.

REFERENCES

1. Murakumo, T., Koizumi, Y., Kobayashi, K. and Harada, H. (2004). Creep strength of Ni-base single-crystal superalloys on the γ/γ' tie-line. *Superalloys*, 155-162.
2. Gao, M. C., Yeh, J. W., Liaw, P. K. and Zhang, Y. (2016). *High-entropy alloys*. Springer International Publishing.
3. Yeh, J. W., Chen, S. K., Lin, S. J., Gan, J. Y., Chin, T. S., Shun, T. T., C.-H. Tsau and Chang, S. Y. (2004). Nanostructured high-entropy alloys with multiple principal elements: novel alloy design concepts and outcomes. *Adv. Eng. Mater.*, **6**, 299-303.
4. George, E. P., Raabe, D. and Ritchie, R. O. (2019). High-entropy alloys. *Nat. Rev. Mater.*, **4**, 515-534.
5. Steinhauser, M. O. and Hiermaier, S. (2009). A review of computational methods in materials science: examples from shock-wave and polymer physics. *Int. J. Mol. Sci.*, **10**, 5135-5216.
6. Müser, M. H., Sukhomlinov, S. V. and Pastewka, L. (2023). Interatomic potentials: Achievements and challenges. *Adv. Phys. X*, **8**, 2093129.
7. Peverati, R. and Truhlar, D. G. (2014). Quest for a universal density functional: the accuracy of density functionals across a broad spectrum of databases in chemistry and physics. *Philos. Trans. A. Math. Phys. Eng. Sci.*, **372**, 20120476.
8. Orio, M., Pantazis, D. A. and Neese, F. (2009). Density functional theory. *Photosynth. Res.*, **102**, 443-453.
9. Hafner, J. and Kresse, G. (1997). Ab initio molecular dynamics for liquid metals. *Phys. Rev. B*, **47**, 558.
10. Brownlee, J. (2016). *Machine learning algorithms from scratch with Python*. Machine Learning Mastery.
11. Behler, J. and Parrinello, M. (2007). Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.*, **98**, 146401.
12. Zhang, L., Han, J., Wang, H., Saidi, W. and Car, R. (2018). End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems. *Adv. Neural Inf. Process Syst.*, **31**, 4436-4446.
13. Zhang, L., Han, J., Wang, H., Car, R. and Weinan, E. (2018). Deep potential molecular dynamics: a scalable model with the accuracy of quantum mechanics. *Phys. Rev. Lett.*, **120**, 143001.

14. Lu, D., Wang, H., Chen, M., Lin, L., Car, R., Weinan, E., Jia, W. and Zhang, L. (2021). 86 PFLOPS Deep Potential Molecular Dynamics simulation of 100 million atoms with ab initio accuracy. *Comput. Phys. Commun.*, **259**, 107624.
15. Dai, F. Z., Wen, B., Sun, Y., Xiang, H. and Zhou, Y. (2020). Theoretical prediction on thermal and mechanical properties of high entropy ($Zr_{0.2}Hf_{0.2}Ti_{0.2}Nb_{0.2}Ta_{0.2}$) C by deep learning potential. *J. Mater. Sci. Technol.*, **43**, 168-174.
16. Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J. P., Kornbluth, M. and Kozinsky, B. (2022). E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat. Commun.*, **13**, 2453.
17. Musaelian, A., Batzner, S., Johansson, A., Sun, L., Owen, C. J., Kornbluth, M. and Kozinsky, B. (2023). Learning local equivariant representations for large-scale atomistic dynamics. *Nat. Commun.*, **14**, 579.
18. Thompson, A. P., Aktulga, H. M., Berger, R., Bolintineanu, D. S., Brown, W. M., Crozier, P. S. and Plimpton, S. J. (2022). LAMMPS—a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Comput. Phys. Commun.*, **271**, 108171.
19. Novikov, I. S., Gubaev, K., Podryabinkin, E. V. and Shapeev, A. V. (2020). The MLIP package: moment tensor potentials with MPI and active learning. *Mach. Learn. Sci. Technol.*, **2**, 025002.
20. Novoselov, I. I., Yanilkin, A. V., Shapeev, A. V. and Podryabinkin, E. V. (2019). Moment tensor potentials as a promising tool to study diffusion processes. *Comp. Mater. Sci.*, **164**, 46-56.
21. Sepp Löfgren, N. (2021). *Accelerating bulk material property prediction using machine learning potentials for molecular dynamics: predicting physical properties of bulk Aluminium and Silicon*. [Master's thesis, Linköping University]. <http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-179894>
22. Cohn, D., Atlas, L. and Ladner, R. (1994). Improving generalization with active learning. *Mach. Learn.*, **15**, 201-221.
23. Bock, F. (2021). *Active learning of interatomic potentials to investigate thermodynamic and elastic properties of $Ti_{0.5}Al_{0.5}N$ at elevated temperature*. [Diploma Project, Linköping University]. <http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-176587>
24. Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. Springer.
25. McGilvry-James, T., Timalisina, B., Mou, M. M. and Sakidja, R. (2022). Deep potential development of transition-metal-rich carbides. *MRS Adv.*, **7**, 468-473.

26. Liang, W., Lu, G. and Yu, J. (2020). Molecular Dynamics Simulations of Molten Magnesium Chloride Using Machine-Learning-Based Deep Potential. *Adv. Theory Simul.*, **3**, 2000180.
27. Gholami, M. D., Hashemi, R. and Sedighi, M. (2020). The effect of temperature on the mechanical properties and forming limit diagram of aluminum strips fabricated by accumulative roll bonding process. *J. Mater. Res. Technol.*, **9**, 1831-1846.
28. Adhikari, P., San, S., Zhou, C., Sakidja, R. and Ching, W. Y. (2019). Electronic Structure and Mechanical Properties of Crystalline Precipitate Phases $M_{23}C_6$ (M= Cr, W, Mo, Fe) in Ni-Based Superalloys. *Mater. Res. Express*, **6**, 116323
29. Liu, Y., Jiang, Y., Xing, J., Zhou, R. and Feng, J. (2015). Mechanical properties and electronic structures of $M_{23}C_6$ (M= Fe, Cr, Mn)-type multicomponent carbides. *J. Alloys Compd.*, **648**, 874-880.
30. De Jong, M., Chen, W., Angsten, T., Jain, A., Notestine, R., Gamst, A., Sluiter, M., Krishna Ande, C. Asta, M. *et al.* (2015). Charting the complete elastic properties of inorganic crystalline compounds. *Sci. Data*, **2**, 1-13.
31. Boccato, S., Torchio, R., Kantor, I., Morard, G., Anzellini, S., Giampaoli, R., Briggs, R., Smareglia, A., Irfune, T. and Pascarelli, S. (2017). The melting curve of nickel up to 100 GPa explored by XAS. *J. Geophys. Res. Solid Earth*, **122**, 9921-9930.
32. Zou, Y. and Chen, L. R. (2005). Pressure dependence of the melting temperature of aluminum. *Phys. Status Solidi B*, **242**, 2412-2416.
33. Haupt, T. (2014). *LAMMPS dislocation mobility*. LAMMPS Dislocation Mobility - EVOCD. Retrieved April 27, 2023, from https://icme.hpc.msstate.edu/mediawiki/index.php/LAMMPS_Dislocation_Mobility.html
34. Stukowski, A. (2009). Visualization and analysis of atomistic simulation data with OVITO—the Open Visualization Tool. *Model. Simul. Mat. Sci. Eng.*, **18**, 015012.
35. Stukowski, A. (2009). *Coordination analysis*. Coordination analysis - OVITO User Manual 3.8.3 documentation. Retrieved April 27, 2023, from https://www.ovito.org/docs/current/reference/pipelines/modifiers/coordination_analysis.html
36. Biewald, L. (2020). *Weights & Biases – developer tools for ML*. Weights & Biases – Developer tools for ML. Retrieved April 27, 2023, from <https://wandb.ai/site>
37. Wang, W. R., Wang, W. L., Wang, S. C., Tsai, Y. C., Lai, C. H. and Yeh, J. W. (2012). Effects of Al addition on the microstructure and mechanical property of $Al_xCoCrFeNi$ high-entropy alloys. *Intermetallics*, **26**, 44-51.

38. Rassoulinejad-Mousavi, S. M., Mao, Y. and Zhang, Y. (2016). Evaluation of copper, aluminum, and nickel interatomic potentials on predicting the elastic properties. *J. Appl. Phys.*, **119**, 244304.
39. Zhang, Y., Wang, H., Chen, W., Zeng, J., Zhang, L., Wang, H. and Weinan, E. (2020). DP-GEN: A concurrent learning platform for the generation of reliable deep learning based potential energy models. *Comput. Phys. Commun.*, **253**, 107206.
40. Vandenhoute, S., Cools-Ceuppens, M., DeKeyser, S., Verstraelen, T. and Van Speybroeck, V. (2023). Machine learning potentials for metal-organic frameworks using an incremental learning approach. *NPJ Comput. Mater.*, **9**, 19.