12-1-2020

Developments of Machine Learning Potentials for Atomistic Simulations

Howard Yanxon

DEVELOPMENTS OF MACHINE LEARNING POTENTIALS FOR ATOMISTIC SIMULATIONS

By

Howard Yanxon

Bachelor of Science – Physics
University of Nevada, Las Vegas
2015

A dissertation submitted in partial fulfillment
of the requirements for the

Doctor of Philosophy – Physics

Department of Physics and Astronomy
College of Sciences
The Graduate College

University of Nevada, Las Vegas
December 2020

UNLV | GRADUATE COLLEGE

Dissertation Approval

The Graduate College
The University of Nevada, Las Vegas

November 23, 2020

This dissertation prepared by

Howard Yanxon

entitled

Developments of Machine Learning Potentials for Atomistic Simulations

is approved in partial fulfillment of the requirements for the degree of

Doctor of Philosophy – Physics
Department of Physics and Astronomy

Qiang Zhu, Ph.D.
Examination Committee Chair

Kathryn Hausbeck Korgan, Ph.D.
Graduate College Dean

Bernard Zygelman, Ph.D.
Examination Committee Member

Aidan Thompson, Ph.D.
Examination Committee Member

Ashkan Salamat, Ph.D.
Examination Committee Member

Yifei Mo, Ph.D.
Examination Committee Member

Monika Neda, Ph.D.
Graduate College Faculty Representative

Abstract

Atomistic modeling methods such as molecular dynamics play important roles in investigating time-dependent physical and chemical processes in the microscopic level. In the simulations, energy and forces, sometimes including stress tensor, need to be recalculated iteratively as the atomic configuration evolves. Consequently, atomistic simulations crucially depend on the accuracy of the underlying potential energy surface. Modern quantum mechanical modeling based on density functional theory can consistently generate accurate description of the potential energy surface. In most of the cases, molecular dynamics simulations based on density functional theory suffer from the highly demanding computational costs. On the other hand, atomistic simulations based on classical force fields have proven to be essential in the computational modelling community due to their unrivaled computational efficiency. However, classical force fields are only useful for inspecting the qualitative insights because they fail to provide confidence in the quantitative results for a lot of cases. In this thesis, I will show the power of machine learning potentials, resolving the predicaments described above. First, the machine learning potential methods will be applied to SiO_2 for investigating the implications of different machine learning potentials. Second, the machine learning potentials will be extended to predict physical properties of crystalline silicon, Ni-Mo system, high entropy alloy of NbMoTaW, and Pt for nanoparticle systems. Finally, the diffusion barriers of Pt adsorptions on Pt(111) and Pt(100) surfaces will be examined in detail.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

Recently, the availability of extensive dataset with the improvement in algorithms, along with the exponential growth in computing power, has guided unparalleled surge of interest in machine learning research. A major portion of our daily life from fraud detection [1], email/spam filtering [2], credit scores [3], image/speech recognition [4, 5], to recommendation system [6] is powered by machine learning algorithms. Many more exciting applications such as self-driving cars [7, 8] are being integrated in our daily life at the time of writing. Unsurprisingly, machine learning methods in solving materials science problems have become hot topic in the community [9].

Traditionally, experiments had dominated discoveries of new materials in the past. Many important discoveries happened mostly through human intuition or serendipity [10]. On the other hand, the first computational revolution in materials science was stimulated by the emergence of tractable computational method such as the state-of-the-art density functional theory (DFT) [11, 12], Monte Carlo simulations, and molecular dynamics (MD) to study large variety of problems, ranging from exploration phase space [13] to protein foldings [14]. In the information age, tremendous efforts have been dedicated to empowering the rise of the second computational revolution inspired by the Materials Genome Initiative (MGI) [15].

The aim of MGI is to accelerate materials development via creation of publicly-available large sets of comprehensive data, which can bridge the gap between experiment and theory. Hence, the high-throughput (HT) computational materials design has become popular in materials science recently [16, 17]. HT computational materials researches are conducted based on the construction of databases that span over a large volume of crystal structures computed by the DFT method. There are several available databases online such as AFLOWLIB ($\sim$3,000,000 data with over 500 millions calculated properties) [18], Materials Project ($\sim$800,000 data) [19], OQMD ($\sim$800,000 data) [20], etc. These databases based on DFT method have built a highway for materials discovery. Despite of these advances, traditional computations based on the accurate DFT are still expensive

in decoding most of the materials science problems.

Despite the accuracy, DFT simulations can be extendable only to a few hundreds of atoms at a few picoseconds in MD simulations. In MD simulations, energy and forces need to be recalculated iteratively as the atomic configuration evolves. Consequently, MD simulations crucially depend on the accuracy of the underlying potential energy surface (PES). DFT method can accurately represent the PES. However, MD simulations based on DFT suffer from the highly demanding computational cost due to solving the Kohn-Sham equation. Solving the equation requires thousands of quantum-mechanical calculations that are scaled at $O(N^3)$ with respect to the number of atoms N. In consequence, simulating the structural evolution of many of complicated systems in DFT remains demanding in spite of the remarkable progress in computational facilities and efficient algorithms. This bottleneck in the DFT method is likely to persist in the foreseeable future.

On the other hand, the classical MD method can model large systems at long-time scale [21, 22, 23], countering the unwavering issue of the DFT method. A great amount of efforts has been dedicated in developing PES using the classical method [24, 25, 26, 27, 28]. The reconstruction of the PES is usually based on simple analytical functions related to the scalar properties of the system. The class force fields can be applied to comprehend the qualitative behavior of the system. However, they are often inadequate to describe the quantitative properties of the system.

Recently, machine learning methods have been widely applied to resolve the dilemma in comprising between accuracy and cost [29]. The machine learning potential (MLP) are trained by minimizing the cost function to attune the model to deliberately describe the *ab initio* data. The cost of atomistic simulation is orders of magnitude lower than the quantum mechanical simulation, allowing the system to be scaled up to 10^5–10^6 atoms [30, 31]. The power of MLP method is illustrated by many applications to a range of materials [32, 33, 34, 35]. A large amount of DFT data (structures, energy, forces, and stresses) are required to develop an accurate MLP. The structures must be represented by appropriate descriptors (high-dimensional real valued array) in order to identify the similarities and/or dissimilarities in the atomic environments. In MLP fitting, a variety of regression techniques are used to correlate between the descriptor and energy/forces.

Several machine learning techniques for developing MLP had been successfully implemented: linear/polynomial regression [36, 37, 38, 39], Gaussian process regression (GPR) [40, 41], and high-dimensional neural network potential (NNP) [42, 29].

1.1 Thesis Layout

The thesis will be presented as follow: In Chapter 2, the methods will be described in details. The method section includes the theories of quantum-mechanical computations, classical force field methods, and machine learning potential techniques. In Chapter 3, a publicly-available package, PyXtal_FF for constructing machine learning potentials will be introduced. The philosophy of PyXtal_FF and example usage will be shown for those practitioners. In Chapter 4, the applications of PyXtal_FF will be demonstrated with SiO_2, crystalline silicon, Ni-Mo alloys, NbMoTaW high entropy alloy, and Pt nanoclusters. In Chapter 5, the diffusion barriers of Pt adsorptions on Pt(111) and Pt(100) surfaces will be examined comprehensively.

Chapter 2 Theory

2.1 Potential Energy Surface

The potential energy surface (PES) is a surface described in a very high-dimensional configuration space, which represents all possible configurations. The surface consists of many local minima, which can be obtained through geometry optimization, via attractive interactions. Meanwhile, the transition states or saddle points are the connecting states amongst minima via steepest-descent pathway. The number of distinct points on the surface can be estimated as:

$$K = \binom{V/\delta^3}{N} \prod_i \binom{N_{atom}}{n_i} \tag{2.1}$$

where n_i is i-th species in the unit cell, N_{atom} refers to the number of atoms enclosed in the volume V. δ represents the discretization parameter. Typically, δ has the value of 1. Moreover, the dimensionality of the PES scales as the number of geometric degrees of freedom of the molecule:

$$D = 3N + 6 \tag{2.2}$$

where $3N$ degrees of freedom for the 3-dimensional space for each atoms, and lattice defines the remaining six dimensions. Any atomistic simulation, including crystal structure search, requires a description of the interacting atoms—the energy and forces, sometimes stress, that governs the atomic motion and formation—that can be described by the PES. Hence, constructing the true PES can be a NP-hard problem.

2.2 Quantum Theory

The most accurate determination of the energy of a system is to be calculated quantum-mechanically. Calculation of energy of a system is crucial in many applications such as geometry optimization,

atomistic simulations, etc. In physics, a calculation is said to be from first principles or *ab initio*, when it begins directly at the fundamental laws of physics. By solving the non-relativistic Schrödinger's equation, the energy (E) and wavefunction (Ψ) of a many-body electron-nuclear system can be obtained:

$$\hat{H}\Psi = E\Psi \tag{2.3}$$

where the full Hamiltonian operator, $\hat{H}$, of the many-body system is:

$$\begin{aligned}\hat{H} = &-\sum_i \frac{\hbar^2}{2m_e}\bigtriangledown_i^2 - \sum_{i,I}\frac{Z_I e^2}{|\boldsymbol{r}_i - \boldsymbol{R}_I|} + \frac{1}{2}\sum_{i\neq j}\frac{e^2}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} \\ &-\sum_I \frac{\hbar^2}{2M_I}\bigtriangledown_I^2 + \frac{1}{2}\sum_{I\neq J}\frac{Z_I Z_J e^2}{|\boldsymbol{R}_I - \boldsymbol{R}_J|}\end{aligned} \tag{2.4}$$

In Eq. 2.4, the top row belongs to the electronic contribution, while the bottom row describes the kinetic energy of the nuclei and nuclear-nuclear Coulomb interaction, respectively. m_e and M_I are the mass of electron and nuclei, respectively. The charge of nucleus is denoted as Z_I. The first term is the kinetic energy of the electron ($\hat{T}_e$); the second term defines the electron-nuclear interaction ($\hat{V}_{ext}$), which corresponds to the the potential created by the nucleus for electron to live in. The third term is the Coulomb interaction amongst electrons ($\hat{V}_{int}$). Solving the Schrödinger's equation is a daunting task as the wavefunction lives in the $3N_{ele}$ dimensional space ($R^{3N_{ele}}$), where N_{ele} is the number of electrons. Therefore, several approximations need to be introduced throughout this section.

The first approximation is the Born-Oppenheimer Approximation. The approximation assumes the electrons instantaneously go to the ground state as an atom is displaced. This is due to the heavier nuclei, which are about 2000 times heavier, have slower motions than the lighter electrons. This allows the fourth and fifth terms in Eq. 2.4 to be neglected as the computation of Schrödinger's equation becomes:

$$(\hat{T}_e + \hat{V}_{ext} + \hat{V}_{int})\Psi = E\Psi \tag{2.5}$$

Due to the wavefunction is a function of $3N_{ele}$ degrees of freedom, this poses a critical problem in computational bottleneck. Even for a small system such as Si dimer, solving the equation is computationally impossible.

2.2.1 Hartree Method

Hartree method is the starting point of the Hartree-Fock and Kohn-Sham methods, which are the methods used in practical computations nowadays. In Hartree model, it is assumed that the wavefunction is written in the product of the individual electronic orbital wavefunctions:

$$\Psi(\boldsymbol{r}_i) = \phi_1(\boldsymbol{r}_1)\phi_2(\boldsymbol{r}_2)\dots\phi_n(\boldsymbol{r}_n) \tag{2.6}$$

Hence, the Schrödinger's equation is transformed into a set of electronic orbital terms:

$$\left[-\frac{\hbar^2}{2m_e} \bigtriangledown_i^2 - \sum_I \frac{Z_I e^2}{|\boldsymbol{r} - \boldsymbol{R}_I|} + e^2 \int d\boldsymbol{r}' \frac{|\Psi(\boldsymbol{r}')|^2}{|\boldsymbol{r} - \boldsymbol{r}'|} \right] \phi_i(\boldsymbol{r}) = \epsilon_i \phi_i(\boldsymbol{r}) \tag{2.7}$$

Notice, that the first and second terms are consistent with Eq. 2.5, while the third term ends up in the interaction between the i-th orbital and all other electrons in the system in a mean-field approximation. This means that the other electrons are the average of their probabilities in time. This term is also sometimes called the Hartree potential. In the computational point of view, there are N_{ele} 3-dimensional functions to be solved in Eq. 2.7, whereas there is one $3N_{ele}$ dimension in Eq. 2.5. This is a huge improvement.

Nevertheless, the critical problem posed in the Hartree model is that the wavefunction (Ψ) suggests that there is no correlation between the orbital wavefunctions (ϕ). This is inconsistent with Pauli exclusion principle, i.e. the electrons should be indistinguishable. If the electrons are indistinguishable, swapping two electrons should change the wavefunction's sign.

2.2.2 Hartree-Fock Method

The Hartree model was improved by Fock by taking account of the antisymmetric behavior of the wavefunction (i.e. Pauli Exclusion principle). The Hartree-Fock equation is expressed as:

$$\hat{F}\phi_i(\boldsymbol{r}) = \epsilon_i\phi_i(\boldsymbol{r}) \tag{2.8}$$

To compensate for the nature of fermionic particles, the Hartree-Fock wavefunction of an N-electron system is written in a single Slater determinant composed of electronic orbital wavefunctions:

$$\Psi(\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \phi_1(\boldsymbol{r}_1, s_1) & \phi_2(\boldsymbol{r}_1, s_1) & \dots & \phi_N(\boldsymbol{r}_1, s_1) \\ \phi_1(\boldsymbol{r}_2, s_2) & \phi_2(\boldsymbol{r}_2, s_2) & \dots & \phi_N(\boldsymbol{r}_2, s_2) \\ \vdots & \vdots & & \vdots \\ \phi_1(\boldsymbol{r}_N, s_N) & \phi_2(\boldsymbol{r}_N, s_N) & \dots & \phi_N(\boldsymbol{r}_N, s_N) \end{vmatrix} \tag{2.9}$$

where $\phi_i(\boldsymbol{r}_i, s_i)$ is spin-orbital wavefunction, in which s_i denotes the spin. Swapping particles (i.e. swapping the rows of the Slater determinant) changes the sign of the wavefunction. This is in agreement with the nature of the Pauli Exclusion principle. Therefore, the explicit form of the Schrödinger's equation can be written explicitly as,

$$\left[-\frac{\hbar^2}{2m_e}\nabla_i^2 - \sum_I \frac{Z_I e^2}{|\boldsymbol{r} - \boldsymbol{R}_I|} + e^2 \int d\boldsymbol{r}' \frac{|\Psi(\boldsymbol{r}')|^2}{|\boldsymbol{r} - \boldsymbol{r}'|}\right]\phi_i(\boldsymbol{r}) - \frac{e^2}{2}\sum_{i\neq j}\int d\boldsymbol{r}' \frac{\phi_j^*(\boldsymbol{r}')\phi_i(\boldsymbol{r}')}{|\boldsymbol{r} - \boldsymbol{r}'|}\phi_j(\boldsymbol{r}) = \epsilon_i\phi_i(\boldsymbol{r}) \tag{2.10}$$

The first three terms are similar to the Hartree method, while the forth term is called exchanged term, which appears as the result of the antisymmetric nature of the electrons. The term corrects the probability of two electrons with parallel spin occupying the same orbital. Hartree-Fock method was the first theory that successfully allowed quantitative predictions of atoms and molecules. Nevertheless, the exchange term is computationally expensive because the wavefunction is determined as the result of the integration over the whole space. Finally, Hartree-Fock approach neglects the

correlation effect (i.e. the Coulomb repulsion) between electrons.

2.2.3 Density Functional Theory

2.2.3 Hohenberg-Kohn Method

Density functional theory (DFT) is the workforce of electronic structure calculations to date. It formulates the quantum theory into the manageable 3D electron density instead of the $3N$ dimensional wavefunction. The foundation of DFT is deduced from the Hohenberg and Kohn theorems [11, 43]:

Theorem 1. The ground state electron density ($n_0(\boldsymbol{r})$) uniquely determines the external potential (V_{ext}). Consequently, the electron density determines the wavefunction and the total energy.

Theorem 2. The electron density that minimizes the energy of the overall functional is the true ground state of electron density, i.e. the exact $n_0(\boldsymbol{r})$ gives the ground state energy of the system.

The electron density $n(\boldsymbol{r})$ is expressed by

$$n(\boldsymbol{r}) = \langle \Psi | \hat{n} | \Psi \rangle \tag{2.11}$$

where,

$$\hat{n} = \sum_{i=1}^{N} \delta(\boldsymbol{r} - \boldsymbol{r}_i)\delta_{\sigma,\sigma_i} \tag{2.12}$$

Hence, the total energy of the system according to the Hohenberg and Kohn theorems is

$$E[n] = F[n(\boldsymbol{r})] + \int n(\boldsymbol{r}) V_{ext}(\boldsymbol{r}) d\boldsymbol{r} \tag{2.13}$$

Since the electron-ion (external) potential can be exactly determined by the electron density (i.e. *Theorem 1*), the challenging part now is to determine the kinetic energy and the electron-electron interactions.

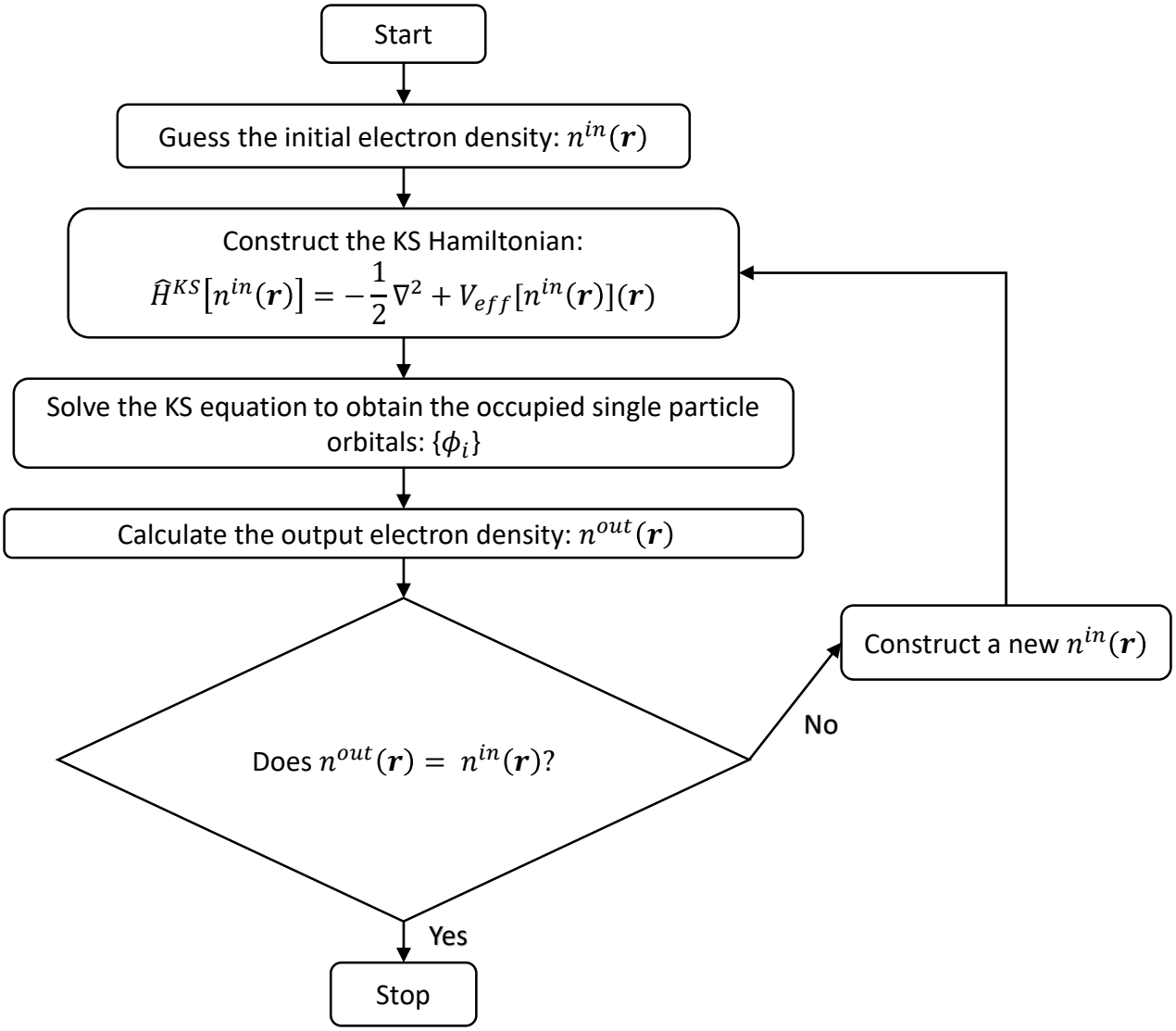


Figure 2.1: Representation of self-consistent field loop for solving Kohn-Sham equation. The V_{eff} includes the electron-ion potential, Hartree potential, and exchange-correlation term.

2.2.3 Kohn-Sham Method

In 1965, a major breakthrough made by Kohn and Sham [12] showing that the energy of many-body of system can be expressed with density $n(\boldsymbol{r})$ composed by non-interacting electronic orbital wavefunctions (i.e. $n(\boldsymbol{r}) = \sum |\phi_i(\boldsymbol{r})|^2$). This concept of non-interacting electronic orbital wavefunction has been shown in the Hartree model (see Subsection 2.2.1). The total energy of the system according to the Kohn-Sham (KS) model is written in the form of

$$E_{KS}[n(\boldsymbol{r})] = T_s[n(\boldsymbol{r})] + \int n(\boldsymbol{r}) V_{ext}(\boldsymbol{r}) d\boldsymbol{r} + \frac{1}{2} \int n(\boldsymbol{r}) V_H(\boldsymbol{r}) d\boldsymbol{r} + E_{xc}[n(\boldsymbol{r})] \tag{2.14}$$

The first term is the non-interacting kinetic energy of electrons. The second term corresponds to the potential energy due to the electron-ion interactions. The third term is the Hartree potential (i.e. the third term in Eq. 2.7). It must be emphasized that the first three terms in Eq. 2.14 contains no information about the interactions amongst the electrons. The fourth term, exchange-correlation energy, is introduced to compensate for the interactions. Essentially, all of the unknown parts about the energy term are placed in the exchange-correlation term. Once the exchange-correlation energy is available, the total energy of the system can be solved using self-consistent field (SCF) method (see Fig. 2.1). The SCF is an iterative process in order to reach solution by first providing initial guess of the electronic density ($n^{in}(\boldsymbol{r})$). The electronic density will be used to construct the effective Hamiltonian. Then, a new set of orbital wavefunctions can be found by solving the single particle Schrödinger's equation. The solution to the KS energy is obtained once the $n^{in}(\boldsymbol{r})$ and $n^{out}(\boldsymbol{r})$ are consistent. Now, the remaining issue in the DFT method is to approximate the exchange-correlation functional.

2.2.3 Exchange-correlation Energy

Local Density Approximation. The Local Density Approximation (LDA) was first proposed by Kohn and Sham in their original breakthrough paper [12] as the simplest method describing the exchange-correlation energy. In the LDA, the homogeneous electron-gas formula is used for the

exchange-correlation energy. The LDA is an integral of over space of a function that depends only on the local density. The exchange-correlation energy density can be written as

$$E_{xc}^{LDA} = \int n(\boldsymbol{r})\epsilon_{xc}(n)d\boldsymbol{r} \tag{2.15}$$

where $\epsilon_{xc}(n) = \epsilon_x(n) + \epsilon_c(n)$. The exchange term is given by the Dirac functional [44]

$$\epsilon_x^{LDA}(n(\boldsymbol{r})) = -\frac{3}{4}\left(\frac{3}{\pi}\right)^{1/3} n^{1/3}(\boldsymbol{r}) \tag{2.16}$$

The values for $\epsilon_c(n)$ have been accurately determined from Quantum Monte Carlo calculations by Ceperly and Alder [45]. Meanwhile, the interpolation of Ceperly and Alder's data was provided by Vosko-Wilk-Nusair (VWN) [46], Perdew-Zunger (PZ) [47], and Perdew-Wang (PW) [48].

Generalized Gradient Approximation. The Generalized Gradient Approximation (GGA) is an major improvement to LDA by considering the gradient of the electron density. This can be written symbolically as

$$E_{xc}^{GGA} = \int \epsilon_{xc}^{GGA}(n(\boldsymbol{r}, \boldsymbol{\nabla} n(\boldsymbol{r}))d\boldsymbol{r} \tag{2.17}$$

The gradient introduces non-locality to the exchange and correlation interactions. There are two main GGA approaches: empirical and non-empirical. The empirical approach employs several fitting parameters to improve the accuracy on atomic and molecular properties. One of the most popular empirical approach is Becke-Lee-Parr-Yang (BLYP) [49, 50]. On the other hand, Perdew-Burke-Ernzerhof (PBE) [51] is the most popular non-empirical GGA approach in materials science. PBE-GGA is also used and extensively mentioned throughout the thesis.

2.2.3 Basis Sets

As it is generally unknown, the wavefunction is usually expanded in terms of a set of known functions. A single-electron wavefunction can be written as

$$\phi_i(\boldsymbol{r}) = \sum_j^{\infty} c_j \chi_j(\boldsymbol{r}) \tag{2.18}$$

where $\chi_j(\boldsymbol{r})$ belongs to a complete set of functions. In practical computation, it is impossible to use an infinite basis functions. Therefore, another approximation has to be imposed by only considering a finite number of functions. Ideally, the functions must behave as if the real wavefunction, decay to zero at large distance for isolated atoms or molecules. Here, two types of basis sets will be introduced: plane wave and atomic orbital.

Plane waves. In a solid crystal with periodicity, Bloch's theorem shows that the wavefunction for a particle can be expressed as the a product of planewave and a periodic function [52]

$$\phi_i(\boldsymbol{r}) = e^{i\boldsymbol{k}\cdot\boldsymbol{r}} u_{ik}(\boldsymbol{r}) \tag{2.19}$$

where $\boldsymbol{k}$ is the wave vector that only carries discrete values defined by the size of the crystal structure. The periodic function, $u_i(\boldsymbol{r})$, can be expanded in a set of plane waves like with reciprocal lattice vector of $\boldsymbol{G}$

$$\phi_i(\boldsymbol{r}) = \sum_{\boldsymbol{G}} c_{i,\boldsymbol{k}+\boldsymbol{G}} e^{i(\boldsymbol{k}+\boldsymbol{G})\cdot\boldsymbol{r}} \tag{2.20}$$

As mentioned before, an infinite plane waves is required to get a perfect expansion. However, the coefficients ($c_{i,\boldsymbol{k}+\boldsymbol{G}}$) must go to zero for high energy plane waves. Equivalently, the kinetic energy of the system will go to infinity. This truncation is justified and is set by an energy cutoff.

In addition, the plane waves as the basis set can also be applied to non-periodic systems by placing the center of a periodic supercell, i.e. creating vacuum around the cell. If the vacuum is large enough, the interaction between a cell with the neighboring cells is negligible. However, more suitable basis sets for molecular systems are based on atomic orbital basis sets.

Atomic orbital. Atomic orbitals as basis sets come in two forms: Slater-type orbitals and Gaussian type orbitals. The Slater-type orbitals can be written in spherical coordinates [53]

$$\phi^{nlm\eta}(r, \theta_0, \theta_1) = \alpha Y_{lm}(\theta_0, \theta_1) r^{n-1} e^{-\eta r} \tag{2.21}$$

where α is a normalization constant. Y_{lm} is the spherical harmonics depending on the angle θ_0 and θ_1. n, l, and m are the principal, angular, and magnetic quantum numbers. η is the radius of the orbit. The exponent term is similar to the exponential term in the one-electron atom solution (i.e. Hydrogen).

Furthermore, the Gaussian type orbital is expressed in

$$\phi^{nlm\eta}(r, \theta_0, \theta_1) = \alpha Y_{lm}(\theta_0, \theta_1) r^{2n-2-l} e^{-\eta r^2} \tag{2.22}$$

The prominent difference between the Gaussian-type and Slater-type orbitals is in the exponential term. In practice, the Slater-type yields to better accuracy. However, the Gaussian-type orbitals are computationally more efficient. As consequence, Gaussian-type orbitals are more frequently used in calculations.

2.2.3 Brillouin Zone Sampling

The Bloch's theorem reduces the infinite plane wave functions into finite wave functions. However, there are still an infinite number of discrete $\boldsymbol{k}$ points to be considered due to the energy of the unit cell is integrate over $\boldsymbol{k}$ within the first Brillouin Zone. Luckily, only a finite set of $\boldsymbol{k}$ points has to be sampled due to the $\boldsymbol{k}$ points that are closed together are similar. Thus, an interpolation scheme can be used. It is important to check the convergence when the interpolation scheme is used as finite number of $\boldsymbol{k}$ points can induce error in energy calculations.

The two most common schemes are the Chadi-Cohen scheme [54] and the Monkhorst-Pack scheme [55]. The later one is the most popular scheme. The Monkhorst-Pack scheme provides a

set of $\boldsymbol{k}$ points sampling as:

$$\boldsymbol{k} = u_p \boldsymbol{b}_1 + u_r \boldsymbol{b}_2 + u_s \boldsymbol{b}_3 \tag{2.23}$$

where $\boldsymbol{b}_1$, $\boldsymbol{b}_2$, and $\boldsymbol{b}_3$ are the reciprocal lattice vectors. The u_p, u_r, and u_s are determined by the relationship of

$$u_r = \frac{2r - q - 1}{2q} \tag{2.24}$$

where $r = 1, 2, 3, ..., q$, and q^3 is the total number of $\boldsymbol{k}$ points. This can be reduced significantly due to symmetry.

2.2.3 Pseudopotentials

For most systems, most physical properties depend on the valence electrons more than the core electrons, e.g. binding energy. Most of the core electrons stay chemically inert and do not participate in bonding. The pseudopotential takes account of this behavior of core electrons and replaces them by a weaker potential that acts on the active valence electrons. There two regions in a pseudopotential separated by radius, r_c. As seen in Fig. 2.2, inside of the core radius ($r < r_c$) the pseudo-wavefunction has smoother form than the all-electron wavefunction while retaining the same number of electrons, whereas the wavefunctions are correctly representing the outside of the core radius ($r < r_c$). The construction of pseudopotentials is non-unique. The larger r_c provides smoother potentials and reduce the number of plane waves, but the accuracy is also decreases.

A set of rules must be satisfied in order to ensure optimum smoothness and transferability of pseudopotential [56]:

1. All-electron energies and pseudo-eigenvalues must agree.

$$E_i^{AE}(\boldsymbol{r}) = E_i^{PP}(\boldsymbol{r}) \tag{2.25}$$

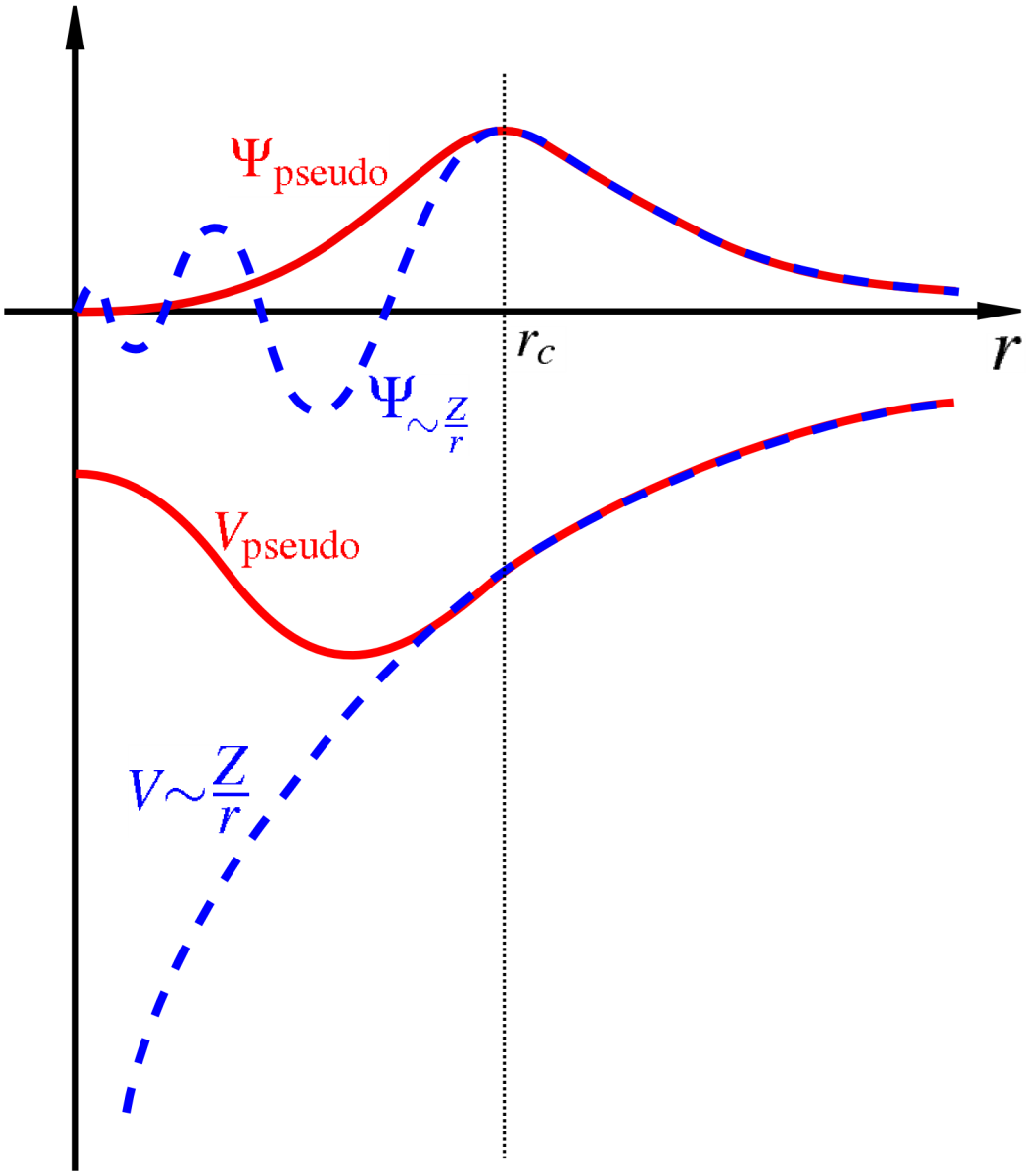


Figure 2.2: The comparison of between pseudo-wavefunction/pseudopotential (red) and the all-electron wavefunction/potential (blue) (from wikipedia).

2. The all-electron wavefunction and pseudo-wavefunction must agree when $r < r_c$.

$$\Psi^{AE}(\boldsymbol{r}) = \Psi^{PP}(\boldsymbol{r}), \text{ for } r \geq r_c \tag{2.26}$$

3. The total charge of pseudo-wavefunction must be equal to the charge of the all-electron wavefunction.

$$\int_0^{r_c} |\phi_i^{AE}(\boldsymbol{r})|^2 d\boldsymbol{r} = \int_0^{r_c} |\phi_i^{PP}(\boldsymbol{r})|^2 d\boldsymbol{r} \tag{2.27}$$

4. The scattering properties must be conserved.

Essentially, there are two kinds of pseudopotentials that satisfied the rules: norm-conserving soft pseudopotential [57] and Vanderbilt ultrasoft pseudopotential [58]. The "softer" pseudopotential implies that fewer plane waves are required to expand it. In the Vienna Ab initio Simulation (VASP) Package, the projector augmented wave (PAW) potentials are used [59, 60].

2.2.4 Summary

The summary of this section is illustrated in Fig. 2.3. The accurate calculation of energy of a system can be determined quantum-mechanically. Solving the Schrödinger's equation for many-body systems (e.g. silicon dimer) is an impossible task as the wavefunction has $3N_{ele}$ degree of freedom ($R^{3N_{ele}}$), where N_{ele} is the number of electrons. For example, there are $10^{3\text{x}28}$ integral operations needed at $10\text{x}10\text{x}10$ grid to compute the energy for a silicon dimer. Hartree model simplified the wavefunction by decomposing it into individual electronic orbital wavefunctions (Eq. 2.6). According to Hartree model, the Schrödinger's equation can be solved within $N_{ele}R^3$ dimensions. Now, the computation of energy of silicon dimer needs $28\text{x}10^3$ calculations. The reduction in dimensionality is a major leap in the field of computational physics, redeeming the impossible task into a tractable one.

Although Hartree model successfully reduces the computational effort, it has no practical use in any real simulation. This is due to the fermionic nature of electron is not considered in the

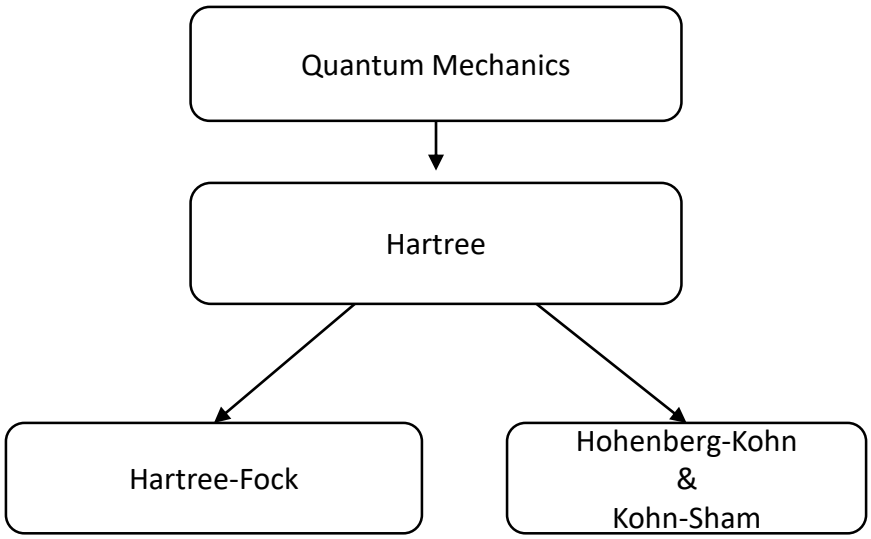


Figure 2.3: The summary for Section 2.

wavefunction. To compensate for this, Hartree-Fock model was introduced by imposing the Slater determinant to the wavefunction. Hartree-Fock model is a successful routine in many real applications such as $MgSiO_3$ [61], $MgCO_3$ [62], SiO_2 [63], etc. It takes account of exchange potential that is missing in the Hartree model. The exchange potential is non-local which makes Hartree-Fock model expensive.

DFT is the most successful model to date that uses the concept of individual electronic orbital from Hartree model. The two theorems of DFT was first formulated by Hohenberg and Kohn. The first theorem states that the external potential V_{ext} is a functional of electron density $n(r)$. The second theorem states that the ground-state energy of a system can be found as the ground-state of electron density is provided. Again, this is a profound scheme in reducing the computational cost to 3 dimensional problem (i.e. the electron density lives in 3D space). However, the error associates to the kinetic energy is too troublesome in real applications.

Later, Kohn and Sham devised a method which rearranging the Hamiltonian into a set of non-interacting individual electronic orbital system (as seen in the Hartree model). The caveat raises in the interaction part of the Hamiltonian is concealed in the exchange-correlation term ($E_{xc}[n(\boldsymbol{r})]$).

Essentially, several approximation schemes have to be introduced to reveal the unknown $E_{xc}[n(\boldsymbol{r}]$. The two most popular $E_{xc}[n(\boldsymbol{r})]$ are LDA and GGA. As the $E_{xc}[n(\boldsymbol{r})]$ portion is available, the total energy of the system can be solved iteratively via self-consistent field method.

2.3 Classical Force Field

Unlike the *ab initio* method, classical force field follows a functional form and try to fit the hyperparameters to the functional form from a set of experimental data or high accuracy *ab initio* calculations. This functional form is usually called interatomic potential. An interatomic potential has the general form of

$$\Phi = \Phi(\boldsymbol{r}_1, ..., \boldsymbol{r}_N, Z_1, ..., Z_N; \boldsymbol{\theta}) \tag{2.28}$$

where N represents the total number of atoms. $\boldsymbol{r}_1, ..., \boldsymbol{r}_N$ and $Z_1, ..., Z_N$ are the atomic Cartesian coordinates and species, respectively. Since classical force field is parametric, the potential energy is parameterized with parameters, $\boldsymbol{\theta}$. Nevertheless, the interatomic potential must satisfy certain laws of physics [64]:

1. The potential energy Φ must be invariant with respect to translational and rotational operations

$$\Phi(\boldsymbol{S}\boldsymbol{r}_1 + \boldsymbol{c}, ..., \boldsymbol{S}\boldsymbol{r}_N + \boldsymbol{t}, Z_1, ..., Z_N; \boldsymbol{\theta}) = \Phi(\boldsymbol{r}_1, ..., \boldsymbol{r}_N, Z_1, ..., Z_N; \boldsymbol{\theta}) \tag{2.29}$$

   for every rotation $\boldsymbol{S} \in$ SO(3) and translational vectors $\boldsymbol{t} \in \mathbb{R}^3$.

2. An interatomic potential must be invariant with respect to inversion operation

$$\Phi(-\boldsymbol{r}_1, ..., -\boldsymbol{r}_N, Z_1, ..., Z_N; \boldsymbol{\theta}) = \Phi(\boldsymbol{r}_1, ..., \boldsymbol{r}_N, Z_1, ..., Z_N; \boldsymbol{\theta}) \tag{2.30}$$

3. An interatomic potential must be invariant with respect to permutation of the coordinates of

atoms within the same species

$$\Phi(P[\boldsymbol{r}_1, ..., \boldsymbol{r}_N], Z_1, ..., Z_N; \boldsymbol{\theta}) = \Phi(\boldsymbol{r}_1, ..., \boldsymbol{r}_N, Z_1, ..., Z_N; \boldsymbol{\theta}) \tag{2.31}$$

where $P[\cdot]$ is permutation operator.

The first and second rules implies that the interatomic potential Φ can only be a function of distances between atoms.

2.3.1 Lennard-Jones Potential

Arguably, the simplest interatomic potential is Lennard-Jones potential [65, 66]:

$$\phi(r_{ij}) = 4\epsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] \tag{2.32}$$

where ϵ is the depth of the potential well, and σ is the distance which the inter-particle is zero. The first term determines the repulsion between atoms when they are close. This behavior is similar to that of short-range Pauli repulsion due to the overlapping electronic orbitals. The second term starts to take control as distance increases, representing attractive interaction due to the long range interactions such as van der Waals force or dispersion force.

2.3.2 Morse Potential

Morse potential is an interatomic potential model for a diatomic molecule. It is more suitable for covalently bonded molecules. The expression of Morse potential is

$$\phi(r_{ij}) = \epsilon(e^{-2\alpha(r_{ij}-r_0)} - 2e^{-\alpha(r_{ij}-r_0)}) \tag{2.33}$$

ϵ is the depth of the well, α governs the width of the potential, and r_0 is the equilibrium bond distance.

2.3.3 Buckingham Potential

Buckingham potential is a modification to the Lennard-Jones potential by replacing the repulsion term $(1/r_{ij}^{12})$ with an exponential term:

$$\phi(r_{ij}) = Ae^{-Br_{ij}} - \frac{C}{r_{ij}^6} \tag{2.34}$$

where A, B, and C are constants. The first and second terms constitute the repulsion and attraction interactions. Generally, the first term yields better physical description of the repulsive force due to the overlapping electron clouds.

2.3.4 Tersoff Potential

The Tersoff potential is a three-body potential functional. The potential includes angular effect of the force. The key idea of the potential is that the bonding strength depends on the local environment. An atom will form weaker bonds as its immediate neighborhood consists of many atoms. On the other hand, an atom will bond stronger to a few number of neighboring atoms. The potential was first applied to silicon [67] and later for carbon [68]. The Tersoff potential calculates the energy E of the system as

$$\begin{aligned} E &= \frac{1}{2}\sum_{i \neq j} V_{ij} \\ V_{ij} &= f_c(r_{ij})\Big[f_R(r_{ij}) + b_{ij} f_A(r_{ij})\Big] \end{aligned} \tag{2.35}$$

where f_c is a smooth cutoff function, and f_R and f_A are the repulsive (two-body term) and attractive pair potential (three-body term), respectively. These three terms can be explicitly written as

$$f_c(R) = \begin{cases} 1 & r < R - D \\ \frac{1}{2} - \frac{1}{2}\sin\frac{\pi}{2}\frac{r-R}{D} & R - D < r < R + D \\ 0 & r > R + D \end{cases} \tag{2.36}$$

$$
\begin{aligned}
f_R(r) &= Ae^{-\lambda_1 r} \\
f_A(r) &= -Be^{-\lambda_2 r}
\end{aligned}
\tag{2.37}
$$

R and D are parameters that are chosen to include the first-neighbor shell only. The main ingredient of this potential is included in the bij term. As mentioned, the strength of bonding between atoms relies on the local environment. It is due to the local environment, the bij term can amplify or shrink the importance of attractive force relative to the repulsive force, such that

$$
\begin{aligned}
b_{ij} &= \frac{1}{(1+\beta^n \zeta_{ij}^n)^{1/2n}} \\
\zeta_{ij} &= \sum_{k \neq i,j} f_c(r_{ij}) g(\theta_{ijk}) e^{\lambda_3^3 (r_{ij} - r_{ik})^3} \\
g(\theta) &= 1 + \frac{c^2}{d^2} - \frac{c^2}{(d^2 + (h - cos\theta)^2)}
\end{aligned}
\tag{2.38}
$$

The ζ term defines the coordinate number of i-th atom with respect to the two neighbors j and k with the bond-angle (θ) between bond ij and bond ik. The parameters A, B, λs, R, D, n, c, d, β, h for carbon and silicon are given in the paper [69].

2.3.5 Embedded Atom Model

Embedded Atom Model (EAM) determines the energy of a metallic material or a metallic alloy through pairwise interactions, which is metallic bonding in particular [70, 24, 25]. The EAM exploits the intuition of the physics of a given metal and the simplicity of the computational model, which is needed for large systems. The total energy of an atom i is given by

$$
E_i = F_\alpha \left(\sum_{j \neq i} \rho_\beta(r_{ij}) \right) + \frac{1}{2} \sum_{j \neq i} \phi_{\alpha\beta}(r_{ij}) \tag{2.39}
$$

where ϕ is a pair potential interaction, and α and β are the element types of atoms i and j. F is a function of the atomic electron density ρ. This indicates that the i-th atom is embedded in a host

of electron gas ρ created by its j neighboring atoms. F can have many forms. For example, the EAM function can be derived from nearly first-principles method to describe elemental Ni [71].

2.4 Machine Learning

Before diving into the theory of MLPs, some machine learning basics will be introduced. Furthermore, a light discussion to linear regression as a machine learning algorithm will be examined.

2.4.1 Machine Learning Basics

Mitchell defines machine learning as [72]: A computer program is said to learn from experience *E* with respect to some class of tasks *T* and performance measure *P*, if its performance at tasks in *T*, as measured by *P*, improves with experience E. Essentially, learning means attaining the ability to perform the task. Machine learning has encountered many breakthroughs from playing Go [73] in 2016 to self-driving cars [74]. Typical tasks that can be solved by machine learning algorithm to date are, includes but not limited to, classifications [75], regressions [29, 41], machine translation [76, 77], anomaly detection [78], denoising [79, 80], and etc. In general, machine learning algorithms can be categorized as unsupervised and supervised depending to the experience *E* that is allowed during the learning process. The experience *E* refers to **training set**.

Unsupervised Learning algorithms experience a dataset that only contains a set of inputs $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$. The algorithms can be decomposed into three groups determined from the their purposes: visualization [81], density estimation, and clustering. Ultimately, the subdivisions aim to learn the structural features or unknown patterns of the given dataset or experience *E*. For example, clustering groups subsets of the dataset according to the similarity within the subsets, while density estimation determines the probability distribution that generated a dataset. In visualization, the dataset inputs can projected into a two or three dimensional graphs. Unsupervised learning algorithms are usually computationally complex and not well-defined since there is no benchmark to the performance measure *P*.

Supervised Learning algorithms experience a dataset that contains a set of inputs $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$

and the corresponding targeted outputs $\{\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_n\}$. There are two common tasks of supervised learning: regression and classification. If the targeted outputs consist of values belonging to the set of distinct or discrete values, the task is called classification. Meanwhile, regression is referred to the continuous values of the outputs belonging to the set of any value within a finite or infinite interval. The abilities of a supervised learning method conducting the tasks *T* are evaluated by some performance measure, *P*.

In consequence, a quantitative measure must be design to evaluate the performance *P*. For instance, the accuracy measure (i.e. mean absolute error (MAE) or root mean squared error (RMSE)) is often used in regression model, corresponding to how well the regression model reproducing the true outputs. In classification, one can also measure the error rate, the gauge of how often the model produces incorrect outputs. Error rate often refers to the expected loss of 0-1. The common routine for machine learning practitioners is to evaluate the accuracy measure with data that have not been seen by the machine learning model. The unseen data is called **test set**.

In summary, machine learning can be categorized into supervised and unsupervised learning. Later in the section, one can identify that the MLP developments belong to the regression problem in supervised learning category. In addition, Principal Component Analysis (PCA), one of the unsupervised learning technique, is used to learn the unknown patterns of the input data for both training and test set. In the next subsection, the core mechanism of machine learning technique will be inspected as linear regression will be exploited.

2.4.2 Linear Regression as Machine Learning Algorithm

As mentioned, a machine learning algorithms is an algorithm that is capable to improve itself upon the performance by carrying out tasks via experience. Here, linear regression model will be examined to illustrate the machine learning definition. Linear regression model solves regression problem. This means that the model accept a vector input of $\boldsymbol{x} \in \mathbb{R}^n$ to predict a scalar output of

$y \in \mathbb{R}$. The predicted output $\hat{y}$ is a linear function of input $\boldsymbol{x}$:

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x} \tag{2.40}$$

where $\boldsymbol{w} \in \mathbb{R}^n$ is a vector parameters.

Parameters control the behavior of the machine learning algorithm, in this case linear regression model. If a weight (w_i) is negative in value, the prediction $\hat{y}$ will decrease as the input value x_i increases. The weight can be large in magnitude to emphasize the importance of the particular input value. On the other hand, zero weight means that the input value has no impact on the prediction. Now, the job is to determine the weight parameters.

First, one must define a performance measure between the predicted output $\hat{y}$ and the true output y. Mean square error (MSE) is one way to measure performance:

$$MSE = \frac{1}{m} \sum_i (y_i - \hat{y}_i)^2 \tag{2.41}$$

where m is the total number of data points. In order to optimize the weight parameters, one can minimize the MSE with respect to the parameters and solve the equation when the gradient is zero:

$$\boldsymbol{\nabla}_w MSE = 0 \tag{2.42}$$

After the derivation of the MSE gradient, the optimal weights ($\hat{\boldsymbol{w}}$ can be obtain by solving the system of equations known as normal equations:

$$\boldsymbol{w} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y} \tag{2.43}$$

The optimal weights depend on input value $\boldsymbol{x}$ and $\hat{y}$, which belong to data points. Due to its simplicity, increasing data points will no longer increase the accuracy of linear regression model at some point. However, linear regression model is sufficient to illustrate the underlying principles of machine learning. Later in the upcoming section, many machine learning algorithms will be

discussed in more details.

2.4.3 Unsupervised Learning: Principal Component Analysis

Principal Component Analysis (PCA) algorithm [82, 83] provides a mean of compressing the a vector input of value $\boldsymbol{x} \in \mathbb{R}^n$ to $\boldsymbol{x}' \in \mathbb{R}^{n\prime}$, where $n' < n$ without demeaning the true representation of $\boldsymbol{x}$. In another words, PCA learns a representation from $\boldsymbol{x}$ and lowers its dimensionality. PCA can represent a new input values $\boldsymbol{x}'$ which are linearly independent from each other.

Suppose that there is an input matrix of $\boldsymbol{X}$, where $\boldsymbol{X}$ is of the size of mxn. In real applications, n is usually much smaller than m. For example, the number of observations m is much larger than the number of input values n. The unbiased covariance matrix associated with $\boldsymbol{X}$ is given by

$$\boldsymbol{\Sigma}[\boldsymbol{X}] = \frac{1}{m-1}\boldsymbol{X}^T\boldsymbol{X} \tag{2.44}$$

Principal components (PCs) can be obtain through singular value decomposition such that $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$. $\boldsymbol{U}$ is a matrix of mxm, $\boldsymbol{S}$ is a mxn matrix, and $\boldsymbol{V}$ is a nxn matrix. It is important to note that $\boldsymbol{S}$ is a diagonal matrix up to n-th row such that

$$\begin{pmatrix} s_1 & 0 & ... & 0 \\ 0 & s_2 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & ... & s_n \\ 0 & 0 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & ... & 0 \end{pmatrix}_{m\text{x}n} \tag{2.45}$$

where $s_1 > s_2 > ... > s_n$ are the eigenvalues in hierarchical ordered. Therefore, $\boldsymbol{U}$ and $\boldsymbol{V}$ are

both hierarchical ordered as well. Hence, the covariance matrix can be recovered as follow

$$\boldsymbol{\Sigma}[\boldsymbol{X}] = \frac{1}{m-1}(\boldsymbol{USV}^T)^T(\boldsymbol{USV}^T) = \boldsymbol{VS}^2\boldsymbol{V}^T \tag{2.46}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary such that $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}$.

Through linear transformation such that $\boldsymbol{X}' = \boldsymbol{XV}$, the transformed matrix $\boldsymbol{X}'$ can be derived in term of singular value decomposition:

$$\begin{aligned} \boldsymbol{X}' &= \boldsymbol{XV} \\ &= \boldsymbol{USV}^T\boldsymbol{V} \\ &= \boldsymbol{US} \end{aligned} \tag{2.47}$$

According to Eq. 2.45, the matrix $\boldsymbol{X}'$ can be reduced to nxn matrix such that

$$\boldsymbol{X}' = \boldsymbol{U}_{red}\boldsymbol{S}_{red} \tag{2.48}$$

Furthermore, the covariance of $\boldsymbol{X}'$ can be written as

$$\begin{aligned} \boldsymbol{\Sigma}[\boldsymbol{X}'] &= \frac{1}{m-1}(\boldsymbol{X}')^T\boldsymbol{X}' \\ &= \boldsymbol{V}^T\boldsymbol{X}^T\boldsymbol{XV} \\ &= \boldsymbol{V}^T\boldsymbol{VS}^2\boldsymbol{V}^T\boldsymbol{V} \\ &= \frac{1}{m-1}\boldsymbol{S}^2 \end{aligned} \tag{2.49}$$

The elements in $\boldsymbol{X}'$ are essentially mutually uncorrelated since $\boldsymbol{S}$ is diagonal matrix. PCA possesses the ability to transform data into elements of mutually uncorrelated representation.

Since $\boldsymbol{X}'$ are mutually uncorrelated, and the $\boldsymbol{S}$ are hierachically arranged, a machine learning practitioner can choose up to n' principal components, where $n' < n$ such that the total variation

in the data is retained:

$$\frac{\sum_{i=1}^{n'} \boldsymbol{S}_{ii}}{\sum_{i=1}^{n} \boldsymbol{S}_{ii}} \geq \epsilon \tag{2.50}$$

In a practical usage, the total variation (ϵ) is usually measured from 95% up to 99%. This implies that the variation of choosing n' number of principal components represents 95% up to 99% of the entire dataset.

2.5 Machine Learning Potentials

2.5.1 Background

As previously mentioned, *ab-initio* calculations can accurately describe most systems with high computational price. Typically, the price is too high for atomistic simulations or crystal structure prediction with large number of atoms. Indeed, most atomistic simulations are normally conducted with classical force fields. While the accuracy of classical force fields is overlooked due to classical force fields scale linearly with the number of atoms, enabling long simulations for large systems. Nevertheless, it is possible to apply *machine learning* concept for constructing the PES, allowing computations to be as accurate as the DFT accuracy without sacrificing too much computational time.

The MLP developments belong to supervised learning category, in particular regression problem. The inputs are the atomic coordinates and species in the configurations. The more sophisticated inputs are atom-centered descriptors. Atom-centered descriptors are mapping of the coordinates and species of atoms along with their neighboring information in an atomic configuration into symmetry-invariant numerical values. The details of atom-centered descriptors deserves a section itself (see Section 2.5.3). Although other target values such as formation energies can be considered as the targeted outputs, the outputs are usually the total potential energies of the configurations.

Technically speaking, the development of classical force field can be considered as MLPs. Nevertheless, the mathematical forms of classical force fields are designed accordingly to the un-

derlying physics in the materials systems. By training the classical force fields with more data, the force fields will likely to gain insignificant improvement. On the other hand, the mathematical forms of MLPs are designed to be fitted against a large amount of data, as the MLPs will gain systematical improvement upon observation of even more data.

Among many different ML models, two regression techniques are becoming increasingly popular in the materials modelling community. They include the neural networks and Gaussian process regressions. The neural networks approach has an unbiased mathematical form that can adapt to any set of reference points through an iterative fitting process given "enough" training data. The first well accepted neural networks potential (NNP) was originally applied to elemental silicon system by Behler and Parrinello [42], which demonstrated that the NNP was able to reproduce the energetic sequences of many silicon phases, as well as the radial distribution function of a silicon melt at 3000 K from DFT simulation. To gain a better predictive power, they also proposed to use a series of symmetry functions (see Section 2.5.3.2), instead of the Cartesian coordinates, as the descriptors to represent the atomic environment. Since then, many attempts have been undertaken to improve the capability of neural networks approach [84]. The accomplishments of neural networks approach have been extended to multi-component [33, 85] and organic [86] systems. In addition, Gaussian Approximation Potential (GAP), in conjunction with the bispectrum coefficients of atomic neighbour density (see Section 2.5.3.4), was first introduced to model the carbon, silicon, germanium, iron, and gallium nitride [40]. GAP was further enhanced by replacing bispectrum coefficients with smooth overlapping power spectrum coefficients with explicit radial basis [87]. Similar to GAP, Thompson *et al.* [36] developed (quadratic) Spectral Neighbor Analysis Potential (SNAP) method based on the Taylor expansion of bispectrum coefficients. In addition, linear regression model based on the moment tensor—comparable to atomic environments inertia tensors—as the descriptor [88] was also demonstrated to be a competitive approach. Many applications based on different MLP models have shown that machine learning potentials work remarkably well in different types of atomistic simulations [89, 90, 91, 92, 93, 94].

2.5.2 Mathematical Form

In this section, in-depth discussions of the two main ingredients in creating MLP will be discussed: atom-centered descriptor and regression technique. The construction of the total energy of a crystal structure can be written as the collections of atomic energy contributions, in which is a functional ($\mathscr{E}$) of the atom-centered descriptor ($\boldsymbol{X}_i$):

$$E_{\text{total}} = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N} \mathscr{E}_i(\boldsymbol{X}_i) \tag{2.51}$$

Specifically, the functional represents regression techniques such as neural networks or generalized linear regressions.

Since neural networks and generalized linear regressions have well-defined functional forms, the analytic derivatives can be derived by applying the chain rule to obtain the force at each atomic coordinate, $\boldsymbol{r}_m$:

$$\boldsymbol{F}_m = -\sum_{i=1}^{N} \frac{\partial \mathscr{E}_i(\boldsymbol{X}_i)}{\partial \boldsymbol{X}_i} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m} \tag{2.52}$$

Force is an important property to accurately describe the local atomic environment especially in geometry optimization and MD simulation. Finally, the stress tensor is acquired through the virial stress relation:

$$\boldsymbol{S} = -\sum_{m=1}^{N} \boldsymbol{r}_m \otimes \sum_{i=1}^{N} \frac{\partial \mathscr{E}_i(\boldsymbol{X}_i)}{\partial \boldsymbol{X}_i} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m}, \tag{2.53}$$

where $\otimes$ is the outer product.

According to Eqs. 2.52 and 2.53, one needs to compute the energy derivative $\frac{\partial \mathscr{E}}{\partial \boldsymbol{X}}$ and the derivatives of descriptor X with respect to the atomic positions. For a structure with N atoms and L descriptors, the energy derivative is a 2D array of $[N, L]$. The force related derivative (dxdr) can be best organized as a 4D array with the dimension of $[N, N, L, 3]$. Note that dxdr$[i, j, :, :]$ is zero when the i-j atomic pair has a distance larger than the cutoff distance. Thus, it may become a sparse array when the structure has a large number of atoms. Correspondingly, one can easily derive the 5D rdxdr array by multiplying r to each dxdr according to the outer product. In Python,

one can simply compute the forces and stresses based on the following Einstein summation as follow:

```
import numpy as np

"""
Einstein summation to compute force and stress.
dedx: 2D array [N, L]
dxdr: 4D array [N, N, L, 3]
rdxdr: 5D array [N, N, L, 3, 3]
force: 2D array [N, 3]
stress: 2D array [3, 3]
"""

force = -np.einsum("ik, ijkl->jl", dedx, dxdr)
stress = -np.einsum("ik, ijklm->lm", dedx, rdxdr)
```

Listing 2.1: Force and stress computation in Python.

2.5.3 Atom-centered Descriptors

Descriptor—a representation of a crystal structure—plays a critical role in constructing reliable MLP. If the MLP is directly mapped from the atomic positions or the Cartesian coordinates, it can only describe systems with the same number of atoms due to the fixed length of the regression input. In addition, Cartesian coordinates are poor descriptors in describing the structural environment of the system, restricted by the periodic boundary conditions. While the total energy of the structure remains the same by translation, rotational, or permutation operations, the atomic positions will change. A good descriptor must satisfy the following requirements:

1. A set of descriptors must invariant with respect to the choice of the frame of reference such as *translation*, *rotation*, and *inversion* of the system and *permutation* of atoms within the same species.

2. A set of descriptors must be complete. A set of descriptors is complete if it uniquely determines the atomic environments. If a subset of the set of descriptors can achieve the uniqueness, then the entire set is over-complete. This requirement enables one-to-one mapping between the descriptors and the atomic environments. In other words, one-to-one mapping cannot be attained unless the descriptors are exactly complete.

3. A set of descriptors must be continuous and differentiable. Continuous descriptor establishes smoothness in the prediction of the PES. It will be challenging to machine learning model to recuperate the smooth PES with non-continuous descriptor. It is important for the descriptor to be differentiable as some of the physical properties such as elastic constants rely on the derivatives.

4. The number of descriptors must be *independent* to the number of atom neighbors.

5. Ideally, the computation of descriptors should be cheap.

The descriptors that satisfied the requirements above will be called atom-centered descriptors in this writing. Several types of atom-centered descriptors have been developed in the past few years [95].

In the atom-centered descriptors, one usually needs to consider the neighboring environment for the centered atom within a cutoff radius of R_c. To ensure the descriptor mapping from the atomic positions smoothly approaching zero at the R_c, a cosine cutoff function (f_c) is included to every mapping scheme:

$$f_c(R) = \begin{cases} \frac{1}{2}\cos\left(\pi\frac{R}{R_c}\right) + \frac{1}{2} & R \leq R_c \\ 0 & R > R_c \end{cases} \tag{2.54}$$

where R is distance. The cutoff function is zero at R_c and the intensity decreases as R approaches

R_c. Consequently, the derivative of the cosine cutoff function is:

$$\frac{\partial f_c}{\partial R} = \begin{cases} -\frac{\pi}{2R_c} \sin\left(\pi \frac{R}{R_c}\right) & R \leq R_c \\ 0 & R > R_c \end{cases} \tag{2.55}$$

One should heed of the importance of the vanishing derivative of cutoff function at R_c, which is important in describing the force. By definition, there is no discontinuity as the slope decays to zero at R_c. Additionally, other types of cutoff functions are available in Appendix A.

2.5.3 Coulomb Matrix

Coulomb matrix has been widely used due to its simplicity (Req. #5). Coulomb matrix encompasses self interaction based on the nuclear charge and Coulomb repulsion between two nuclei [96, 97].

$$M_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{Rij} & \text{for } i \neq j \end{cases} \tag{2.56}$$

Logically, the Coulomb matrix can be upgraded for periodic crystals through Ewald summation that includes long range interaction calculated in reciprocal space. In addition, many-body tensor representation—derives from Coulomb matrix while related to bag of bonds which corresponds to different types of bonding in molecular systems—can be used for both finite and periodic systems when interpretability/visualization is desirable [98]. These descriptors have been widely used to model the molecules [99, 100].

The coulomb matrix captures the translational, rotational, and inversion symmetries, but it unsuccessfully describes the difference in permuting two atoms with the same species. Another possible source of error for MLP development is that Coulomb matrix may be able to distinguish two chemical environments in isolation, it may not be able to distinguish the two well-separated paired chemical environments [101]. In another words, Coulomb matrix as descriptors is not complete (Req. #2). Hence, better descriptors needs to be developed.

2.5.3 *(Weighted) Atom-centered Symmetry Functions (G)*

The atom-centered symmetry functions (ACSFs) are the very first types of descriptors used in the MLP development [42] that satisfy all of the requirements. In general, there are two classes of ACSFs: radial and angular symmetry functions [29]. The radial symmetry function or $G^{(2)}$ describes the radial distribution of the atomic environment, and the angular symmetry functions, $G^{(4)}$ and $G^{(5)}$, account for the three-body angular distribution of atoms in the neighborhood. The $G^{(2)}$ is expressed as the sum of the radial distances between the center atom i and the neighbor atoms j as follow:

$$G_i^{(2)} = \sum_{j \neq i} e^{-\eta(R_{ij}-R_s)^2} \cdot f_c(R_{ij}) \tag{2.57}$$

Here, $G^{(2)}$ value is controlled by the width (η) and the shift (R_s).

$G^{(4)}$ and $G^{(5)}$ symmetry functions are a few of many ways to capture the angular information via three-body interactions (θ_{ijk}). As the structures are constraint by the periodic boundary condition, a three-body periodic description such as $\cos(\theta_{ijk})$ is used. The explicit form of $G^{(4)}$ and $G^{(5)}$ are:

$$\begin{aligned} G_i^{(4)} =& 2^{1-\zeta} \sum_{j \neq i} \sum_{k \neq i,j} [(1 + \lambda \cos\theta_{ijk})^{\zeta} \cdot e^{-\eta(R_{ij}^2+R_{ik}^2+R_{jk}^2)} \cdot \\ & f_c(R_{ij}) \cdot f_c(R_{ik}) \cdot f_c(R_{jk})] \end{aligned} \tag{2.58}$$

$$\begin{aligned} G_i^{(5)} =& 2^{1-\zeta} \sum_{j \neq i} \sum_{k \neq i,j} [(1 + \lambda \cos\theta_{ijk})^{\zeta} \cdot e^{-\eta(R_{ij}^2+R_{ik}^2)} \cdot \\ & f_c(R_{ij}) \cdot f_c(R_{ik})] \end{aligned} \tag{2.59}$$

ζ determines the strength of angular information. The degree of ζ is normalized by $2^{1-\zeta}$ for unvarying the values of $G^{(4)}$ and $G^{(5)}$ symmetry functions due to ranges of ζ. λ values are set to +1 and -1, for inverting the shape of the cosine function. The difference between $G^{(4)}$ and $G^{(5)}$

symmetry functions is in the interactions between the neighbors j and k. The modification in $G^{(5)}$ symmetry function yields in dampening value of $G^{(5)}$, which can be beneficial in representing larger atomic separation between the two neighbors.

Clearly, the number of ACSFs will grow depending on chemical species as the separations of chemical species are needed. For instance, in a binary AB system, the number of $G^{(2)}$ ACSFs on specie A need to double to distinguish A-A and A-B pair interactions. For $G^{(4)}$, three different triplets A-A-A, A-A-B, B-A-B (where the middle position denotes the center atom) will be needed. To avoid this unpleasant growth, one can apply a weighting parameter based on the chemical species when counting these atomic pairs and triplets. One popular choice is simply to use the atomic number as the weighting parameters. Hence, Gastegger and coauthors proposed the weighted version of ACSF [102], in which each component of the radial and angular symmetry functions in Eqs. (2.57, 2.58, 2.59) can be multiplied by the followings:

$$\text{the weighted ACSF:} \quad \begin{cases} Z_j & \text{radial} \\ Z_j Z_k & \text{angular} \end{cases}$$

where Z_j, Z_k represents the atomic number of neighboring atom j and k.

To obtain a satisfactory MLP model, one has to choose a set of parameters to construct the (w)ACSF descriptors, which may require demanding human intervention [102, 103, 104]. As mentioned, the choice of λ is straightforward. In general, ζ takes the value of 1. Increasing ζ focuses on the strength of the angular information in region close to 0° and 180°, and decreasing it will weaken the contribution of angular information at around 90°. Since the exponential term has larger effect on the symmetry functions, the selection of η and R_s can be more elaborate. The selection routines are done by fixing η while varying R_s or vice versa [102, 105, 106].

Here, the parameters selection method suggested by Imbalzano *et. al.* will be discussed. First,

the (w)ACSF can be generated by centering the reference atom (i.e. R_s=0) as the η varies:

$$\eta_m = \left(\frac{n^{m/n}}{R_c}\right)^2 \tag{2.60}$$

where n is the chosen number of intervals, and m = $\{0, 1, ..., n\}$. Second, the parameter R_s can be selected by the following:

$$R_{s,m} = \frac{R_c}{n^{m/n}} \tag{2.61}$$

while the η are selected as follow:

$$\eta_{s,m} = \frac{1}{(R_{s,n-m} - R_{s,n-m-1})^2} \tag{2.62}$$

The selection method is effective by choosing finer grid closer to the central atom, while it creates wider grid as the distances increases. The phenomena can be seen in in Fig. 2.4.

2.5.3 Embedded Atom Density (ρ)

Embedded atom density (EAD) descriptor [105] is inspired by embedded atom method (EAM)—description of atomic bonding by assuming each atom is embedded in the uniform electron cloud of the neighboring atoms [24, 70]. In EAD, the electron density is modified by including the square of the linear combination the atomic orbital components:

$$\rho_i = \sum_{l_x,l_y,l_z}^{l_x+l_y+l_z=L_{\max}} \frac{L_{\max}!}{l_x!l_y!l_z!}\left(\sum_{j\neq i}^{N} Z_j \Phi(R_{ij})\right)^2 \tag{2.63}$$

where Z_j represents the atomic number of neighbor atom j. L_{max} is the quantized angular momentum, and $l_{x,y,z}$ are the quantized directional-dependent angular momentum. For example, $L_{max} = 2$ corresponds to the d orbital. Lastly, the explicit form of Φ is:

$$\Phi(R_{ij}) = \frac{x_{ij}^{l_x} y_{ij}^{l_y} z_{ij}^{l_z}}{R_c^{l_x+l_y+l_z}} \cdot e^{-\eta(R_{ij}-R_s)^2} \cdot f_c(R_{ij}) \tag{2.64}$$

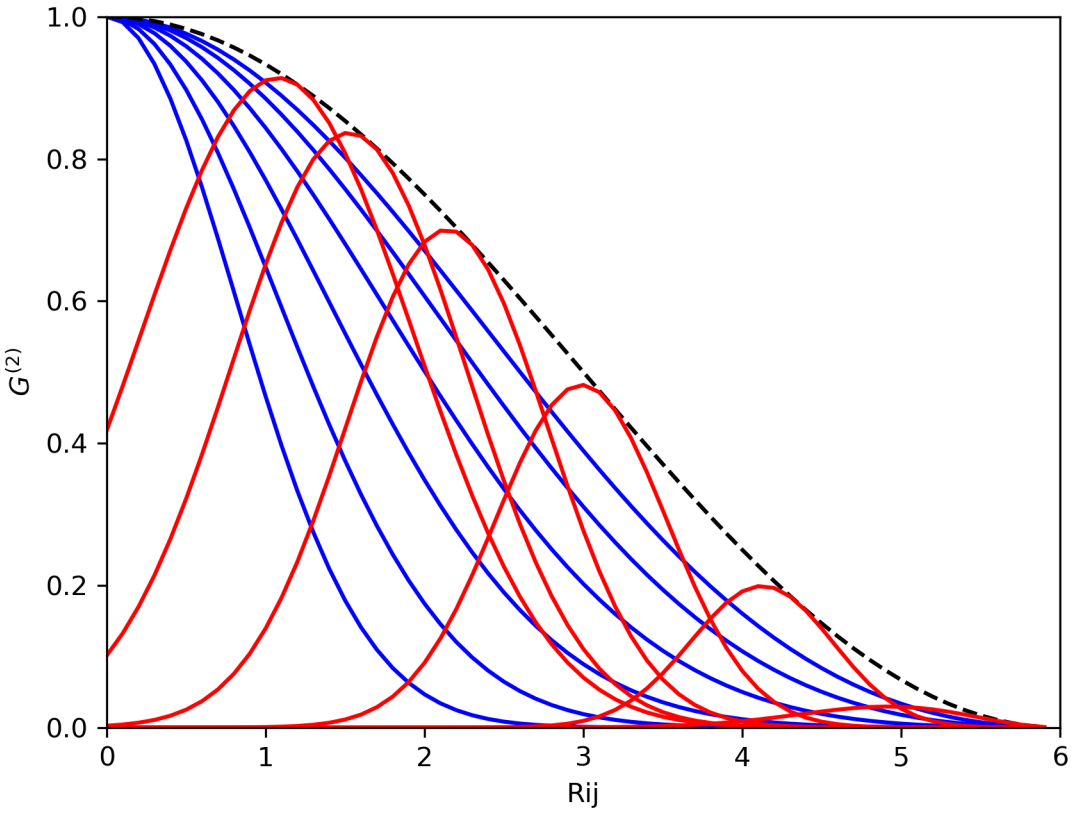


Figure 2.4: $G^{(2)}$ symmetry function generated using $n = 5$ and $R_c = 6$Å. The red curves are the radial symmetry functions with shifted R_s and η (Eq. 2.61 and 2.62). The blue curves are the results of the centring R_s method, as described in Eq. 2.60. The black dashed curve represents the cosine cutoff function with $R_c = 6$Å.

According to quantum mechanics, ρ follows the similar procedure in determining the probability density of the states, i.e. the Born rule.

EAD can be regarded as an alternative version of ACSF without classification between the radial and angular term. The angular or three-body term is implicitly incorporated in when $L_{\max} > 0$ [105]. By definition, the computation cost for calculating EAD is cheaper than angular symmetry functions by avoiding the extra sum of the k neighbors. In term of usage, the parameters η and R_s are similar to the strategy used in the Gaussian symmetry functions, and the maximum value for L_{max} is 3, i.e. up to f orbital.

2.5.3 SO(4) Bispectrum (B)

The SO(4) bispectrum components [40, 87, 36] are another type of atom-centered descriptor based on the harmonic analysis of the atomic neighbor density function on the 3-sphere. The atomic neighbor density function is given by [87]:

$$\rho(\boldsymbol{r}) = \delta(\boldsymbol{r}) + \sum_{i}^{R_c} w_i f_c(\boldsymbol{r}_i)\delta(\boldsymbol{r} - \boldsymbol{r_i}) \tag{2.65}$$

where w_i is a species dependent weight factor and f_c is a cutoff function. The cutoff function is f_c is introduced to ensure that the atomic neighbor density function goes smoothly to zero at the cutoff.

Then we map the atomic neighbor density function from 3-D euclidean space to another 3-D space, the surface of a four dimensional hypersphere:

$$\begin{aligned} s_1 &= r_0 \cos\omega \\ s_2 &= r_0 \sin\omega \cos\theta \\ s_3 &= r_0 \sin\omega \sin\theta \cos\phi \\ s_4 &= r_0 \sin\omega \sin\theta \sin\phi, \end{aligned}$$

where r_0 is a parameter and the polar angles are defined by:

$$\begin{aligned} \theta &= \arccos\left(\frac{z}{r}\right) \\ \phi &= \arctan\left(\frac{y}{x}\right) \\ \omega &= \frac{\pi r}{r_0} \end{aligned} \tag{2.66}$$

The Winger-D matrix elements ($D^j_{m',m}$) are the harmonic functions on the 3-sphere, therefore an arbitrary function defined on the 3-sphere can be expanded in terms of Wigner-D matrix elements. Here we expand the atomic neighbor density function on the 3-sphere in terms of Wigner-D

matrices.

$$\rho(\boldsymbol{r}) = \sum_{j=0}^{+\infty} \sum_{m',m=-j}^{+j} c^{j}_{m',m} D^{j}_{m',m}(\omega;\theta,\phi)$$

where the expansion coefficients $c^{j}_{m',m}$ are given by the following inner product

$$c^{j}_{m',m} = \left\langle D^{j}_{m',m} | \rho(\boldsymbol{r}) \right\rangle = D^{*j}_{m',m}(\mathbf{0}) + \sum_{i}^{r_i \leq R_{\mathrm{c}}} f_{\mathrm{c}}(r_i) D^{*j}_{m',m}(\omega_i;\theta_i,\phi_i) \tag{2.67}$$

Finally, the SO(4) bispectrum components can then be calculated using third order products of the expansion coefficients:

$$\begin{aligned} B^{j_1,j_2,j}_{i} = & \sum_{m',m=-j}^{j} c^{*j}_{m',m} \sum_{m'_1,m_1=-j_1}^{j_1} c^{j_1}_{m'_1,m_1} \times \\ & \sum_{m'_2,m_2=-j_2}^{j_2} c^{j_2}_{m'_2,m_2} C^{jj_1j_2}_{mm_1m_2} C^{jj_1j_2}_{m'm'_1m'_2}, \end{aligned} \tag{2.68}$$

where C is a Clebsch-Gordan coefficient. Finally, the derivation of the derivatives can be found in Appendix B.

2.5.3 Smooth SO(3) Power Spectrum (P)

The Smooth SO(3) Power Spectrum components were been proposed to describe the atomic local environment [87]. In contrast to the SO(4) bispectrum components, the Smooth SO(3) power spectrum is based on an alternative atomic neighbor density while also expanded on the 2-sphere and a radial basis. The alternative atomic neighbor density is defined in terms of Gaussians as follows:

$$\rho'(\boldsymbol{r}) = \sum_{i}^{r_i \leq R_{\mathrm{c}}} w_i e^{-\alpha |\boldsymbol{r} - \boldsymbol{r_i}|^2}, \tag{2.69}$$

Then the atomic neighbor density function is then expanded in terms of spherical harmonics and a radial basis $g_n(r)$ as shown in Eq. 2.69:

$$\rho'(\boldsymbol{r}) = \sum_{l=0}^{+\infty} \sum_{m=-l}^{+l} c_{nlm} g_n(r) Y_{lm}(\hat{\boldsymbol{r}})$$

where the expansion coefficients c_{nlm} are given by

$$c_{nlm} = \langle Y_{lm} g_n(r) | \rho' \rangle = 4\pi \sum_{i}^{r_i \leq R_c} w_i e^{-\alpha r_i^2} Y_{lm}^*(\hat{\boldsymbol{r}}_{\boldsymbol{i}}) \times \int_0^{R_c} r^2 g_n(r) I_l(2\alpha r r_i) e^{-\alpha r^2} dr \tag{2.70}$$

where I_l is a modified spherical Bessel function of the first kind. A convenient radial basis for this purpose, $g_n(r)$, consisting of cubic and higher order polynomials, orthonormalized on the interval $(0, R_c)$ has been suggested by Bartok [87].

$$g_n(r) = \sum_{\alpha} W_{n,\alpha} \phi_\alpha(r) \tag{2.71}$$

where $W_{n,\alpha}$ are the orthonormalization coefficients given by the relation to the overlap matrix $\boldsymbol{S}$ by $\boldsymbol{W} = \boldsymbol{S}^{-1/2}$ and

$$\phi_\alpha(r) = (R_c - r)^{\alpha+2} / N_\alpha$$

$$N_\alpha = \sqrt{\frac{2 r_{\text{cut}}^{(2\alpha+7)}}{(2\alpha+5)(2\alpha+6)(2\alpha+7)}}$$

and the elements of the overlap matrix $\boldsymbol{S}$ are given by

$$\begin{aligned} S_{\alpha\beta} &= \int_0^{r_{\text{cut}}} r^2 \phi_\alpha(r) \phi_\beta(r) dr \\ &= \frac{\sqrt{(2\alpha+5)(2\alpha+6)(2\alpha+7)(2\beta+5)(2\beta+6)(2\beta+7)}}{(5+\alpha+\beta)(6+\alpha+\beta)(7+\alpha+\beta)} \end{aligned} \tag{2.72}$$

and finally, the Smooth SO(3) power spectrum is given by

$$P_i^{n_1 n_2 l} = \sum_{m=-l}^{+l} c_{n_1 l m} c^*_{n_2 l m} \tag{2.73}$$

2.5.4 Regression Models

Here, we discuss the regression model, i.e., the functional form ($\mathscr{E}$) presented in Eq. 2.51. Each regression model is species-dependent, i.e. as the the number of species increases, the regression parameters will increase. For the sake of simplicity, we will explain the regression models for the single-species system.

In any regression model, the objective is to minimize a loss function which describes the discrepencies between the prediction and true reference values (including energy, force, and stress tensors) for each atomic configuration in the training dataset.

$$\begin{aligned}\Delta = \frac{1}{2M} \sum_{i=1}^{M} \Bigg[\left(\frac{E_i - E_i^{\text{Ref}}}{N_{\text{atom}}^i} \right)^2 + \frac{\beta_f}{3N_{\text{atom}}^i} \sum_{j=1}^{3N_{\text{atom}}^i} (F_{i,j} - F_{i,j}^{\text{Ref}})^2 \\ + \frac{\beta_s}{6} \sum_{p=0}^{2} \sum_{q=0}^{p} (S_{pq} - S_{pq}^{\text{Ref}})^2 \Bigg]\end{aligned} \tag{2.74}$$

where M is the total number of structures in the training pool, and N_i^{atom} is the total number of atoms in the i-th structure. The superscript Ref corresponds to the target property. β_f and β_s are the force and stress coefficients respectively. They scale the importance between energy, force, and stress contribution as the force and stress information can overwhelm the energy information due to their sizes. Additionally, a regularization term can be added to induce penalty on the entire parameters preventing overfitting:

$$\Delta_{\text{p}} = \frac{\alpha}{2M} \sum_{i=1}^{m} (\boldsymbol{w}^i)^2 \tag{2.75}$$

where α is a dimensionless number that controls the degree of regularization.

Clearly, one has to choose differentiable functional as well as its derivative due to the existence of force (F) and stress (S) contribution along with the energy (E) in the loss function. In the following sections, generalized linear regression and neural network regression will be introduced.

2.5.4 Generalized Linear Regression

This regression methodology is a type of polynomial regression. Essentially, the quantum-mechanical energy, forces, and stress can be expanded via Taylor series with atom-centered descriptors as the independent variables:

$$E_{\text{total}} = \gamma_0 + \boldsymbol{\gamma} \cdot \sum_{i=1}^{N} \boldsymbol{X}_i + \frac{1}{2} \sum_{i=1}^{N} \boldsymbol{X}_i^T \cdot \boldsymbol{\Gamma} \cdot \boldsymbol{X}_i + \cdots \tag{2.76}$$

where N is the total atoms in a structure. γ_0 and $\boldsymbol{\gamma}$ are the weights presented in scalar and vector forms. $\boldsymbol{\Gamma}$ is the symmetric weight matrix (i.e. $\boldsymbol{\Gamma}_{12} = \boldsymbol{\Gamma}_{21}$) describing the quadratic terms. In this equation, we only restricted the expansion up to polynomial 2 due to to enormous increase in the weight parameters.

In consequence, the force on atom j and the stress matrix can be derived according to Eqs. (2.52, 2.53), respectively:

$$\boldsymbol{F}_m = -\sum_{i=1}^{N} \left(\boldsymbol{\gamma} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m} + \frac{1}{2} \left[\frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_m} \cdot \boldsymbol{\Gamma} \cdot \boldsymbol{X}_i + \boldsymbol{X}_i^T \cdot \boldsymbol{\Gamma} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m} \right] \right) \tag{2.77}$$

$$\boldsymbol{S} = -\sum_{m=1}^{N} \boldsymbol{r}_m \otimes \sum_{i=1}^{N} \left(\boldsymbol{\gamma} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m} + \frac{1}{2} \left[\frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_m} \cdot \boldsymbol{\Gamma} \cdot \boldsymbol{X}_i + \boldsymbol{X}_i^T \cdot \boldsymbol{\Gamma} \cdot \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_m} \right] \right) \tag{2.78}$$

Note that the energy, force, and stress share the weight parameters $\{\gamma_0, \boldsymbol{\gamma}_1, ..., \boldsymbol{\gamma}_n, \boldsymbol{\Gamma}_{11}, \boldsymbol{\Gamma}_{12}, ..., \boldsymbol{\Gamma}_{nn}\}$, where n is total the number of descriptors of the center atom. Once the energy, force and stress tensors are known, the derivative of the loss function can be evaluated. Finding the zero derivative of loss function (Eq. 2.74) in linear regression is equivalent to solve a set of linear equations of

$$\boldsymbol{Ax} = \boldsymbol{b}.$$

2.5.4 Neural Network Regression

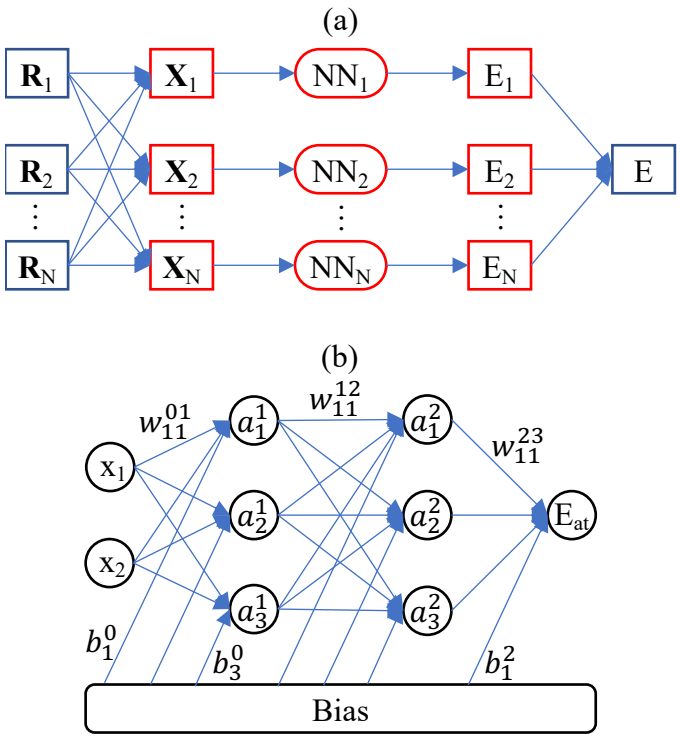


Figure 2.5: (a) A schematic diagram of high-dimensional neural networks. (b) A zoom-in version of the color-coded part in (a).

Compared to the linear regression, neural networks provides more flexible functionals to fit a large datasets. Figure 2.5 shows a schematic diagram based on neural networks training. Prior to the neural networks architecture, the atom-centered descriptors are mapped based on the atomic environment of a structural configuration as discussed in the previous section. These descriptors serve as the input to the neural networks architecture and are arranged in the first layer as shown in Figure 2.5b. The next layers are the hidden layers. Neural networks can simply cast more

weights parameters as needed through increasing number of hidden layers and/or hidden layers nodes without the increasing number of descriptors. The nodes in the hidden layers carry no physical meaning.

For each of the hidden nodes, activation functions such as Tanh and Sigmoid functions are frequently used in our NNP implementation. While ReLU as an activation function is extremely popular in image processing, we believe ReLU is not an appropriate choice in constructing MLP, due to the function carries discontinuity at zero. These nodes are connected via the weights and biases and propagate in forward direction only. In the end, the output node represents the atomic energy. A mathematical form to determine any node value can be written as:

$$X_{n_i}^{l} = a_{n_i}^{l}\left(b_{n_i}^{l-1} + \sum_{n_j=1}^{N} W_{n_j,n_i}^{l-1,l} \cdot X_{n_j}^{l-1}\right) \tag{2.79}$$

The value of a neuron ($X_{n_i}^{l}$) at layer l can determined by the relationships between the weights ($W_{n_j,n_i}^{l-1,l}$), the bias ($b_{n_i}^{l-1}$), and all neurons from the previous layer ($X_{n_j}^{l-1}$). $W_{n_j,n_i}^{l-1,l}$ specifies the connectivity of neuron n_j at layer $l-1$ to the neuron n_i at layer l. $b_{n_i}^{l-1}$ represents the bias of the previous layer that belongs to the neuron n_i. These connectivity are summed based on the total number of neurons (N) at layer $l-1$. Finally, an activation function ($a_{n_i}^{l}$) is applied to the summation to induce non-linearity to the neuron ($X_{n_i}^{l}$).X_{n_i} at the output layer is equivalent to an atomic energy, and it represents an atom-centered descriptor at the input layer. The collection of atomic energy contributions are summed to obtain the total energy of the structure. At the end, the total energy, forces, and stress tensors are compared to the reference values (see Eq. 2.74). This process is called forward propagation.

Similar to the linear regression, one needs to obtain a set of weight parameters to minimize the loss function. In NN architecture, the gradient of loss with respect to the weight parameters can be conveniently done by the backpropagation algorithm [107]. The backpropagation algorithm technique demands an $\mathcal{O}(W)$ computational cost, proportional to the number of weight parameters [81].

A number of optimization algorithms can be applied here to update the weights iteratively, until the optimal solution is found. Limited Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm [108] and Adaptive Moment Estimation (ADAM) [109] will be used for most of our applications,. In ADAM, mini-batch optimization method is usually performed by randomly selecting a subset of the entire dataset for each optimization step. Since one can use mini-batch optimization for neural networks regression, neural networks algorithm is ideal for training huge dataset.

2.5.4 Gaussian Process Regression

Gaussian Process Regression (GPR) model is categorized in the domain of Bayesian statistics. The Bayesian uses probability to reflect degrees of certainty in the states of knowledge. It estimates all of the predictions as a random variable account of it being a function of the dataset. Typically, Bayesian models generalize much better when limited dataset is available. It tends to protect against overfitting by dealing it with integrating over the uncertainty in the estimator. However, it suffers from high computational cost as the dataset gets larger [110].

Here, the Bayesian approach will be utilized to learn linear regression model. In particular, the energy of an atom is assumed to be mapped linearly from atom-centered descriptors ($\boldsymbol{X}_i$) (see Eq. 2.51)

$$E_i = \boldsymbol{w}\boldsymbol{X}_i \tag{2.80}$$

As the name GPR is implied, the prior probability distribution of the weights ($\boldsymbol{w}$) is initiated to be Gaussian distribution with zero mean and covariance $\boldsymbol{C}$:

$$P(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \boldsymbol{C}) \tag{2.81}$$

Due to one of the property of Gaussian distribution, the energy is also distributed in Gaussian

statistics. Hence, the covariance of two atomic energies can be written as

$$\begin{aligned}\langle E_i E_j \rangle &= \langle E_i E_j \rangle = \langle \boldsymbol{w}\boldsymbol{X}_i \boldsymbol{w}'\boldsymbol{X}_j \rangle \\ &= \sigma_w^2 \langle \boldsymbol{X}_i \boldsymbol{X}_j \rangle\end{aligned} \tag{2.82}$$

In the equation, the relation of $\langle w_h w'_{h'} \rangle = \delta_{hh'}\sigma_w^2$ is exploited, where σ_w is a scalar. Since the atomic energies are unavailable in practical quantum calculations such as DFT method, one can only predict the total energy:

$$\langle E_N E_M \rangle = \sigma_w^2 \sum_{i \in N} \sum_{j \in M} \langle \boldsymbol{X}_i \boldsymbol{X}_j \rangle \tag{2.83}$$

where i and j are the indices of atoms for structures N and M, respectively. Now, the challenge is to determine the relationship between the two atom-centered descriptors, i.e. $\langle \boldsymbol{X}_i \boldsymbol{X}_j \rangle$.

A kernel function can be used to determine $\langle \boldsymbol{X}_i \boldsymbol{X}_j \rangle$. Kernel functions are to be understood as a similarity measure between two atomic neighbor environments. General rules in choosing a kernel function are discussed in detail in the literature [111]. In this study, a similarity kernel function can be written as

$$K(\boldsymbol{X}_i, \boldsymbol{X}_j) = exp\left(\frac{-(1 - d(\boldsymbol{X}_i, \boldsymbol{X}_j)^2)}{2l^2}\right) \tag{2.84}$$

where,

$$d(\boldsymbol{X}_j, \boldsymbol{X}_j) = \frac{\boldsymbol{X}_i \cdot \boldsymbol{X}_j}{|\boldsymbol{X}_i||\boldsymbol{X}_j|} \tag{2.85}$$

Hence, Eq. 2.83 can be written as for simplicity

$$\begin{aligned}C_{EE} &= \langle E_N E_M \rangle \\ &= \sigma_w^2 \sum_{i \in N} \sum_{j \in M} K(\boldsymbol{X}_i, \boldsymbol{X}_j)\end{aligned} \tag{2.86}$$

In the MLP development, one can also consider the derivative of total quantum mechanical

energy with respect to the relative positions of atoms (see Eq. 2.52) in the training process. This can be achieve by differentiating the Eq. 2.86, one obtains the covariance between the force and energy relation.

$$\begin{aligned}\boldsymbol{C}_{FE} &= \left\langle \frac{\partial E_N}{\partial \boldsymbol{r}_n} E_M \right\rangle \\ &= \frac{\partial \langle E_N E_M \rangle}{\partial \boldsymbol{r}_n} \\ &= \sigma_w^2 \sum_{i \in N} \sum_{j \in M} \frac{\partial K(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\partial \boldsymbol{r}_n} \\ &= \sigma_w^2 \sum_{i \in N} \sum_{j \in M} \frac{\partial K(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\partial \boldsymbol{X}_i} \frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_n}\end{aligned} \tag{2.87}$$

where $\boldsymbol{r}_n$ is the x, y, and z positions of n-th atom. Here, $\boldsymbol{C}_{FE}$ is an array of three components. $\frac{\partial \boldsymbol{X}_i}{\partial \boldsymbol{r}_n}$ becomes zeros as the pair distance between $\boldsymbol{r}_i$ and $\boldsymbol{r}_n$ is beyond the cutoff radius (see Eq. 2.54). Due to the symmetry of the kernel function (see Eq. 2.84 and 2.85), the covariance between the energy and force can be written as the transpose of the the covariance between the force and energy such that

$$\boldsymbol{C}_{FE} = \boldsymbol{C}_{EF}^T \tag{2.88}$$

Similarly, the derivative between two forces is

$$\begin{aligned}\boldsymbol{C}_{FF} &= \left\langle \frac{\partial E_N}{\partial \boldsymbol{r}_n} \frac{\partial E_M}{\partial \boldsymbol{r}_m} \right\rangle \\ &= \frac{\partial^2 \langle E_N E_M \rangle}{\partial \boldsymbol{r}_m \partial \boldsymbol{r}_n} \\ &= \sigma_w^2 \sum_{i \in N} \sum_{j \in M} \frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_n} \frac{\partial^2 K(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\partial \boldsymbol{X}_i \partial \boldsymbol{X}_j} \frac{\partial \boldsymbol{X}_j}{\partial \boldsymbol{r}_m}\end{aligned} \tag{2.89}$$

Furthermore, one can include the training of stress by including the covariance of stress-energy,

stress-force, and stress-stress relations such that

$$
\begin{aligned}
\boldsymbol{C}_{SE} &= \left\langle \sum_n \boldsymbol{r}_n \otimes \frac{\partial E_N}{\partial \boldsymbol{r}_n} E_M \right\rangle = \sigma_w^2 \sum_{i\in N}\sum_{j\in M} \left(\sum_n \boldsymbol{r}_n \otimes \frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_n} \right) \frac{\partial K(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\partial \boldsymbol{X}_i} \\
\boldsymbol{C}_{SF} &= \left\langle \sum_n \boldsymbol{r}_n \otimes \frac{\partial E_N}{\partial \boldsymbol{r}_n} \frac{\partial E_M}{\partial \boldsymbol{r}_m} \right\rangle = \sigma_w^2 \sum_{i\in N}\sum_{j\in M} \left(\sum_n \boldsymbol{r}_n \otimes \frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_n} \right) \frac{\partial^2 K(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\partial \boldsymbol{X}_i \partial \boldsymbol{X}_j} \frac{\partial \boldsymbol{X}_j}{\partial \boldsymbol{r}_m} \\
\boldsymbol{C}_{SS} &= \left\langle \sum_n \boldsymbol{r}_n \otimes \frac{\partial E_N}{\partial \boldsymbol{r}_n} \sum_m \boldsymbol{r}_m \otimes \frac{\partial E_M}{\partial \boldsymbol{r}_m} \right\rangle \\
&= \sigma_w^2 \sum_{i\in N}\sum_{j\in M} \left(\sum_n \boldsymbol{r}_n \otimes \frac{\partial \boldsymbol{X}_i^T}{\partial \boldsymbol{r}_n} \right) \frac{\partial^2 K}{\partial \boldsymbol{X}_i \partial \boldsymbol{X}_j} \left(\sum_m \boldsymbol{r}_m \otimes \frac{\partial \boldsymbol{X}_j}{\partial \boldsymbol{r}_m} \right)
\end{aligned}
\tag{2.90}
$$

where $\otimes$ is the cross product and $\boldsymbol{r}_m$ is the x, y, and z positions of the m-th atom. The K_{SE}, K_{SF}, and K_{SS} are vector with 6 components, 6x3 matrix, and 3x3 matrix, respectively. In stress tensor, there are supposed to be 9 components to be considered such that xx, xy, xz, yx, yy, yz, zx, zy, and zz. Due to the rotational invariant of the atom-centered descriptor, there are only 6 components to be considered: xx, yy, zz, xy, xz, and yz. This also helps for the effort of computational efficiency. Moreover, the energy-stress and force-stress relations can be found to inherit the symmetrical behavior, i.e. $K_{ES} = K_{SE}^T$ and $K_{FS} = K_{SF}^T$, respectively.

Finally, the covariance matrix ($\boldsymbol{C}$) can be written as follow

$$
\boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_{EE} & \eta_f \boldsymbol{C}_{EF} & \eta_s \boldsymbol{C}_{ES} \\ \eta_f \boldsymbol{C}_{FE} & \eta_f^2 \boldsymbol{C}_{FF} & \eta_f \eta_s \boldsymbol{C}_{FS} \\ \eta_s \boldsymbol{C}_{SE} & \eta_f \eta_s \boldsymbol{C}_{SF} & \eta_s^2 \boldsymbol{C}_{SS} \end{pmatrix} \tag{2.91}
$$

The covariance matrix above are the measure of signals between multiple configurations. It should be noted that the $\boldsymbol{C}_{EE}$ is now a matrix of NxN, where N is the number of configurations in a training dataset. Assuming there are M atoms for every structure, the $\boldsymbol{C}_{FF}$ is a $3NM$x$3NM$ matrix. The $\boldsymbol{C}_{SS}$ is a $6N$x$6N$ matrix. η_f and η_s are hyperparameters to emphasize the importance of force and stress contributions, respectively, with respect to the energy.

Normally, a hyperparameter called noise (ν) is introduced to the diagonal elements of the

covariance matrix to ensure positive definiteness such that $\boldsymbol{C} = \boldsymbol{C} + \nu^{-1}\delta nm$. This is sometimes called noisy covariance. The noise ν is typically in the order of 10^6. Decreasing ν can increase positive definiteness in general.

In model above, the covariance contains two hyperparameters: σ_w and l. There are several tensorial choices of l according to Ramussen et. al. [111]. Here, we choose l such that it is a scalar. The hyperparameter l is also called the characteristic length-scale. It apprises the deviation between two input variable. The two hyperparameters will be bundled as $\boldsymbol{\theta}$ for simplicity. To optimize the two hyperparameters, the *log marginal likelihood* is applied as the measure of performance or loss function:

$$logP(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{t}^T\boldsymbol{C}^{-1}\boldsymbol{t} - \frac{1}{2}log|\boldsymbol{C}| - \frac{n}{2}log(2\pi) \tag{2.92}$$

where $\boldsymbol{t}$ is the target values. Target comprises of structural energies and may include force and stress information. The consequence of the target only appears in the first time, which means the data fitting is captured in the first term. The second term is the complexity penalty. Lastly, the final term is the normalization constant, where n is the total number of target values. Obviously, the hyperparameters are found by minimizing the log marginal likelihood with respect to $\boldsymbol{\theta}$:

$$\frac{\partial}{\partial\boldsymbol{\theta}}logP(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{\theta}) = \frac{1}{2}tr\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^T - \boldsymbol{C}^{-1})\frac{\partial\boldsymbol{C}}{\partial\boldsymbol{\theta}}\right) \tag{2.93}$$

where $\boldsymbol{\alpha} = \boldsymbol{C}^{-1}\boldsymbol{t}$. The computational complexity is dominated by the demanding inverse of the covariance matrix. Standard methods for inverting positive definite symmetric matrix will cost $\mathcal{O}(n^3)$, while the computation of the derivatives costs $\mathcal{O}(n^2)$ after $\boldsymbol{C}^{-1}$ is known. Nevertheless, the $\mathcal{O}(n^3)$ computational cost can be reduced to $\mathcal{O}(nk^2)$ by employing Cholesky decomposition, where k is less than n.

Cholesky decomposition can be thought of as the square root of amatrix such that $\boldsymbol{C} = \boldsymbol{L}\boldsymbol{L}^T$, where $\boldsymbol{L}$ is the lower triangular matrix. Using Cholesky decomposition, one can evaluate the first term in the log marginal likelihood by reconstructing of $\boldsymbol{Z}^T\boldsymbol{Z}$, where $\boldsymbol{Z} = \boldsymbol{L}^T\boldsymbol{t}$. The log

determinant or the second term becomes the sum of the diagonal element of the Cholesky matrix such that $log|\boldsymbol{C}| = \sum_i^n log(l_{ii})$, where l_{ii} is the diagonal of $\boldsymbol{L}$. Finally, the linear equation in Eq. 2.80 can be solved by employing the Cholesky matrix in the form of

$$\boldsymbol{t} = \boldsymbol{\alpha}\boldsymbol{L} \tag{2.94}$$

where α is the coefficients of training data points in kernel space. The number of coefficients equals to the total number of data points in the training dataset.

Finally, prediction of the energy, force, and stress can be determined once the α is solved. Suppose that, there is a new input of a structure represented as $\boldsymbol{X}_i^{new}$, $\frac{\partial \boldsymbol{X}_i^{new}}{\partial \boldsymbol{r}}$, and $\sum_k \boldsymbol{r}_k \otimes \frac{\partial \boldsymbol{X}_i^{new}}{\partial \boldsymbol{r}_k}$. The new covariance matrix needs to be constructed such that the kernels between the training dataset and the new data points have to be computed. For instance, the energy prediction is

$$\hat{E} = \boldsymbol{\alpha} \cdot (\boldsymbol{C}_{E^{train}E^{new}} \boldsymbol{C}_{F^{train}E^{new}} \boldsymbol{C}_{S^{train}E^{new}}) \tag{2.95}$$

The first index in the indices of $\boldsymbol{C}$ is of all of the training dataset, while the second index represents the new data point. The force and stress predictions will yield

$$\begin{aligned} \hat{\boldsymbol{F}} &= \boldsymbol{\alpha} \cdot (\boldsymbol{C}_{E^{train}F^{new}} \boldsymbol{C}_{F^{train}F^{new}} \boldsymbol{C}_{S^{train}F^{new}}) \\ \hat{\boldsymbol{S}} &= \boldsymbol{\alpha} \cdot (\boldsymbol{C}_{E^{train}S^{new}} \boldsymbol{C}_{F^{train}S^{new}} \boldsymbol{C}_{S^{train}S^{new}}) \end{aligned} \tag{2.96}$$

In conclusion, GPR formalism for MLP development has been presented in details. For a linear model, the weights are assumed to be Gaussian distributed. Due to the property of Gaussian distribution, the linear model can be considered to be Gaussian as well. In order to determine the multi-dimensional Gaussian distribution, the mean and covariance matrix have to be determined. The determination of covariance matrix with kernel has been shown in details. In the end, one can predict the energy, force, and stress from solving the covariance matrix and determining the $\boldsymbol{\alpha}$ coefficients of training data points in the kernel space. The coefficients $\boldsymbol{\alpha}$ can be viewed as

a function of covariance matrix. Therefore, the construction of the covariance matrix is the vital component in GPR model.

2.6 Comparison between Neural Networks and Gaussian Process Regressions

One of the advantage of using GPR is that overfitting is not necessarily a problem as the models get large [112], since GPR is based on Bayesian formalism. On the other hand, neural networks are known to suffer from overfitting. Nonetheless, overfitting can be rectify by adding penalty term (see Eq. 2.74) in the loss function. Second, GPR captures model uncertainty. For example, GPR gives the mean and the distribution for the prediction value via standard deviation, rather than just one value. Whereas, the uncertainty is not directly available in neural network regression. By knowing the uncertainty in the prediction, one can further improve the accuracy of GPR model by including the predictions with high uncertainty in the training process.

The disadvantage of GPR model is that the algorithm is not built for training large dataset. The inversion of covariance matrix can be extremely inconvenient as the training size gets large. More importantly, the prediction depends on the training dataset as shown in Eq. 2.95 and 2.96. It is not only the training can be computationally intractable, but also the prediction can be computationally expensive. On the other hand, neural networks model is great for developing MLP with large dataset size. As long as neural networks model has the optimal weight parameters, the prediction do not scale as a function of training dataset.

Chapter 3 PyXtal_FF Package

In this chapter, the development of PyXtal_FF package will be introduced. The aim of PyXtal_FF is to promote the application of atomistic simulations with MLPs by providing several choices of atom-centered descriptors and machine learning regressions in one platform. PyXtal_FF is an open source package, written in Python, for developing MLPs, in particular NNPs and GLPs [113]. Since it is written in Python, this platform is constructed in modular approach, enabling convenient effort to add new functionality. It provides unified interfaces to train the MLPs and perform atomistic simulations. PyXtal_FF package, along with the documentation, is publicly available at https://pyxtal-ff.readthedocs.io.

3.1 Capabilities of PyXtal_FF

Presently, the package is equipped with two regression models and four types atom-centered descriptor, as explained in Chapter 2. As mentioned, these regression models and atom-centered descriptors are easily extendable without changing the core user-interface features. Figure 3.1 represents the workflow of PyXtal_FF.

Dataset. The dataset is comprised of a set of atomic configurations along with the energy, forces, and stress, typically obtained from the first-principle calculations. An atomic configuration data should include the lattice matrix of the configuration cell, atomic species, and the coordinates of all atoms. Although the reference output values such as energy, forces, and stress are usually presented in the unit of eV, eV/Å, and GPa as the default unit in PyXtal_FF, they can be easily converted. PyXtal_FF utilizes the Atomic Simulation Environment (ASE) package [114] to parse and store the DFT data assembled in several formats, including but not limited to, ASE database, JSON, extended XYZ and the VASP OUTCAR formats. Furthermore, ASE is employed to compile the atomic neighborhood of each atom in the unit cell based on the periodic boundary conditions within a cutoff radius. After the neighboring data are gathered, it will compute the user-defined

type of descriptor.

Atom-centered descriptors. Users have five types of atom-center descriptors to choose from: ACSF, wACSF, EAD, SO(4) Bispectrum, and SO(3) Power Spectrum. The computation of the atom-centered descriptors follows the theory described in Section 2.5.3 utilizing NumPy—a Python library for scientific computing [115]. Since SO(3) Power Spectrum and SO(4) Bispectrum require heavy computations, the implementation of SO(3) Power Spectrum and SO(4) Bispectrum is wrapped in Numba—a just-in-time compiler that translates a subset of Python/NumPy code into fast machine code [116]. The computations can approach the speed of C or FORTRAN languages with Numba decorators added to descriptor calculator. For every structure, the descriptor calculator will return the descriptors and derivatives related to the force and stress, as seen in Eq. 2.52 and Eq. 2.53, respectively. Here, the force and stress descriptors are 4-D and 5-D arrays. The first and the second, and third dimensions represent the i-th center atom and the j-th pair neighboring atom, and the k-th descriptor. For stress descriptors, the second dimension is usually compressed, since atomic stress is not required in MLP development. The forth dimension of force descriptors describes the direction of forces, whereas the forth and fifth dimension for stress descriptor incorporates the stress tensor. In addition, the calculations of descriptors along with the derivative components can be computed in parallel using multiple cores. Due to a large number of data, a common approach is storing and reading the descriptors in disk. After the computation of descriptors is finished, they will go through a data parser along with the DFT energy, forces, and stress. The data parser will ensure the descriptors are coupled with the correct DFT data. In neural network, the data parser includes normalization for the calculated descriptors, whereas the first dimension of the force descriptors is compressed for linear regression. Eventually, the normalized descriptors represent the independent variables in the models to obtain the predicted energy and the derivative terms are needed to compute the force and stress values.

Regression Models. The Pyxtal_FF supports two models, linear/quadratic regression and neural networks. The neural network regression is powered by PyTorch [117]—an open-source deep learning framework centered on automatic differentiation [118]. Automatic differentiation if Py-

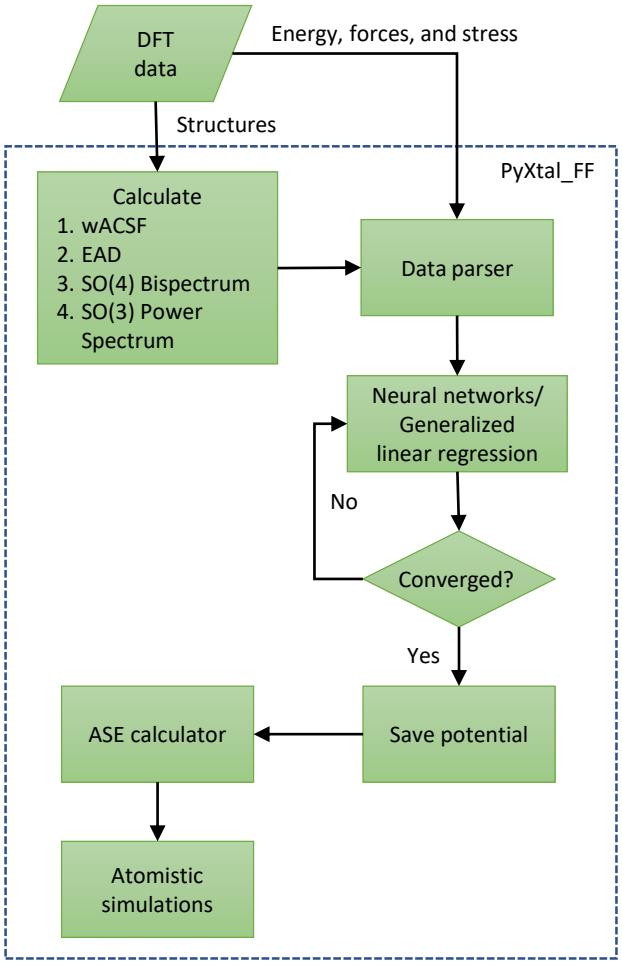


Figure 3.1: Schematic diagram of PyXtal_FF workflow for NNP/GLP training.

Torch records operation history (i.e. addition or multiplication) by constructing computational graph and defines formulas for differentiating operations. When computing the derivatives, the graph is processed in the topological ordering. Meanwhile, PyXtal_FF supports three iterative optimization algorithms for training or weights optimization: Limited Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) [108], Adaptive Moment Estimation (Adam) [109], and stochastic gradient descent (SGD) with momentum [119]. PyTorch uses Kaiming method [120] as the default to generate the initial guesses for the weights. The recommended option for optimizing the weights is the L-BFGS method with approximated line search as the training data is relatively small. This is due to the quasi-Newton method is generally more stable and can find local optima more efficiently.

With larger training datasets, however, the L-BFGS method is memory demanding, and one can seek to use first-order methods such as Adam or SGD with momentum prior to applying L-BFGS method. Both SGD and Adam optimizers are usually done in mini-batches, where the gradients for each weight update are calculated based on a subset of the entire training data set. Training in mini batches can reduce the variance of the parameter updates leading to stable convergence. The training is terminated after the convergence criteria is reached, i.e. maximum allowed minimization step or the loss value (see Eq. 2.74 have reached a negligible improvement. In addition, the training can also be done in graphical processing units (GPU) mode. After the training is done, the model will be saved in a PyTorch binary file, usually in .pth extension.

Atomistic Simulations. In addition to the force field generation, PyXtal_FF also provides the supports to utilize the trained models for several types of atomistic simulations, including geometry optimization, MD simulation, physical properties prediction, nudge elastic band simulation, and phonon calculation. These features are managed by ASE calculator, in which the MLP potential generated by PyXtal_FF passes the energy, forces, and stress tensors to the calculator. Then, ASE can perform the relevant atomistic simulations. Since these simulation will be powered by Python, we recommend to use them only for light-weight simulations. In the near future, we will be interfacing the trained MLP with LAMMPS [121] to enable the large-scale atomistic simulations.

3.2 Example Usage of PyXtal_FF

PyXtal_FF can be used as stand-alone library in Python scripts. A PyXtal_FF example code to train an MLP for elemental platinum (Pt) is shown in the Listing 3.1. The atom-centered descriptors and the model are described in dictionary. The dictionary keys determine the necessary command for the code and are made as intuitive as possible. In this case, the SO4 Bispectrum is used as the descriptors. For the descriptors, most of keys follow the hyperparameters naming as mentioned in the Section 2.5.3. The cutoff radius (R_c) is 4.9 Å, and the bispectrum components will be computed up to l_{max} =3.

```
from pyxtal_ff import PyXtal_FF

# Define the path of train/test data.
train, test = 'train.json', 'test.json'

# Define the descriptor.
descriptor = {'type': 'Bispectrum', 'Rc': 4.9
              'parameters': {'lmax': 3}}

# Define the neural network regression model.
model = {'system' : ['Pt'],
         'hiddenlayers': [16, 16],
         'epoch': 1000,
         'path': 'Pt-Bispectrum/'
         'optimizer': {'method': 'lbfgs'}}

# Define the descriptor and model to PyXtal_FF and run the training.
mlp = PyXtal_FF(descriptor, model)
mlp.run(TrainData=train, TestData=test)
```

Listing 3.1: PyXtal_FF script for force field training

By default, PyXtal_FF will use neural networks as the regression algorithm. Here, PyXtal_FF will look for *train.json* and *test.json* files to parse them in ASE format as described in Subsection 3.1. The optimizing algorithm L-BFGS will iteratively run for 1000 epoch. After the training is complete, the trained model is saved in the result directory—*Pt-Bispectrum*—with a name of *16-16-checkpoint.pth*, in which 16-16 denotes two hidden layers with 16 nodes for each layer. Although user can train their model in polynomial (linear or quadratic) regression as shown in Listing 3.2. Linear or quadratic regression is defined by the *order*: *1* for linear and *2* for quadratic.

For both polynomial and neural network regression, a regularization factor with Euclidean norm (*'norm': 2*) can be easily added to the loss function by enabling the *alpha* keyword.

```
model = {'system' : ['Pt'],
         'algorithm': 'PolynomialRegression',
         'order': 1,
         'alpha': 1e-4,
         'norm': 2,
         'path': 'Pt-Bispectrum/'}
```

Listing 3.2: An example to define linear regression model.

Additionally, PyXtal_FF provides a built-in interface with the ASE code, in which one can use the model to perform different types of calculations through ASE. A simple example to perform the geometry optimization on a Pt bulk crystal (*Pt_bulk.cif*) based on the trained model from the Listing 3.1 can be viewed in Listing 3.3. In addition to geometry optimization and MD simulation, PyXtal_FF also provides several utility functions to simulate the elastic and phonon properties, which are based on several external Python libraries including Phonopy [122], seekpath [123] and matscipy [124]. More detailed examples can be found in the online documentation https://pyxtal-ff.readthedocs.io.

```
from pyxtal_ff import PyXtal_FF
from pyxtal_ff.calculator import PyXtalFFCalculator, optimize
from ase.io import read

# Load the trained model.
mlp = "Pt-Bispectrum/16-16-checkpoint.pth"
ff = PyXtal_FF(model={'system': ["Pt"]}, logo=False)
ff.run(mode='predict', mliap=mlp)
calc = PyXtalFFCalculator(calc)

# Read the initial structure.
pt_bulk = read('Pt_bulk.cif')

# Perform the geometry relaxation.
pt_bulk.set_calculator(calc, box=True)
pt_bulk = optimize(pt_bulk)
print('energy: ', pt_bulk.get_potential_energy())
```

Listing 3.3: PyXtal_FF script to perform geometry optimization

Chapter 4 Applications

4.1 Binary System

The SiO_2 dataset [125] was generated by the DFT method within the framework of VASP [126], using the generalized gradient approximation Perdew-Burke-Ernzerhof (PBE) exchange-correlation functional [127]. The kinetic energy cutoff was set to 500 eV, and the energy convergence criterion for constructing the **k**-point mesh is within 10 meV/atom. The MD trajectories are taken at different temperatures including liquid, amorphous and crystalline (α-quartz, α-cristobalite, and tridymite) configurations. The original dataset contain 3,048 SiO_2 configurations (60 atoms per structure). For simplicity, a subset that consists of 1,316 structures are considered, with the goal of to gaining an overview of performances and computation costs for each descriptor. Below gives the parameters to define each descriptor.

```
# ACSF (70)
para = {'G2':
        {'eta': [0.003214, 0.035711,
                 0.071421, 0.124987,
                 0.214264, 0.357106,
                 0.714213, 1.428426],
         'Rs': [0]},
        'G4':
        {'lambda': [-1, 1],
          'zeta':[1, 2, 4],
          'eta': [0.000357, 0.028569, 0.089277]}}
descriptor = {'type': 'ACSF',
              'Rc': 4.9,
              'parameters': para,}
```

```
# wACSF (26)
descriptor = {'type': 'wACSF',
              'Rc': 4.9,
              'parameters': para,}

# EAD (30)
para = {'eta': [0.003214, 0.035711,
                0.071421, 0.124987, 0.214264],
        'Rs': [0, 1.50],
        'lmax': 2,}

descriptor = {'type': 'EAD',
              'Rc': 4.9,
              'parameters': para,}

# SO3 (40)
descriptor = {'type': 'SO3',
              'Rc': 4.9,
              'parameters': {'nmax': 4,
                             'lmax': 3},}

# SO4 (30)
descriptor = {'type': 'SO4',
              'Rc': 4.9,
              'parameters': {'lmax': 3},}
```

Listing 4.1: PyXtal_FF script to define the descriptors.

In short, a universal cutoff value of 4.9 Å is chosen for all descriptors. Each descriptors requires some manual selection of hyperparameters in the real (e.g., $\eta, \lambda, \zeta, R_s$) or integer ($l_{\max}$,

$n_{\max}$) space. The ACSF parameters were taken from Ref. [125] which lead to 70 descriptors. In its wACSF version, the number is reduced to 26. For EAD, a similar set of parameters for η and R_s is chosen, which make 30 descriptors when $L_{\max}$ = 2. For SO3 and SO4, only the integer type hypeparameters need to be provided. In this work, 40 SO3 descriptors with $n_{\max}$ = 4 and $l_{\max}$ =3 and 30 SO4 descriptors with $l_{\max}$ =4 are set. Table 4.1 summarizes the performances for neural networks and linear regressions.

First, the neural networks regression is used with two hidden layers with 30 nodes each. The training was conducted with 12000 steps. The ACSF-70 set yields the best accuracy in both energy (1.3 meV/atom) and forces (81.2 meV/Å), while the errors in its corresponding wACSF-26 set rise by 60-70% in both energy (2.1 meV/atom) and forces (141.8 meV/Å). On the other hand, the weighted EAD-30 descriptor, supposed to mimic G2 and G4 ACSFs, gives the highest errors (4.0 meV/atom for energy and 300 meV/Å for forces). This may be due to lack of optimization on the hyperparameters. However, it should be noted that the computation of EAD is much faster than ACSF. Therefore, it is worth exploring a systematic approach to obtain the optimum set for EAD. For the two spectral descriptors, SO3-40 seems to outperform SO4-30 while it cost about a similar level of CPU time. In terms of accuracy, SO3-40 (1.4 meV/atom in energy MAE and 115.1 meV/Å in force MAE) is in the middle of ACSF-70 and wACSF-26. Another remarkable advantage of the spectral descriptors is that tuning the hyperparameters is much easier. If one does not want to spend too much time on choosing the hyperparameters, SO3 seems to be a better choice than ACSF. It is important to note that all descriptor computations are based on Python. It is expected that the speed will be much faster when they are implemented in FORTRAN or C languages. In addition, the overall performances of neural networks are superior than those of linear regressions.

It is worth noting that the training of linear regression takes a fraction of the time it takes to train neural networks, despite the performances. The performances between neural networks and linear regression are expected due to neural networks are more flexible, meaning they carry more adjustable weight parameters. This can be further verify that ACSF-70 in both linear regression

Table 4.1: The MAE values of the predicted energy/forces of 1316 SiO_2 dataset from the neural networks, linear regression, and quadratic regression models with different descriptors. The neural networks model was performed with 2 hidden layers at 30 nodes for each layer within 12000 L-BFGS steps of training. The force is measured in meV/atom, and the energy is measured in eV/Å. For each type of descriptors, the average CPU time for descriptor computation per structure is also given.

	CPU time (secs/60 atoms)	Neural Networks	Linear Regression	Quadratic Regression
ACSF (70)	4.374	1.3/81.2	3.1/184.9	N/A
wACSF (26)	4.372	2.1/141.8	6.5/322.2	2.6/179.3
EAD (30)	0.584	4.8/259.0	12.9/553.3	6.2/360.4
SO3 (40)	1.028	1.4/115.1	7.5/359.6	1.9/187.9
SO4 (30)	1.078	3.3/204.2	12.8/540.0	6.1/364.2

and neural networks perform the best out of all descriptor types. Although SO4-30 performs slightly better than EAD-30, EAD-30 is a more suitable descriptors considering the computation of EAD is about 50% cheaper than SO4-30. An impressive improvement can be seen in SO3-40 with nearly 40% reduction in the MAEs for energy and forces. Nevertheless, wACSF-26 gives better performance than SO3-40. This behaviour can be due to the neural networks can learn the relationship between the descriptors of SO3-40 better than linear regression. For example, the first hidden layer contains information amongst the input layer, and the second layer contains in information amongst the first layer. As the layer propagation continues, the neural networks can learn the relationship between the neurons. On the other hand, linear regression has no capability of learning the relationship between descriptors. In consequence, quadratic regression can be used to learn the relationship between two descriptors.

As seen in Table 4.1, quadratic regression gives significant improvement on SO3-40 as well as wACSF-26. The energy MAE results between SO3-40 and wACSF-26 have the opposite effect as seen in linear regression. Meanwhile, the force MAE results between SO3-40 and wACSF-26 have almost negligible differences. This can be due to more correlation between three or more descriptors are needed as seen in the neural networks results. Moreover, SO4-30 and EAD-30 gain

significant improvement as the number of weight parameters increases. However, the computational cost in term of memory can hinder the popularity of quadratic regression since quadratic regression algorithm is not design to handle big data. Therefore, the quadratic regression for ACSF-70 is not explored, but it is expected to yield results in between those of neural networks and linear regression models, while close to the results of neural networks model.

4.2 Silicon MLP for General Purposes

In this section, the development of accurate and transferable MLP will be discussed. First, there are two types of datasets—a localized dataset and a diverse dataset—used in this study. Second, the validation on the machine learning framework with the localized dataset as the baseline will be inspected. Third, the interplay between bispectrum coefficient and the two machine learning regression (generalized linear regression and neural networks) on the localized dataset will be explored. The interplay is dedicated to further validate the localized dataset with a new NNP fitting strategy. Finally, a transferable silicon MLP based on the new strategy will be explored.

4.2.1 Datasets

In this study, two silicon datasets will be used. The Set #1 is the localized data set, obtained from Ref. [128]. Set #1 contains 244 structures in total, which includes the ground state of crystalline structure, strained structures, slabs, vacancy and liquid configurations from MD simulations. To generate the diverse data set, an in-house PyXtal code [129] is used to produce thousands of silicon structures with various numbers of atoms in the unit cell from 1, 2, 4, 6, 8 to 16. Random space group (1 to 230) assignment was applied to these silicon structures. For each random structure, four consecutive geometry optimization steps at the level of DFT with steady increase in precision were performed. The maximum numbers for each ionic step were 10, 25, 50 and 50. The relaxed images were then selected to our training pool to represent the shape of PES towards to the energy minima. With this scheme, it was ensured that not only the minima, but also the configurations around the minima would be captured during the energy fitting. Afterwards, single-point DFT

calculations were carried out for all configurations in the training pool. For each configuration, the total energy and forces were calculated at DFT level through the ASE package [114]. ASE provides interface to the VASP code [130] within projector augmented wave methodology [59] to perform geometry relaxations. In the calculation, the PBE-GGA [127] as the exchange-correlation functional with an energy cutoff of 600 eV and a Γ-centered KSPACING of 0.15 is used. Finally, 5352 silicon structures (Set #2) were selected by removing structures with energies that are higher than -4.000 eV/atom (i.e., 1.400 eV/atom higher than the ground states). In total, Set #1 has 15078 atoms, and Set #2 has 31004 atoms. It is important to note that the energy cutoff (600 eV) used in Set #2 DFT calculations is slightly higher than the one (520 eV) used in Ref. [128]. However, the differences resulted in negligible results according to the test for the same structures. Therefore, these two data sets can be used for direct comparison.

As shown in Fig. 4.1, Set #2 covers more diverse atomic environments in terms of energy, force, and density. Set #1 includes 244 structures that span from -4.560 to -5.425 eV/atom in energy, and 17.56 to 40.89 Å^3/atom in density. The energy of Set #2 ranges from -4.000 to -5.425 eV/atom, and the density ranges from 8.295 to 52.81 Å^3/atom. The force distribution in Set #2 is wider than that in Set #1. The principal component analysis (PCA) technique is utilized to further assess the similarity between the two datasets. The projection of two most dominating principal components are shown in Fig. 4.1. The principal components were fitted with the bispectrum coefficients mapped from the Set #2 structures. The inset shows that the data points of Set #1 cover mostly the empty space in the concentrated area. In other words, it appears to be that Set #1 and Set #2 rarely overlap. This indicates that two data sets encompass different atomic environments, which is expected since two different strategies were employed in generating the atomic configurations. Therefore, the two data sets are complementary and can be used to cross-validate each other in the MLP development for Si.

It is important to note that these two sets of data were obtained through entirely different approaches. Set #1 was not designed to generate an accurate force-field for Si, but rather to compare different MLPs on a small, standardized data set applicable to several elemental systems (for ex-

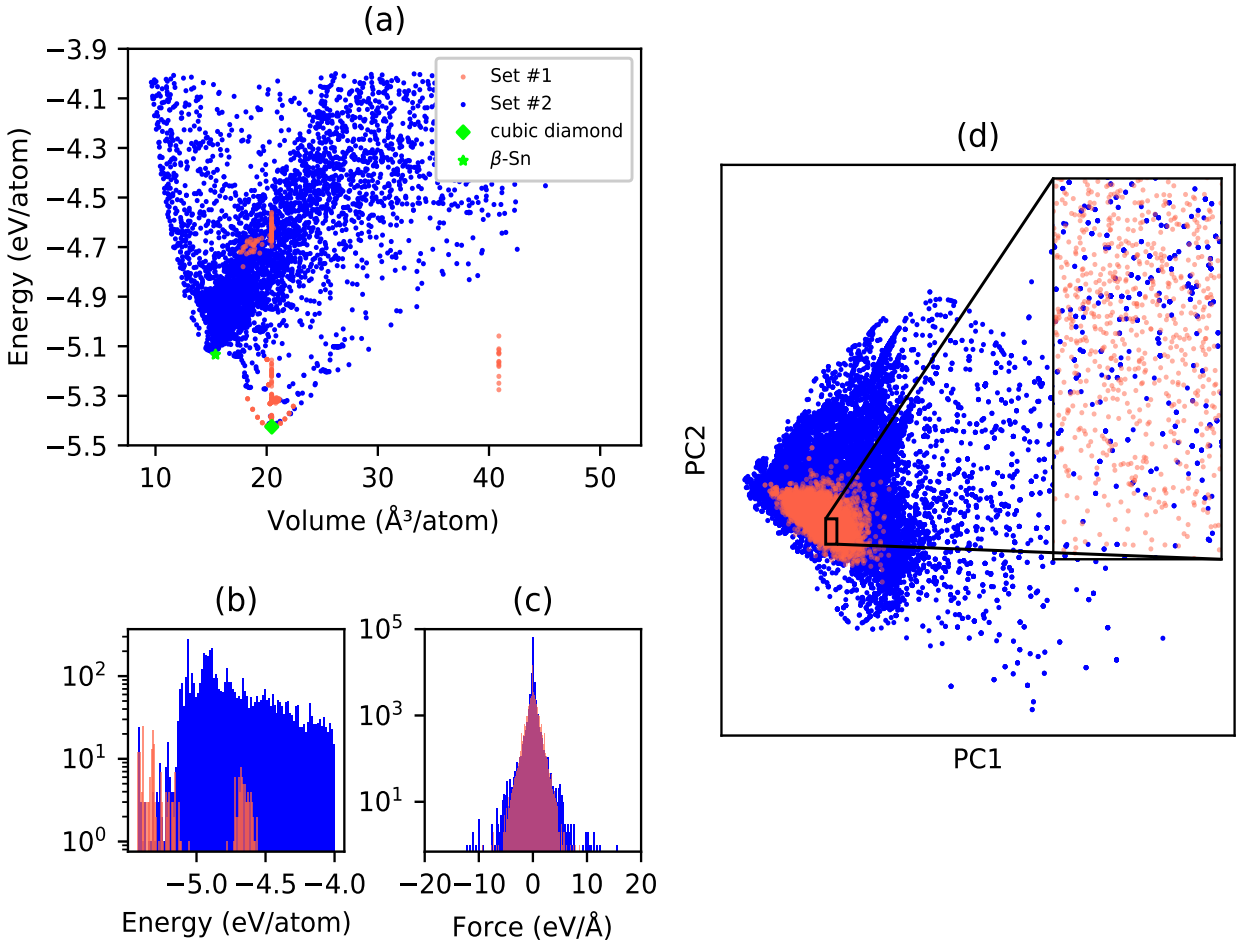


Figure 4.1: (a) The energy versus volume plot for training Set #1 and Set #2. The histograms of energy and forces are presented in (b) and (c), respectively. (d) The projection of two most dominating principal components of the atomic bispectrum coefficients. The inset illustrates a zoom-in view of the concentrated area. In the area, Set #1 is highly concentrated, whereas Set #2 is more widely spread.

ample only 20 snapshots from ab-initio MD were included into it). In a typical MLP development, a few thousand or more configurations will be needed for both Gaussian Process [131, 132] and neural networks regressions [133, 134]. Therefore, the training results from Set #1 are expected to gain some improvements by employing a larger version Set #1 with the same strategy (e.g., adding more MD snapshots). However, many other features in the PES will remain missing. Compared to Set #1, Set #2 covers more energy basins in the PES since it was obtained from an unbiased and more uniform sampling. For instance, it was found that Set #2 contains the high pressure β-Sn phase of silicon and many other phases with five- and six-coordinated silicon atoms. While such atomic environments can also exist in silicon grain boundaries and other types of defects generated from high temperate MD simulations, the MLP training may not describe these atomic energetics accurately when it attempts to fit the total energy of the system. Therefore, it is believed that a MLP with better coverage of the PES landmarks by small structures is more effective for an accurate modeling of rare events under various conditions (e.g. phase transitions, pronounced deformations, and chemical reactions). As it will be discussed in the next subsections, fitting on Set #2 is considerably more challenging than Set #1. While many relatively simple models data can yield satisfactory errors for Set #1, the overall accuracy for Set #2 is notably lower regardless the machine learning methods. Therefore, the goal of this work is to fit a Si MLP for general purposes which can describe Set #2 reasonably well while retaining a similar level of accuracy for Set #1. Prior to showing this result, the MLP implementation should be verify with Set #1 in the next subsection.

4.2.2 Verification with the Localized Data Set

In Ref. [128], the authors presented an extensive benchmark for silicon (as well as several other elemental systems) with different MLP approaches. This provided us a foundation to verify our MLP implementations by using their data for training and testing. With Set #1, it is necessary to reproduce the results based on the NNP, SNAP, and quadratic SNAP (qSNAP) methods. To compute the descriptors, the same parameter setting as reported in Ref. [128] was employed, which is

Table 4.2: The setting used to compute the atom-centered descriptors in this study. The ACSF descriptors are consistent with Ref. [128], except that R_c was set to 4.8 Å for the quadratic regression in the previous literature. Moreover, bispectrum coefficients with the band limit l_{max} up to 8 were considered. The asterisk symbol denotes the reduced parameter set for ACSF descriptors.

Descriptors	Parameters	Values
G^2	R_c (Å)	5.2
	R_s (Å)	0
	η (Å^{-2})	0.036*, 0.071*, 0.179*, 0.357*, 0.714*, 1.786*, 3.571, 7.142, 17.855
G^4	R_c (Å)	5.2
	λ (Å)	-1, 1
	ζ	1
	η (Å^{-2})	0.036*, 0.071*, 0.179*, 0.357*, 0.714, 1.786, 3.571, 7.142, 17.855
B	R_c (Å)	4.9
	l_{max}	2, 3, 4, 5, 6, 7, 8
	Normalization	True, False

summarized in Table 4.2. In the original literature, there were 9 G^2 and 18 G^4 descriptors. A further inspection was performed to reduced the number of hyperparameters, i.e. the η by constructing the histogram of the computed ACSFs of the entire Set #1. Apparently, the descriptors with large η values span in a very narrow range. Narrow-range descriptors are less likely to discriminate different local atomic environments, and they could introduce numerical noise. Therefore, the reduced parameter set, which included only 6 G^2 and 8 G^4 descriptors, are marked with asterisk symbol. For convenience, the 27 ACSFs descriptors are named as G27, and the reduced ACSFs descriptors are denoted as G14. For bispectrum coefficient, the expansion is limited to several finite orders, since the higher indices of l can only be beneficial in detecting subtle signals on the neighbor density map. In this study, only band limits (l_{max}) up to 8 are considered. However, the band limits of 3, 4, and 5 (30, 55, and 91 bispectrum coefficients) will be repeatedly used here. They are denoted as B30, B55, and B91. There are two additional cases with bispectrum as the descriptors: normalized and non-normalized. The normalized bispectrum are denoted as

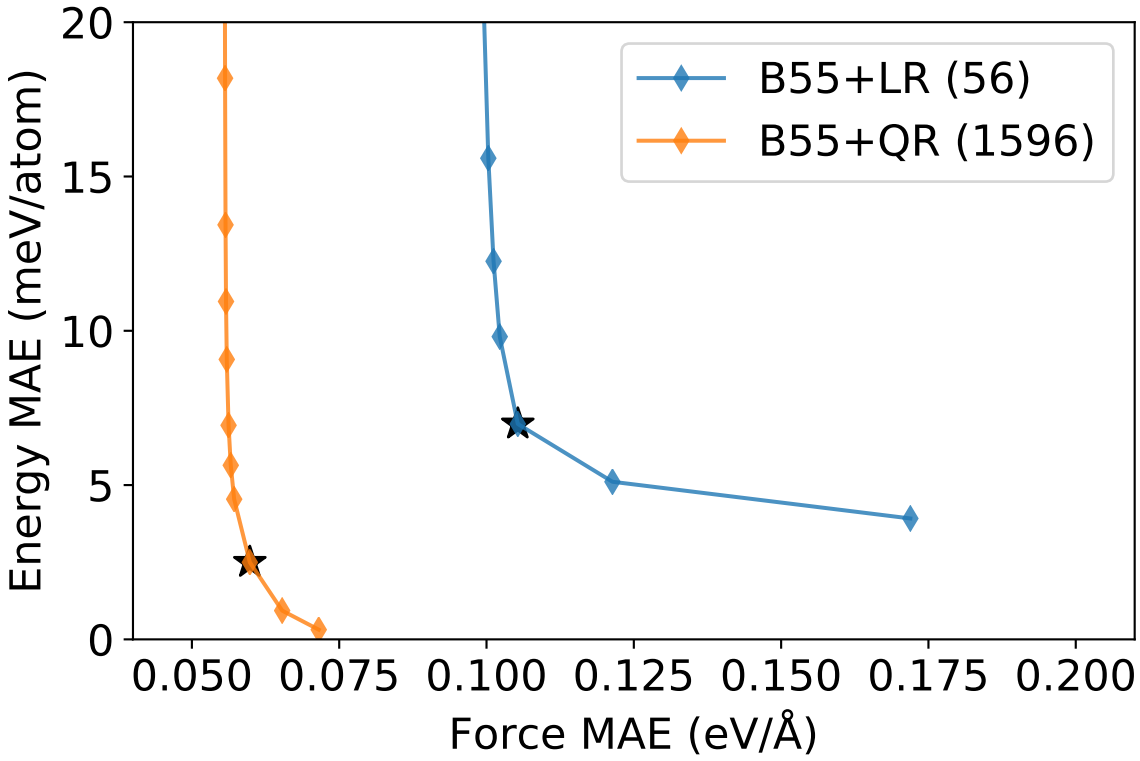


Figure 4.2: The comparison of fitting between linear and quadratic regression based on the B55 descriptors ($l_{\max} = 4$) applied to Set #1. For each regression, the energy MAE and force MAE values were collected by gradually varying the force coefficients from 1e-6 to 1. The numbers of weight parameters are given in the parentheses. The marked black asterisks correspond the results when the force coefficient is at 1e-4.

$\hat{\text{B}}$30, $\hat{\text{B}}$55, and $\hat{\text{B}}$91. Correspondingly, the labelings with the regression techniques are NNP+G27 for the neural networks regression with G27 descriptors, LR+B55 for linear regression with B55 descriptors, and QR+B55 for quadratic regression with B55 descriptors.

For the cases of linear and quadratic regressions, the results are deterministic as long as the force coefficient in Eq. 2.74 is given. Fig. 4.2 displays the gradual changes of mean absolute error (MAE) values for energy and forces by varying the force coefficient (β) from 1e-6 to 1e+0 for both LR+B55 and QR+B55. For each regression, these points seem to form a Pareto front. Namely, there is no single point which can beat the other points in both energy and force MAE values. Here, a range from the Pareto front which leads to an approximately even change on other sides is chosen. This point corresponds to the force coefficient closest to 1e-4. When β=1e-4, B55+LR yields the

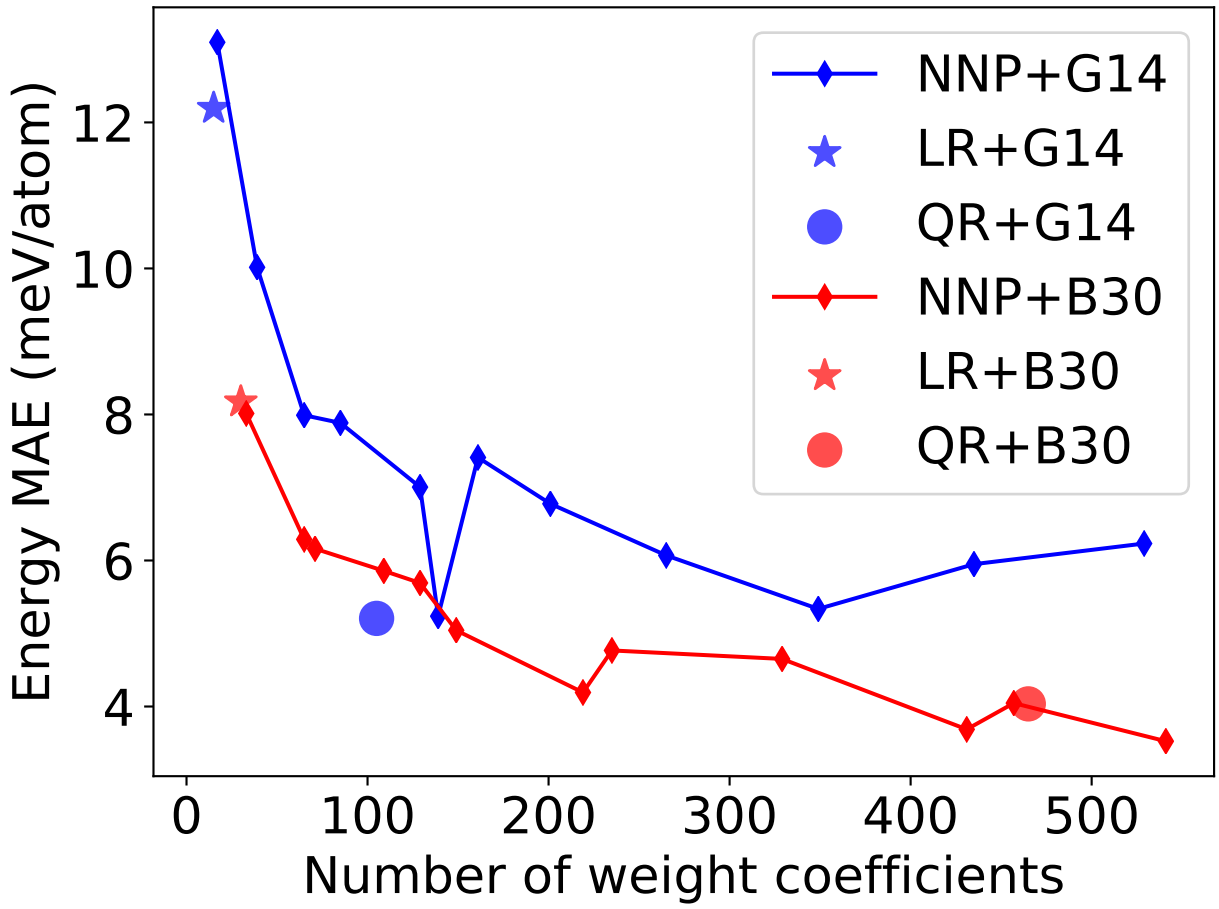


Figure 4.3: The performance of neural networks regression on G14 and B30 as a function of weight parameters. For comparison, the results from linear and quadratic regressions are also included.

MAE values of 6.94 (6.28) meV/atom for energy and 0.11 (0.12) eV/Å for force in training (test) data set. For B55+QR, the results gain significant improvement. The final energy MAE value is 2.50 (2.21) meV/atom, and the force MAE value is 0.06 (0.08) eV/Å. The results are expected since the quadratic form allows the coupling of bispectrum coefficients [37]. However, the number of weight parameters also increases notably from 56 to 1596, which increases the computational cost for both FF training and prediction.

For NNP+G27, the NNP fitting with neural networks architecture of 27-24-24 was tested. The predicted MAE values are 5.65 meV/atom in the training dataset and 5.60 meV/atom in the test dataset. The metrics are close to the previously reported values: 5.88 and 5.60 meV/atom in Ref.

Table 4.3: The comparison of mean absolute error (MAE) values between this work and Ref. [128] for the same 244 Si data set (Set #1). The results from Ref. [128] are shown in parentheses. For LR+B55 and QR+B55, the results are shown when force coefficient is at 1e-4. For the NNP fitting, neural networks architectures of 27-24-24 and 14-12-12 were used.

Fitting Method	Train Energy (meV/atom)	Test Energy (meV/atom)	Train Force (eV/Å)	Test Force (eV/Å)
LR+B55	6.94 (6.38)	6.28 (6.89)	0.11 (0.21)	0.12 (0.22)
QR+B55	2.50 (3.98)	2.21 (3.81)	0.06 (0.18)	0.08 (0.17)
NNP+G27	5.65 (5.88)	5.60 (5.60)	0.09 (0.12)	0.11 (0.11)
NNP+G14	5.95	6.33	0.10	0.11

[128]. Our force MAE values are 0.095 and 0.106 eV/Å, agreeing with the previous report as well. Furthermore, reduced ACSFs as the descriptor were employed to the NNP fitting (NNP+G14). It is found that the training with NNP+G14 also yielded comparable metrics. This indicated that the removed ACSFs descriptors were indeed redundant, and they can cause numerical noise during the NNP training. Correspondingly, an adjustment to training strategy was made to investigate the impacts of hyperparameters on the NNP training. In contrast to linear regression, the NNP training is much less vulnerable to the choice of force coefficient since the NNP can compromise for more flexible functional forms. It is rather reliant to the hidden layer size. Fig. 4.3 shows the energy MAE values scanning across the hidden layer sizes for NNP+G14 with β fixed at 0.03. Overall picture suggests that NNP performances tend to improve as the NNP model becomes more flexible. However, the NNP accuracy will saturate at some point. Beyond the saturation point, increasing the hidden layer size will only raise the computational cost and lower the chance of finding optimal weight parameters. The results from QR+B14 yields better performance than NNP with the same number of parameters. In principle, NNP should be able to self-learn a model similar to QR with the same number of weight parameters. However, different NNP trainings from different initial random guesses may yield somewhat less optimal solutions. This practice suggests that quadratic regression can be an alternative approach when the descriptor size is relatively small.

The results of verification with different training strategies are summarized in Table 4.3. Compared to Ref. [128], our results are close or maybe slightly better, especially in the force performances for generalized linear regression. Therefore, further investigations can be carried out on Set #1 by using different strategies.

4.2.3 Bispectrum Coefficients & Algorithms Interplay

In this section, MLP fitting with bispectrum coefficients will be discussed in details by using both generalized linear and neural networks regressions on Set #1. First, the performances of generalized linear regression can be improved based on the normalization factor of bispectrum coefficients prior to the MLP fitting. In the original implementation of SNAP [36], the bispectrum coefficients are not normalized prior to the MLP fitting. However, Fig. 4.4 shows the benefits of normalization prior to the MLP fitting. Linear regression achieves better performances for both energy and forces as l_{max} increases. At $l_{\text{max}} > 5$, there are no significant gains in the MAE values as the computational cost increases. The insignificance of normalization can be due to the limitation of linear regression ability to express the complexity.

Second, Fig. 4.3 shows the overall NNP fitting with bispectrum coefficients as the inputs to the neural network architecture. The results of NNP+B30 are trained with different hidden layer sizes. The best accuracy is achieved with the hidden layer size of [24, 24]. The 30-24-24 architecture consists of 1369 parameters in total. The training MAE values are 3.18 meV/atom and 0.07 eV/Å, and the test MAE values are 3.54 meV/atom and 0.08 eV/Å. These metrics reach comparable values to that from QR+B55 (see Table 4.3) with less bispectrum coefficients. For reference, linear regression and quadratic regression results with the corresponding number of bispectrum coefficients are also marked in Fig. 4.3. NNP with bispectrum coefficients can gain notable improvements in comparison to linear regression and quadratic regression. The improvements are expected since neural networks allows more flexible functional forms to describe the deviation from linearity. Meanwhile, quadratic regression achieves significant improvement in accuracy compared to linear regression due to the extended polynomial forms. However, similar accuracy can be attained with

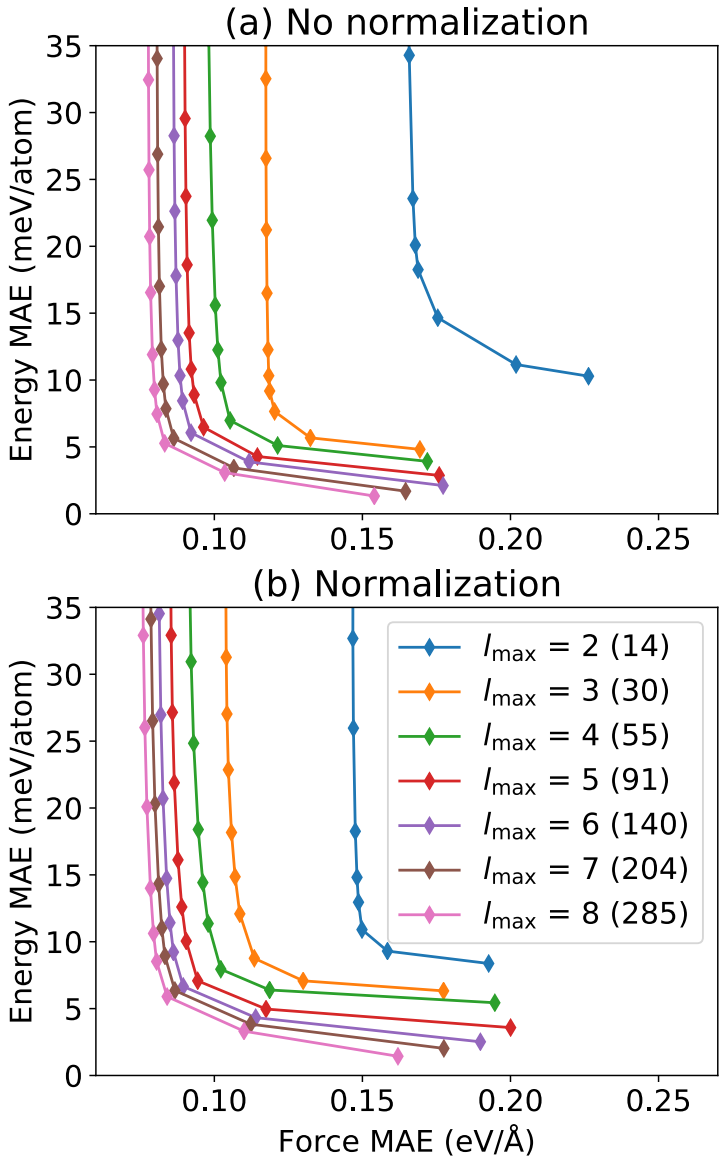


Figure 4.4: The performance of linear regression based on the bispectrum coefficients without (a) and with normalization (b). In each plot, $l_{\max}$ values from 2 to 8 were considered. The number of descriptors are given in the parenthesis.

Table 4.4: The MAE values of the predicted energy and forces of Set #2 by training on Set #1. The 30-10-10 is used as the neural networks architecture for providing comparable weight parameters as the quadratic regression. The numbers inside parentheses are the test MAEs.

	Neural networks	LR	QR
Energy (meV/atom)	4.7 (70)	7.5 (110)	4.0 (265)
Force (eV/Å)	0.08 (0.13)	0.12 (0.15)	0.08 (0.21)
Number of parameters	431	31	496

NNP fitting with smaller number of weight parameters.

4.2.4 Transferability of the MLP from a Localized Dataset

PyXtal_FF has the ability to apply various descriptors and regression techniques to train MLPs with satisfactory accuracy ($<$10 meV/atom in energy MAE and $<$0.15 eV/Å in force MAE) on Set #1. From computational perspective, bispectrum coefficients can cover more orthogonal sets and are easier to be expanded. Therefore, the bispectrum coefficients will be used as the main descriptors from now on. Using the MLP trained on Set #1, the prediction power on Set #2 (the more diverse data set) will be validated. The models include NNP with 30-10-10 architecture (431 parameters, with β at 0.03), linear regression (31 parameters), and quadratic regression (528 parameters). The three scenarios use normalized bispectrum coefficients with l_{max} of 3, as normalized bispectrum coefficients suggest slight accuracy improvement. Table 4.4 summarizes the results. In general, the prediction power of the MLP on Set #2, especially in energy, is still poor, though the force errors are acceptable. It is not surprising as the machine learning ability in extrapolation is known to be poor. The performance of the MLP yields great accuracy based on the given training data set. The characteristic of atomic environments of Set #2 is too broad and most of the data points lay outside of the Set #1. Therefore, the predicted energy and force are no longer reliable.

Despite the unsatisfactory accuracy, some insights can be gained from this numerical experiment. Neural networks regression can achieve better transferability in comparison to linear and

quadratic regression. Although the quadratic regression yields the best accuracy in training (3.99 meV/atom in energy and 0.08 eV/Å) in force, it also produces the largest error on the test set. On the contrary, neural networks regression achieves a similar level of accuracy on the training (4.70 meV/atom in energy and 0.08 eV/Å), but the errors on the test set (69.8 meV/atom energy MAE and 0.13 eV/Å force MAE) are much smaller. This can be partially explained by the fact that neural networks adopts more flexible functional forms during fitting.

4.2.5 Training with Data from Random Structure Generator

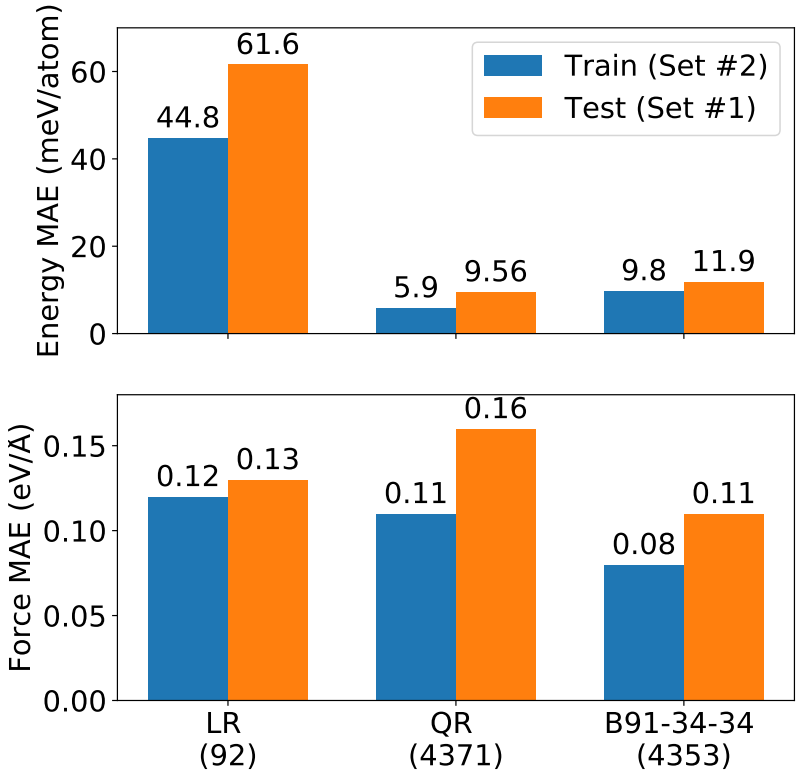


Figure 4.5: The performance of trained MLP on Set #2.

For the sake of data diversity, it is more natural to train the MLP based on Set #2, and test its performance on Set #1. To train reliable MLP on Set #2, more bispectrum coefficients are employed with a larger neural networks architecture. In addition, polynomial fittings are included for the purpose of comparison. In the end, the trained NNP will be tested on Set #1. For polynomial regression, l_{max} at 5 with cutoff radius of 4.9 Å was applied. According to Fig. 4.4, normalizing the bispectrum coefficients had negligible effect on the results. Hence, normalization was ignored. The β value was fixed at 1e-4 for quadratic regression, and 1e-3 for linear regression. The neural networks architecture of 91-34-34 was used to give a comparable weight parameters as the quadratic regression.

Fig. 4.5 summarizes the results of Set #2 training. In term of energy, quadratic regression performs the best accuracy (5.90 meV/atom), whereas NNP can predict less accurate energy (9.81 meV/atom) but better forces (0.08 eV/Å). It should be emphasized that Set #2 contains smaller unit cell (1-16 atoms in a unit cell) than Set #1 (up to 64 atoms in a unit cell). This transition from smaller to larger cells can introduce long-range effects that were not accounted for in the training [135]. Therefore, the MAE values on the test set are consistently larger than the training set. Furthermore, Set #1 may contain some manually-selected atomic configurations. These configurations may not be fully covered by our random generated structures. While linear regression predicts well on the forces, it guides the energy predictions to unsatisfactory results. This may due to the limitation of the regression technique as the smaller number of parameters fail to describe the true PES. Therefore, our recommendation is to use either quadratic regression (similar to the recently proposed qSNAP method [37]) or neural networks for a better fitting of a diverse dataset. Compared to the quadratic regression, neural networks is our preferred choice due to its flexibility.

In comparison to literature, this study yields comparable results as a previous study of diverse silicon cluster [136]. The authors juxtaposed among several atom-centered descriptors, including bispectrum coefficients, that were coupled with Gaussian process regression. In particular, the root mean square errors (RMSEs) for energy and forces with l_{max} at 5 are 20.2 meV/atom and 0.25 eV/Å. Meanwhile, the training RMSEs of our quadratic regression yield 9.7 meV/atom and

0.22 eV/Å and the training RMSEs of 91-34-34 architecture are 14.8 meV/atom and 0.16 eV/Å. In another study, Kuritz *et al.* focuses on training atomic forces using deep learning model with the environmental distances as the descriptors. The force predictions are performed at a scaling from 16 atoms to 128 atoms yields MAE of 0.12 eV/Å [135], given that the neural networks nodes are in the order of 10^3/layer. This phenomenon proves that the choice of descriptor can reduce the complexity of MLP.

Table 4.5: Comparison of different machine learning potentials of Si shown in RMSE values.

	QR	Neural networks	GPR[136]	DL[135]
Energy (meV/atom)	9.7	14.8	20.2	N/A
Force (eV/Å)	0.22	0.16	0.25	0.12

4.2.6 Physical Properties

One of the critical requirements for MLPs is to predict basic material properties, including but not limited to lattice parameter, elastic constants, and bulk moduli of diamond cubic Si. To obtain the elastic constants, the stress-strain relation are computed and fitted the relation to a set of linear equations built from the symmetry. For each applied deformation, the geometry of the structure are optimized to gain net force of zero. The summary of the properties is tabulated in Table 4.6.

First, it is crucial to validate on Set #1. On the column of Set #1 in Table 4.6, the performances of the MLPs are presented with different training strategies: energy-force linear regression (EF-LR), energy-force-stress linear regression (EFS-LR), energy-force quadratic regression (EF-QR), energy-force-stress quadratic regression (EFS-QR), energy-force neural networks (EF-NN), and energy-force-stress neural networks (EFS-NN). All of the training involved bispectrum coefficients as the descriptor. Linear and quadratic regression used bispectrum coefficients with l_{max} of 4, whereas neural networks used l_{max} of 3. Here, the neural network architecture of 30-10-10 was

used. Moreover, EF were trained with DFT energy and forces only as the reference values, while EFS included the DFT stress information in the training. Without stress involvement, the quadratic regression performances are the closest to the DFT values. Seemingly, linear and neural networks regressions fail to extrapolate the C_{12}. However, the C_{12} values tend to get closer to the DFT with tiny sacrifice in accuracy of C_{11}, when stress is involved.

Second, without stress information, linear and quadratic regression are considered to be more transferable in predicting the physical properties on Set #2. Evidently, linear regression gains no prominent refinement without trade-off between elastic constants as stress information is added. However, the values are the closest to the experimental values. On the other hand, quadratic regression exhibits accuracy boosts in C_{11} and the lattice constants in comparison to the DFT. As stress training is employed, NNP seems to benefit the most in term of transferability. Consequently, it is crucial to include stress tensors during the training of NNP.

4.2.7 Discussion

Training dataset. In general, MLP lacks extrapolative ability, unlike the traditional force field method. The training data set plays an extremely important role in MLP development. A more complete data set can grant the trained MLP with more powerful predictive ability. The use of randomly pre-symmeterized crystal structures is able to produce a data set with highly diverse atomic distribution [129, 132, 139]. In this work, the training dataset from crystal structure prediction techniques was prepared. However, several recent works demonstrated the MLP can be also trained on-the-fly [94, 132]. In addition to generating random structures, advanced sampling techniques such as metadynamics [35, 140] and stochastic surface walking [133] have also been used to provide the training data for MLP development. In general, each method focuses on different aspects of the PES. For instance, the surface walking method may work better in describing the transition path between different low energy configurations, while random structure generation offers more energy basins of the PES. On the other hand, metadynamics method excels at describing the liquid and amorphous states. At the moment, it remains challenging in obtaining a universal

Table 4.6: The experiment elastic constants [137] of cubic-diamond silicon are shown at zero-Kelvin values, while the DFT data are obtained from Ref. [128]. In comparison to the Gaussian approximation potential (GAP) of Si[138], the GAP results is shown below. The numbers of weight parameters are displayed in parentheses. EF and EFS stands for energy-force and energy-force-stress training. LR, QR, and neural networks are linear, quadratic, and neural network regressions, respectively. The neural networks architecture for Set #1 is 30-10-10 and 91-34-34 for Set #2.

				Set #1						Set #2					
	Exp	DFT	GAP	EF-LR (56)	EFS-LR (56)	EF-QR (1596)	EFS-QR (1596)	EF-NN (431)	EFS-NN (431)	EF-LR (91)	EFS-LR (91)	EF-QR (4371)	EFS-QR (4371)	EF-NN (4353)	EFS-NN (4353)
a(Å)	5.429	5.469	—	5.467	5.466	5.462	5.467	5.473	5.468	5.415	5.469	5.503	5.468	5.509	5.467
C_{11}(GPa)	167	156	153	153	151	149	152	157	154	137	167	173	158	167	153
C_{12}(GPa)	65	65	56	100	62	60	57	96	58	76	73	55	55	128	57
C_{44}(GPa)	81	76	72	69	70	75	75	66	68	73	85	81	71	43	76
B_{VRH}(GPa)	99	95	89	118	92	90	89	117	90	96	104	94	89	141	89

MLP to fully replace DFT simulation for general purpose [138]. Given the increasing power of regression techniques such as deep learning [103, 141], it will be interesting to know if a MLP can ultimately achieve the DFT accuracy by considering all training configurations from different sampling techniques.

Fitting scheme. Linear regression, as the simplest method in curve fitting, has been used in developing several MLPs [36, 39]. In particular, the MTP approach [128] can predict energy and forces with great accuracy while maintaining acceptable computational cost. The advantage of linear regression method lies in its simple algorithm which provides easy and fast computation. It should be emphasize that by applying normalization to the atom-centered descriptors can help improving the linear regression training. Despite the simplicity, linear/quadratic regressions are usually sensitive to the noise in the data set. In this work, neural networks regression is the focus, since it has more flexibility, which can yield better accuracy. Compared to the linear/quadratic regressions, including stress training in NNP is critical to promote the transferability. Beside neural networks, some non-parametric regression techniques, such as Gaussian Process Regression, have also been proved to be efficient in MLP development [41]. However, this is beyond the scope of the current study.

Applicability. For the purpose of MD simulation around the equilibrium state, fitting the MLFF with a localized data set generated from MD simulation is, perhaps, sufficient. However, the primary goal of this work is to generate high quality silicon MLP for a more general purpose, which requires a complete description of PES for a given chemical system. As discussed above, the MLP trained with Set #2 is generally capable of describing the entire PES of crystalline system better. We expect that the MLP generated in this work can be used to replace DFT simulation in predicting the structures of crystalline silicon, given that similar works have been done in several elemental systems [94, 132]. Yet, one needs to keep in mind that the quality still depends on the coverage of training data set. For instance, additional data is needed to enable predictions for surfaces and clusters [138], including liquid and amorphous configurations. Moreover, the trained MLP may not be able to describe the high energy configurations well, since Set #2 only contains

structures with energy less than 1.400 eV/atom from the ground state. It was found that some nonphysical configurations (e.g., short distances and overly clustered) may be favored under high temperature MD simulations. In this case, it is useful to either add a few explicit two-body and three-body terms to prevent the nonphysical configurations [131], or include some highly strained configuration in the training. Therefore, the the combination between the physical and machine learning terms in the training can be further prove and investigate for the applicability.

4.2.8 Summary

In conclusion, a systematic investigation of MLPs fitting for elemental silicon using PyXtal_FF has been presented. The silicon MLPs are developed by implementing different regression techniques based on ACSFs and bispectrum coefficients as the descriptors. The MLPs trained with Set #1 (the localized data set) can be described accurately in both energy and forces using generalized linear regression and neural networks based on both descriptor choices. Among the MLPs, fitting NNP with the bispectrum coefficients is the most favorable option. This is due to the expansion of bispectrum coefficients is more straightforward than ACSF descriptors. In addition, NNP provides more flexible framework in which the functional form can be easily adjusted by adding/reducing the size of weight parameters. For Set #2 generated from random symmetric structures, the NNP fitting with bispectrum coefficients achieves accuracy at 9.8 meV/atom for energy and 80 meV/Å for force, which is comparable to the current state of arts based on other approaches.

4.3 Ni-Mo

In this section, the computational costs will firstly be compare for bispectrum and power spectrum descriptors as a function of the hyperparameters.The accuracy of each representation in relation to both the number of descriptors and its computational cost will be then investigated by regressing on energies, forces, and stresses of a representative binary alloy Ni_3Mo/Ni_4Mo system using linear regression. Last, a more flexible neural network regression model will be introduced to improve the accuracy of fitting on the extended Ni-Mo dataset within a larger chemical space.

4.3.1 Dataset

In parallel to force field fitting, generating a diverse training data set is also a challenging task. Recently, there is an increasing trend for research groups to share their own data to the entire MLIAP community. Thanks to this trend, the examination of the dataset from a recent work by Li *et. al.* [90] will be conducted in this study.

There are 4019 atomic configurations for elemental Ni, elemental Mo, Ni_3Mo alloys, Ni_4Mo alloys, and doped Ni-Mo alloys. The dataset consists of (1) undistorted ground state structures for Ni, Mo, Ni_3Mo, and Ni_4Mo, (2) distorted structures obtained by applying strains of -10% to 10% at 1% intervals to a bulk supercell, (3) surface structures of elemental structures, (4) snapshots from ab initio molecular dynamics simulations of the bulk supercell at several temperatures, (5) doped alloy structures constructed by partial substitution of the bulk fcc Ni with Mo and the bulk bcc Mo with Ni. In addition, a dataset on Mo from Ref. [142] will be added to the Ni-Mo dataset. For the computation of each descriptor below, a uniform cutoff distance of 4.9 Åwas applied.

4.3.2 Computational Cost Comparison

This subsection will start with evaluating the computational cost of the SO(4) bispectrum components and the Smooth SO(3) power spectrum components, which requires some measure of the cost of each method. By far, the gradient is the most expensive part of the calculation so we estimate the cost of each method by the accumulation of the gradient for one neighbor. The cost function for each method is evaluated by the asymptotic cost of accumulating the gradient plus the cost of precomputing the expansion coefficients and their gradients for a given truncation.

For the SO(4) bispectrum components the cost of precomputation is equivalent to the number of Wigner-D matrix elements to evaluate, $\sum_{j=0}^{2j_{\max}}(j+1)^2$, where for the smooth SO(3) power spectrum the cost of precomputation is equal to the number of Wigner-D matrix elements to evaluate, $(l_{\max}+2)^2$, added to the number of radial functions to evaluate for the quadrature (Eq. 4.2), to compute each integral, we use $10(n+l+1)$ quadrature nodes. In the implementation, the cost of evaluating the radial functions is less than that of evaluating the D-functions. Nevertheless, the cost model

are treated as equal for the sake of simplicity.

$$\text{cost} = \text{cost}_{\text{accum}} + \text{cost}_{\text{precomputation}} \tag{4.1}$$

Therefore, the estimate the computational cost of each descriptor works as follows,

$$\begin{aligned} &\text{SO(4): } j_{\text{max}}^5 + \sum_{j=0}^{2j_{\text{max}}} (j+1)^2 \\ &\text{SO(3): } n_{\text{max}}^2 l_{\text{max}}^2 + \left[(l_{\text{max}}+2)^2 + \sum_{n=1}^{\text{n}_{\text{max}}} \sum_{l=0}^{\text{l}_{\text{max}}} 10(n+l+1) \right] \end{aligned} \tag{4.2}$$

The cost of each descriptor is then compared with the number of elements of that descriptor. The number of unique elements of each descriptor are given by:

$$\begin{aligned} N_{\text{SO(4)}} &= (j_{\text{max}}+1)(j_{\text{max}}+2)(j_{\text{max}}+3/2)/3 \\ N_{\text{SO(3)}} &= n_{\text{max}}(n_{\text{max}}+1)(l_{\text{max}}+1)/2 \end{aligned} \tag{4.3}$$

Fig. 4.6 is the plotted computational cost given by Eq. 4.2 with respect to the number of descriptors (Eq. 4.3) for both SO(4) bispectrum and SO(3) power spectrum.

As shown in Figure 4.6, the SO(4) bispectrum components are much less costly than the Smooth SO(3) power spectrum components in the low band limit ($N \leq 30$). At higher band limits, including more terms in the radial expansion of the smooth SO(3) power spectrum results in a less costly computation in comparison to the SO(4) bispectrum components.

4.3.3 Linear regressions on Ni_4Mo and Ni_3Mo

A subset of data from Ni-Mo dataset, which includes 642 atomic configurations only in the Ni_3Mo and Ni_4Mo stoichiometries, is utilized to evaluate the performance of the two descriptors. Linear regression is fitted to this data for each representation using a set of descriptors obtained through different hyperparameters in Eq. 4.2, while varying the coefficients of force's contribution to the total loss function.

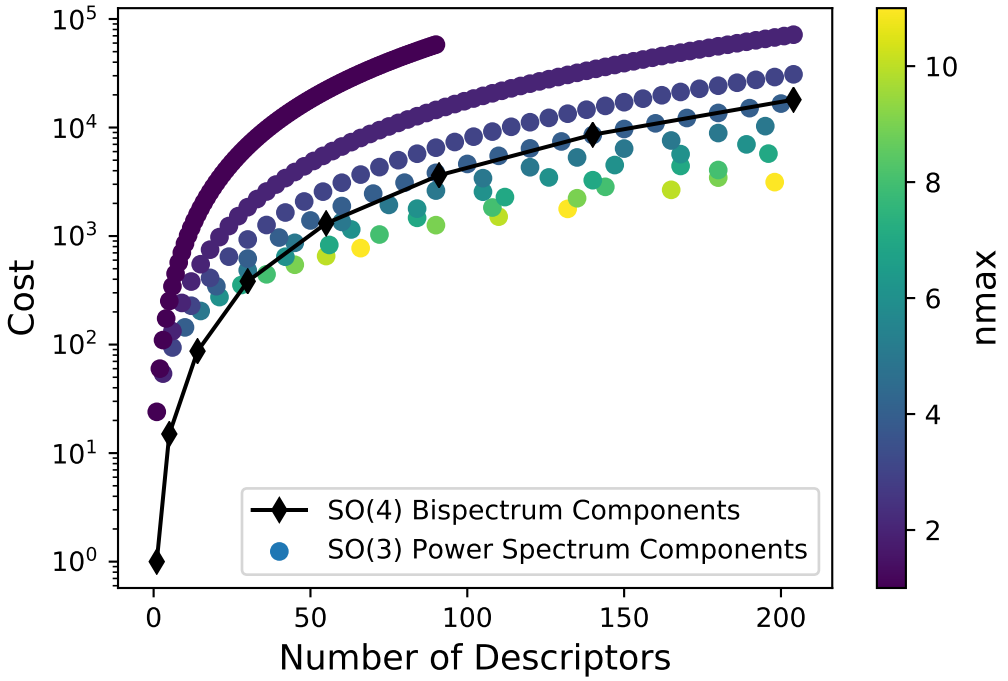


Figure 4.6: The computational cost of the SO(4) bispectrum descriptor and the smooth SO(3) power spectrum descriptor versus the total number of elements of that descriptor. The smooth SO(3) power spectrum is also colored according to the number of radial components in the expansion.

The results of these regressions are shown in Figure 4.7. Clearly, there is a general trend that both SO(3) power spectrum and SO(4) bispectrum can continuously achieve better accuracy with the inclusion of more components, although at high bandlimits that increased accuracy becomes marginal. In addition, the results show that high bandlimit fits vary less with respect to the change of force coefficient, indicating a convergence of the regression. However, a full convergence at high bandlimits results an in incredibly expensive calculations. In real applications, it is generally advised to choose a smaller bandlimit. For the SO(4) bispectrum components, holding the truncation of $j_{\max} = 3$ is a rather common choice [128, 90, 143]. A more detailed analysis regarding the cost of computing the SO(4) bispectrum components with respect to $j_{\max}$ can be found in Ref. [37]. Therefore, the aim is to for a better solution through investigating the smooth SO(3) power spectrum.

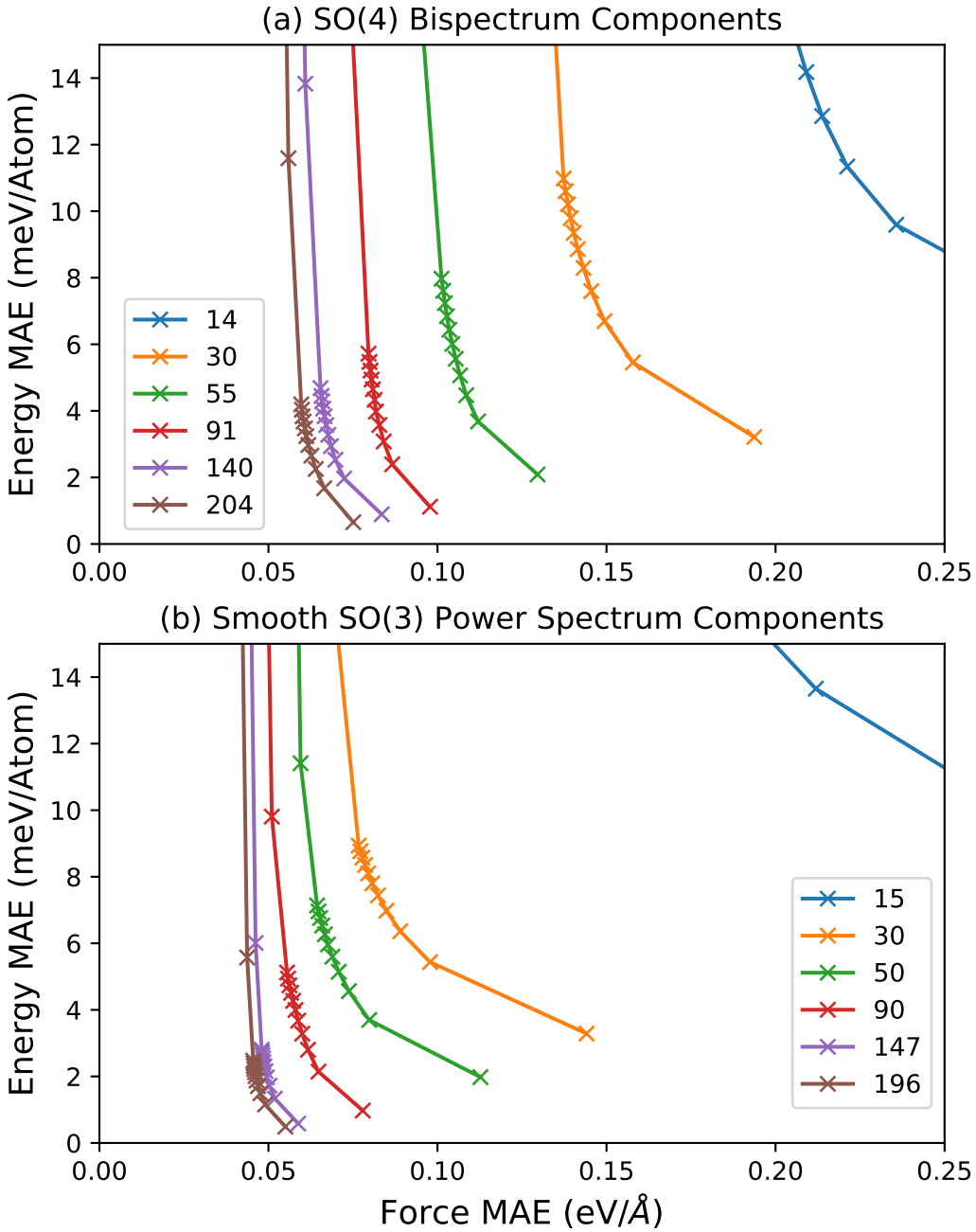


Figure 4.7: Linear regressions of both the SO(4) and SO(3) representations with varying numbers of components. The force coefficients used fall between 1e-6 and 1e+0 with most points falling between 1e-5 and 1e-4.

Although both descriptors yield satisfactory accuracy on the Ni_3Mo/Ni_4Mo data set, it is difficult to maintain the same level of accuracy when extending the training dataset with other stoichimetries (e.g., elemental Ni/Mo) for the regression. In principle, one can improve the regression by tuning force and stress coefficients, applying regularization, and adopting a nonuniform weight scheme on each sample [37, 90]. However, a more automated approach to dealing with large data is to employ a more flexible regression model such as NN regression to be presented in the following subsection.

4.3.4 Neural network regressions on Ni-Mo alloys

When dealing with a large amount of data, linear regression requires very fine tuning of hyperparameters to achieve acceptable accuracies. To achieve these accuracies, optimization schemes are adopted to adjust hyperparameters such as descriptor size, specie weights, cutoff radii, and nonuniform data weighting so that obtaining an optimal fit requires many training cycles [37, 90, 142]. Neural network regression provides a more automated approach to achieve greater accuracy on larger datasets without the need for high bandlimit descriptors or heavy hyperparameter optimization. In this study a small set of descriptors (30) is applied to train a MLP for the entire Ni-Mo dataset consisting of over 4000 structures to satisfactory accuracy through a simple feed forward neural network consisting of two hidden layers of 16 neurons each. For a fair comparison, two sets of descriptors are prepared: (1) the bispectrum components with $j_{\max} = 3$; and (2) the smooth SO(3) power spectrum components with $l_{\max} = 4$ and $n_{\max} = 3$. To ensure that the results can describe elastic deformation well, virial stress contribution is included in the training. Correspondingly, the hyperparameters β, γ, and λ for the evaluation of the loss function (Eq. 2.74) are set to 3e-3, 1e-4, and 1e-8 in all subsequent neural network runs.

Table 4.7 lists the training results in terms of energy and force for all three models. In the previously reported linear SNAP model [90], the overall fitting results are 22.5 meV/atom in energy MAE, and 0.23 eV/Å. Clearly, both neural networks models are able to yield significantly better results (6 meV/atom for energy and 0.08 eV/Å for force) than the previous reported linear model.

Table 4.7: Comparison of the spectral Neural networks models' MAE values from different descriptors. For reference, the previous NiMo model trained from SNAP [90] is also included. Note that in the SNAP model [90], only 247 Mo structures were used for training. In this work, the elastic configuration data are replaced with the data set from Ref. [142]. The parentheses gives the number of configurations for each group.

Properties	Descriptor	$j_{\max}$	$l_{\max}$	$n_{\max}$	Architecture	Mo (377)	Ni (414)	Mo_{Ni} (918)	Ni_{Mo} (1668)	Ni_3Mo (321)	Ni_4Mo (321)	Overall (4019)
	SO(4)[90]	3			Linear Reg.	16.2	7.9	22.7	33.9	5.2	4.0	22.5
Energy	SO(4)	3			30-16-16-1	6.2	7.3	5.6	6.1	6.1	6.4	6.1
(meV/atom)	SO(3)		4	3	30-16-16-1	6.3	3.6	6.2	6.7	4.9	4.6	5.9
	SO(4)[90]	3			Linear Reg.	0.29	0.11	0.13	0.55	0.16	0.14	0.23
Force	SO(4)	3			30-16-16-1	0.19	0.07	0.06	0.10	0.10	0.09	0.10
(eV/Å)	SO(3)		4	3	30-16-16-1	0.18	0.04	0.06	0.10	0.09	0.07	0.08

Notably, the linear regression also reports drastically lower accuracy in both energy and force for the Mo_{Ni}/Mo_{Ni} sets, suggesting that the elemental Ni/Mo and Ni_3Mo/Ni_4Mo portions of the data were weighted much higher in the regression. In particular, the 1668 Ni_{Mo} set, occupying the largest percentage of the data, has a energy MAE of 33.9 meV/atoom and force MAE of 0.55 eV/Å. As such, the predictability of linear SNAP model is likely to be limited in describing the configurations in the vicinity of the Mo_{Ni}/Mo_{Ni} alloys. In contrast, the neural network regressions do not need a special weighting scheme. The models from both SO(4) bispectrum and SO(3) power spectrum yield not only lower energy and force errors for the overall fitting. The energy/force errors for each group are also more evenly distributed.

The elastic tensor is another important metric to check if the trained MLPs are able to reproduce the fine details of the PES on the representative basins. To ensure a satisfactory fitting to the elastic properties, training on the stress tensors for the elastic configurations from the previous works [90, 142] is included. Table 4.8 shows the predicted elastic properties from each model for the ground state structures of BCC Mo, FCC Ni, Ni_3Mo, and Ni_4Mo. In agreement with the previously reported linear SNAP model [90], the elastic data predicted by each MLP agrees with the reported DFT result within similar levels of accuracy across all four ground state structures. In the previous work, it is likely that the authors adjusted the weight for each group of structures in order to achieve a better fit in the elastic properties at the expense of accuracy in energy and force. However, these neural networks regressions can circumvent this trade-off by using a more flexible expression in describing the target properties (energy, force, stress tensor) in fitting. As such, the neural networks models can yield greater accuracy with respect to energy and force while maintaining accuracy in elastic properties all without the need for heavy hyperparameter optimization.

Last, it is also of interest to compare the performance of fitting between the SO(4) bispectrum and smooth SO(3) power spectrum models. In the previous section, it is clear that SO(3) is superior to SO(4) in the context of linear regression. However, this is no longer the case for neural network regression. With the same number of descriptors (30), both NN models yield very similar levels of accuracy. In terms of elastic properties prediction, the SO(4) model seems to be slightly better

Table 4.8: Comparison of elastic properties predicted from several different models for Ni-Mo dataset. B and G denote the empirical Voigt-Reuss-Hill average of bulk and shear moduli respectively. ν is the Poisson's ratio.

	DFT	SNAP[90]	SO(4)	SO(3)
σ(MAE) (GPa)		N/A	0.295	0.289
Mo				
c_{11} (GPa)	472	475	487	479
c_{12} (GPa)	158	163	153	168
c_{44} (GPa)	106	111	108	82
B (GPa)	263	267	265	271
G (GPa)	124	127	129	106
ν	0.30	0.29	0.29	0.33
Ni				
c_{11} (GPa)	276	269	275	271
c_{12} (GPa)	159	150	162	150
c_{44} (GPa)	132	135	137	120
B (GPa)	198	190	199	188
G (GPa)	95	97	96	88
ν	0.29	0.28	0.29	0.30
Ni_3Mo				
c_{11} (GPa)	385	420	426	402
c_{22} (GPa)	402	360	354	382
c_{33} (GPa)	402	408	379	394
c_{12} (GPa)	166	197	159	159
c_{13} (GPa)	145	162	133	109
c_{23} (GPa)	131	145	208	173
c_{44} (GPa)	58	N/A	54	70
c_{55} (GPa)	66	N/A	68	52
c_{66} (GPa)	94	84	79	58
B (GPa)	230	243	240	229
G (GPa)	89	100	80	80
ν	0.33	0.32	0.35	0.34
Ni_4Mo				
c_{11} (GPa)	313	326	319	343
c_{33} (GPa)	300	283	294	293
c_{12} (GPa)	166	179	166	160
c_{13} (GPa)	186	164	199	193
c_{44} (GPa)	130	126	136	131
c_{66} (GPa)	106	N/A	102	113
B (GPa)	223	220	221	222
G (GPa)	91	95	96	102
ν	0.33	0.31	0.31	0.30

than SO(3) though SO(3) generated a slightly lower MAE value for stress tensors overall. It is important to note that each neural networks training follows a stochastic optimization process. So each time, it may generate slightly different results. Hence the comparison is not definitive. From the point view of computational cost, computing the 30 bispectrum components is less expensive than computing the same number of power spectrum components. Therefore, it is fair to conclude that two descriptors are competitive for the application of neural networks regression.

4.3.5 Summary

First, the computational cost of bispectrum components and smooth power spectrum has been evaluated as the SO(3) power spectrum are much costly than the SO(4) bispectrum components at low band limit ($N \leq 30$). However, more term on the radial expansion of the SO(3) power spectrum gives lower cost in comparison to the SO(4) bispectrum compoenents when higher band limit is considered. Using these descriptors to fit machine learning interatomic potentials for a small set of Ni-Mo stoichiometries within a narrow chemical composition space, both descriptors are able to yield satisfactory accuracy within the framework of linear regression. However, the linear regression is not easily extended to fit a more diverse data set from a larger chemical composition space and even then accuracy can still be lacking without hyperparameter optimization such as descriptor size, specie weights, cutoff radii, and nonuniform data weighting. Hence, neural networks regression paired with the SO(4) bispectrum components or the smooth SO(3) power spectrum components can provide a better trained model without the need for large band limit descriptors or heavy hyperparameter optimization. The validity of the trained models are further supported by the accuracy of elastic property calculations. Last, the SO(3) power spectrum descriptor clearly exhibits better agreement with the total energy than the SO(4) bispectrum components, thus it is a better choice for linear regression. However, when adopted to the neural networks regression, both descriptors tend to yield the same level of accuracy.

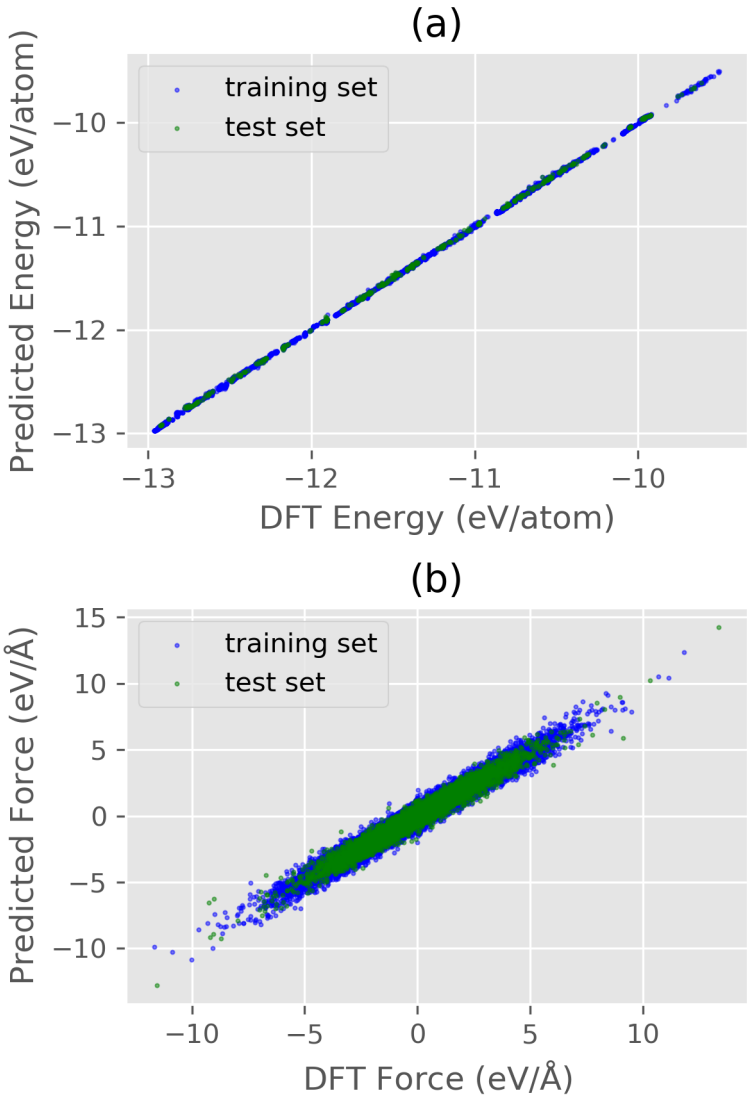


Figure 4.8: The correlation plots between NNP and DFT for (a) energy and (b) forces in the HEA system. The energy MAE values are 6.69 meV/atom and 7.57 meV/atom for training and test sets, while the force MAE values are 0.14 eV/Å and 0.17 eV/Å.

4.4 High Entropy Alloy

High entropy alloys (HEAs) are systems that encompass four or more equimolar/near-equimolar alloying elements. It has been shown that HEAs carry many interesting properties such as high hardness and corrosion resistance [144, 145]. Due to the high computational cost of DFT method, HEA serves as a great example in MLP development with PyXtal_FF. Here, NbMoTaW HEA will be used as an example [146], which are comprised of elemental, binary, ternary, and quaternary systems. Each of the elemental systems has their ground state, strain-distorted, surface, and AIMD configurations. The binary alloys are composed of solid solution structures with the size of 2 $\times$ 2 $\times$ 2 supercell. Lastly, 300 K, 1000 K, and 3000 K AIMD configurations along with special quasi-random structures establish the ternary and quaternary data points. The total structures used in developing the MLP are 5529 configurations for training set and 376 configurations for test set.

In the original work [146], SNAP based on the linear regression predicted that the MAE values for energies are 4.3 meV/atom and 5.1 meV/atom for the training and test sets, and the MAE values in forces are 0.13 eV/Å and 0.14 eV/Å. The results demonstrated a quite satisfactory accuracy comparable to the quantum calculation. However, it needs to be noted that the energy training in Ref. [146] was based on the comparison of formation energy relative to the elemental solids, which spans from -0.193 to 0.934 eV/atom for the entire dataset, whereas the atomic energy spans from -12.960 to -9.502 eV/atom. Training with the energy in a normalized range can surely reduce the error of fitting. However, this fitting method does not fully solve the force field prediction problem since it relies on some DFT reference data. Therefore, the linear regression model to fit only the absolute DFT energy based on the same descriptor as used in Ref. [146] was attempted to be reproduced. The resulting MAE values are 944 meV/atom and 6.329 eV/Åfor energy and force when the force coefficient is 10^{-4}. Furthermore, the MAE values of formation energy fitting yield remarkable improvement of 22 meV/atom and 0.243 eV/Åfor energy and force, respectively. Despite this improvement, the accuracy is insufficient. Perhaps, it is due to lack of fine tuning of hyperparameters, such as atomic weights and cutoff radii for each species.

Table 4.9: Comparison of physical properties predicted with SO3-NNP. The DFT and experiment values are obtained from Ref [146]. B and G denote the empirical Voigt-Reuss-Hill average bulk and shear moduli. ν is the Poisson's ratio. The DFT results were taken from open database of Materials Project [19].

	C_{11} (GPa)	C_{12} (GPa)	C_{44} (GPa)	B (GPa)	G (GPa)	ν
Mo						
DFT	472	158	106	262	127	0.30
Ref[146]	435	169	96	258	110	0.31
NNP	**453**	**161**	**107**	**259**	**121**	**0.30**
Nb						
DFT	233	145	11	174	24	0.45
Ref[146]	266	**142**	**20**	183	**32**	0.42
NNP	**255**	130	-3	**171**	N/A	**0.47**
Ta						
DFT	265	158	69	194	63	0.35
Ref[146]	**257**	**161**	**67**	**193**	59	0.36
NNP	280	165	72	203	**66**	**0.35**
W						
DFT	510	201	143	304	147	0.29
Ref[146]	560	218	154	332	160	**0.29**
NNP	**527**	**196**	**143**	**306**	**151**	**0.29**

To obtain a better accuracy, it is natural to employ a model with more flexibility such as NNP. The NNP will be trained to fit the absolute DFT energy and forces with the smooth SO(3) power spectrum as the descriptors, which are formed by $n_{\max}$=4 and $l_{\max}$=3 with 40 components in total up to the cutoff radius of 5.0 Å. The NNP training is executed with 2 hidden layers with 20 nodes for each layer while energy, force, and stress contributions are trained simultaneously. The importance coefficients of force and stress are set to 10^{-3} and 10^{-4}, respectively. The results of the NNP training is illustrated in Figure 4.8. The NNP energy MAE values for the training and test sets are 6.69 meV/atom and 7.57 meV/atom, and the NNP force MAE values are 0.14 eV/Å and 0.17 eV/Å. In addition, the MAE value of stress for training set is 0.078 GPa. Our results of energy and force yield worse performance compared to the previous report. Nevertheless, our NNP model offers a more general representation of the DFT PES since it does not rely on any prior reference values (i.e. formation energy relies on prior reference values).

Furthermore, Table 4.9 shows the physical properties calculations such as elastic constants, bulk and shear moduli, and the Poisson's ratio of the cubic elemental crystals. From the table, the overall performances of NNP in predicting the physical properties are reasonable, except that the C_{44} value of Nb is negative. However, this is consistent with the fact that the DFT's C_{44} is also significantly lower than other terms. Hence, the negative C_{44} is acceptable if one considers the noise of stress data in training. Meanwhile, the C_{44} value may be remedied by providing additional training dataset focusing on the shearing effect of Nb, increasing the importance of the stress coefficient, or increasing the hidden layer size in the NNP training. In addition, SO3 descriptor can be conveniently expanded in terms of both radial basis ($n_{\max}$) and angular momentum ($l_{\max}$) for achieving better overall accuracy.

In summary, linear regression model has applied to the absolute energy and forces on NbMoTaW HEA dataset, while unsatisfactory results were obtained at 944 meV/atom and 6.329 eV/Åfor energy and force, respectively. Despite the major improvement, fitting to the formation energy and forces with linear regression results in insufficient performances at 22 meV/atom and 0.243 eV/Åfor energy and force, respectively. The discrepancies in results with the original study [146]

can be due to the lack of hyperparameters optimizations. Hence, the neural networks model with SO(3) power spectrum as the descriptors was applied to train on the absolute energy and forces. The NNP energy MAE values yields 6.69 meV/atom and 7.57 meV/atom. Meanwhile, the force MAE values are 0.14 eV/Åand 0.17 eV/Å. In addition, the MAE value of stress is 0.078 GPa for training set. Finally, the elastic constants, bulk and shear moduli, and the Poisson's raito of cubic elemental crystals were predicted with reasonable results (see Table 4.9).

4.5 Pt MLP for General Purposes

4.5.1 Dataset

Compared to crystalline systems, surfaces and nanoparticles generally represent the more challenging cases in MLP training as the nanoparticle contain more versatile atomic environments and more complex PES is expected. Here, the NNP model was applied to a Pt dataset [103], which consists of three data types: Pt surface, Pt bulk, and Pt cluster. There are 927 clusters of 15 atoms, and the Pt bulk type consists of 1717 configurations which are composed of 256 atoms. Pt surface are constructed from (001), (110), and (111) surfaces. Respectively, there are 949, 819, and 700 structures which consist of 320, 160, and 320 atoms.

4.5.2 MLP Setup

The SO3 power spectrum descriptor with $l_{\max} = 3$ and $n_{\max} = 4$ at radius cutoff of 4.9 Å was used to construct the MLP in the NNP model with two hidden layers with 30 nodes each. Unlike the previous examples, the minibatch scheme with the Adam optimizer was employed. In each iteration, the training process was updated in a batch size of 25 configurations.

4.5.3 MLP results

In the original literature [103], DeepPot-SE includes MoS_2 slab and Pt clusters on MoS_2 substrate (MoS_2/Pt). The performance of DeepPot-SE yields satisfactory results. Meanwhile, embedded atom neural networks method can achieves outstanding results using a fraction of the same dataset

Table 4.10: The trained RMSE values of the predicted energy and forces of Pt dataset from the 30-40-40-1 NNP model. For reference, the results from the DeepPot-SE model [103] is also reported. It should be noted that that DeepPot-SE results were based on training the entire MoS_2/Pt dataset.

	SO3-NNP		DeepPot-SE [103]	
	Energy (meV)	Force (meV/Å)	Energy (meV)	Force (meV/Å)
Bulk	1.64	64	2.00	84
Surface	5.91	87	6.77	105
Cluster	7.63	247	30.6	201

[105]. Both of the methods exploit a large number of neural networks parameters, in the order of 10^4–10^5, while the current study only adopts 2191 weight parameters for exemplary purpose. As shown in Table 4.10, the accuracy from our small neural network model is comparable to that of DeepPot-SE results. Not surprisingly, the group of Pt bulk has the lowest errors with only 1.64 meV/atom for the RMSE in energy and 67 meV/Å in force. On the contrary, the errors on Pt clusters are about 2-4 time higher for both energy and forces. This is expected since the local atomic environments in the clusters are more diverse and thus learning the relation is harder. Nevertheless, the values from this exploratory study is comparable to the results from deep learning models. This example also suggests that a small NNP model with the properly constructed features can be a complementary solution for MLP development.

Furthermore, the equilibrium lattice constant (a) and the total energy of fcc Pt are calculated with the trained Pt MLP. The structural optimization was performed in the framework of ASE through BFGS (a second order optimization method) optimizer with 4 atoms in the unit cell. The equilibrium lattice constant predicted by the Pt MLP is 3.996 Å, whereas the lattice constant from DFT method is 3.985 Å [147]. The DFT ground state energy [19] of the fcc Pt is -6.057 eV/atom, while our result suggests -6.054 eV/atom with 3 meV/atom discrepancy with respect to the DFT result. Along with the physical properties of fcc Pt, the energy of fcc Pt are tabulated in Table 4.11.

Table 4.11: The equilibrium energy and physical properties of fcc Pt. Δ is the absolute relative error in percentage between DFT and NNP values. The DFT results were collected from open database of Materials Project [19].

	E_0	a	C_{11}	C_{12}	C_{44}	B	G	ν
	eV/atom	Å	(GPa)	(GPa)	(GPa)	(GPa)	(GPa)	
DFT	-6.057	3.985	303	220	54	247	49	0.41
NNP	-6.054	3.996	293	243	131	259	68	0.37
$\Delta(\%)$	0.05	0.28	3.30	10.5	143	4.86	38.8	9.76

The overall result is encouraging except for the C_{44}. As the training dataset was generated from MD-simulated method, there are probably a few of the structures to none that describe the shearing behavior of fcc Pt. Therefore, it is expected that the NNP fails to describe C_{44}. To improve the accuracy of C_{44}, one can include the training of stress, which was not involved previously, or adding structures with shearing effect of fcc Pt.

Chapter 5 Nudged Elastic Band Simulations

5.1 Introduction

Nudged elastic band (NEB) simulation [148] is a method for finding the minimum energy of a transition path for a given system [149, 150]. NEB simulations have successfully applied in the framework of DFT calculations [151, 152, 153], and in combination with classical force field [154, 155]. Studies for large systems (over 3 x 10^6 atoms) have been performed [156]. Given an initial and a final states, NEB constructs images in between the initial and the final states of the system, and relaxes the images. These "images" are sometimes called "chain" of "replicas", typically in the order of 4-20 replicas between the initial and final states. A spring interaction is attached in between two consecutive images ensuring a continuous path (i.e. this mimics an elastic band). In Fig. 5.1, a chain of 5 replicas are used to illustrate the NEB simulation to find the minimum energy path (MEP). In practice, the chain will be initially interpolated as a "straight" line between the initial configuration and the final configuration.

Here, the diffusion barriers of Pt adsorption on Pt(111) and Pt(100) surfaces will be examined using the NEB method. The MLP from the earlier section will be employed for the simulations. The simulations will be enabled by PyXtal_FF via ASE package. The structural optimizations will be performed using L-BFGS method while the surface structures will be constructed using the ASE package as well. The initial guess from initial to final images. The convergence criteria is that the all of the atomic forces is less than a certain value (f_{max})such that:

$$f_{max} > \max|\boldsymbol{F}_a| \tag{5.1}$$

where most of the energy convergence criteria is set to 0.01 eV/Å.

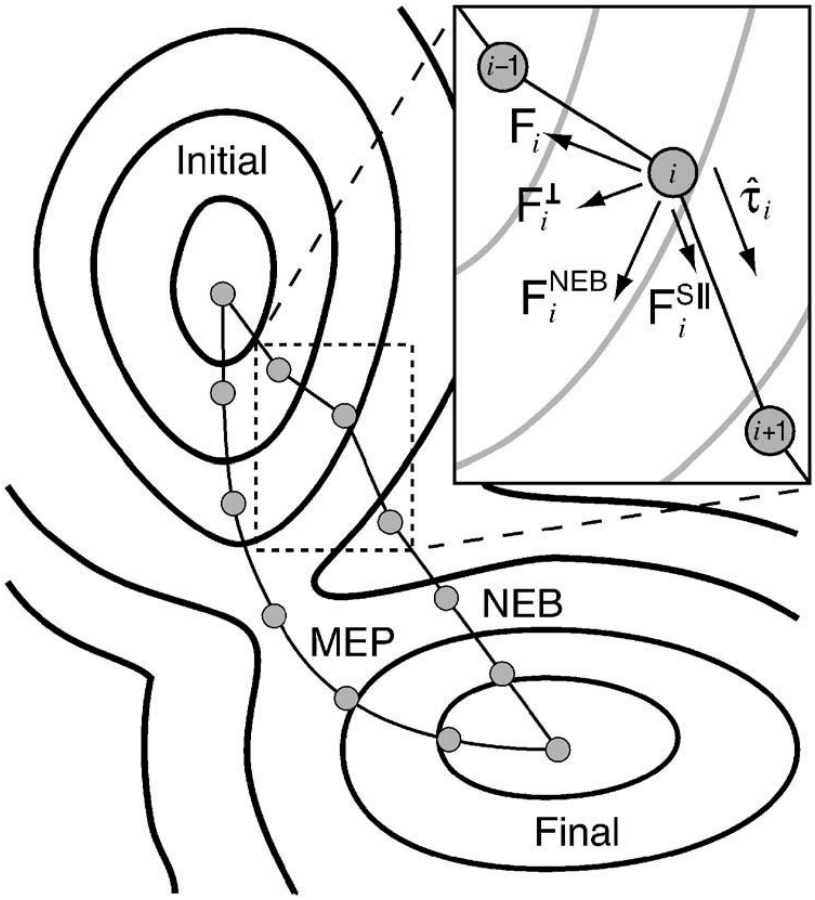


Figure 5.1: An illustration of a NEB simulation. The MEP is the optimal transition state for that particular system. In order to find the MEB, forces (perpendicular force $\boldsymbol{F}^{\perp}$ and parallel/spring force $\boldsymbol{F}^{\parallel}$) are to be optimized. $\boldsymbol{F}^{NEB}$ is the total net force. Source: https://theory.cm.utexas.edu/henkelman/research/saddle/.

5.2 Pt(111) Surface

Here, we will discuss adsorption and diffusion of a Pt adatom on the Pt(111) surface. The Pt(111) surface is constructed with a supercell of (5x5) with 10 layers deep as the convergence was examined with respect to the number of layers. A vacuum of 20 Å are applied to the two surfaces. The bond length of Pt-Pt is 2.81 Å. An illustration of a Pt(111) surface is depicted in Fig. 5.2. There are 4 types of arrangements for the adatom on the surface: "ontop", "bridge", "hcp", and "fcc".

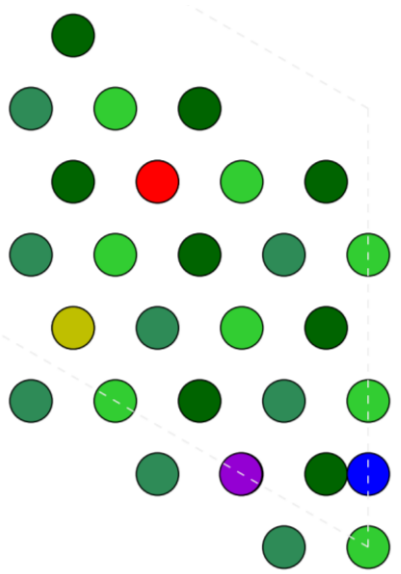
Fcc site is the preferred binding site in Pt(111) surface. This site can also be considered as the "threefold" site the adatom is bonded to the three nearest atoms on the surface (see Fig. 5.2). The Pt adatom at fcc site lies closer to the surface than the others, because the binding in the fcc geometry is somewhat stronger. The predicted bond length of the fcc adatom to the nearest neighboring atom is 2.63 Å (i.e. about 93% of adatom-adsorbent to Pt-Pt bond length). In consequence, it agrees with the expected bond-order bond-length correlation [157].

The bridge site is an unstable site. Initially, the adatom at the bridge site is relaxed by constraining the x and y axes (i.e. relaxation only on the distance above the surface) yielding the adatom-adsorbent bond length of 2.59 Å. Afterwards, the adatom and the top layer atoms are permitted to relax resulting the energy difference of 0.34 eV between the bridge and the fcc sites. In the previous study [157, 158], the range for the diffusion barrier is between 0.33-0.47 eV. In the lower end of the predicted diffusion barriers, the adatom and the first two top layer are allowed to be relaxed. Hence, our result yields a consistency with the previous study.

The hcp site is a local minima site. Relaxing the adatom and the top layer atoms produces a preference of 0.18 eV for the fcc against the hcp site, where the bond length of adatom-adsorbent is 2.63 Å. Our result is consistent with the previous studies where the adsorption energy difference between the fcc and hcp site is 0.17 ± 0.03 eV [157, 158]. This difference is considered large due to the angular behavior of the d orbitals in the Pt system [159]. Due to this difference, the adatom does not visit the hcp site when it diffuses from one fcc site to another fcc site [158].

Now, the diffusivity from one fcc site to another fcc site will be performed using NEB method.

(a)



(b)

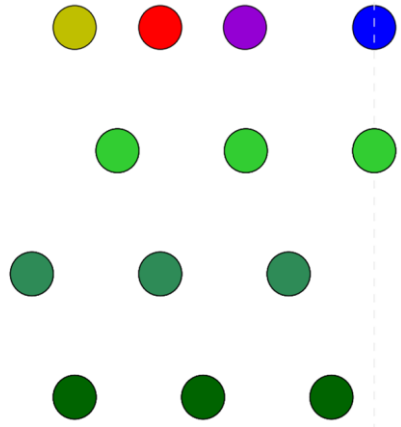


Figure 5.2: A 3-layer deep Pt(111) surface shaded by multiple layers of greens with (a) top view and (b) side view. The blue, yellow, and red adatoms are on the ”bridge”, ”fcc”, and ”hcp” sites, respectively. For comparison, the purple adatom represent the ”ontop” site. The adsorbent is arranged in ABC packing. Since Pt is a fcc metal, it prefers high to low coordination. Hence, the one-fold bonding type (i.e. ”ontop” site) is not considered.

Both equilibrium sites are fully relaxed by allowing the adatom and the first top layer atoms to move as needed. There are 7 ”chain” of replicas in between the two equilibrium sites. The energy barrier crossing the ”bridge” site from the NEB method is 0.21 eV, whereas we obtained 0.34 eV. This energy difference is due to NEB simulation allows the adatom's neighbors to move accordingly as it passes the ”bridge” site. Hence, it lowers the overall energy barrier. In addition, the adatom nearly crosses the hcp site as it diffuses with the bond length to the nearest surface atom of 2.62 Å (bond length at the hcp site is 2.63 Å). Due to the small change in bond length, the energy difference between near hcp and hcp sites does not exceed 0.001 eV. Experimentally, the diffusion barrier of 0.25 $\pm$ 0.02 eV has been performed using field-ion miscroscopy (FIM) measurements within temperature range of 92-100 K [157].

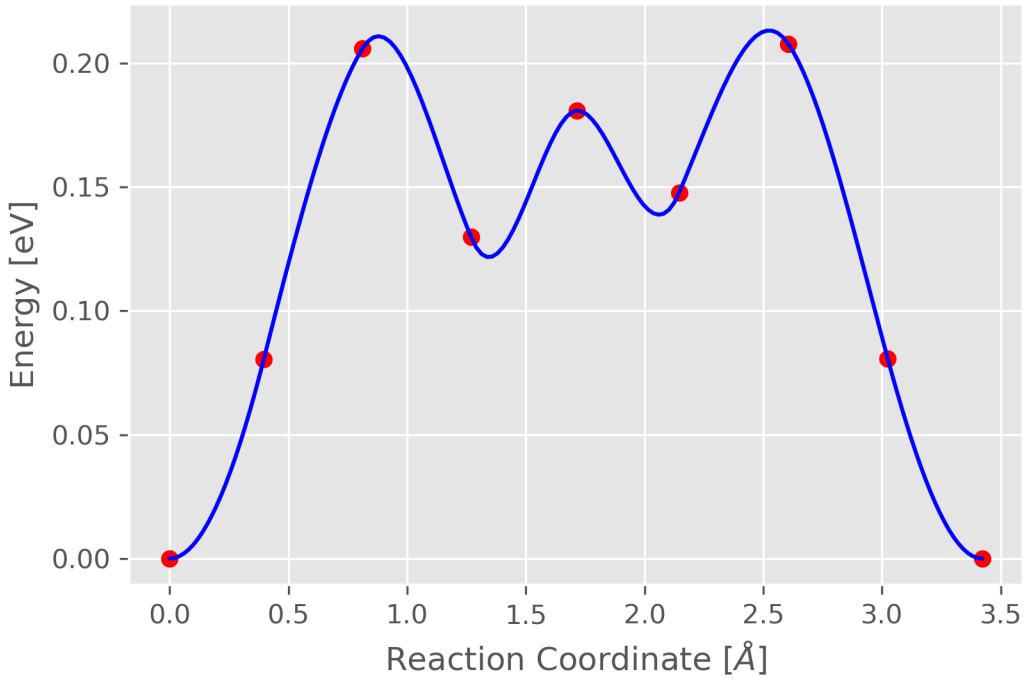


Figure 5.3: The result of NEB simulation of Pt adatom on Pt(111) surface diffused from one fcc site to the other nearest fcc site.

5.3 Pt(100) Surface

Here, we will discuss adsorption and diffusion of a Pt adatom on the Pt(100) surface. The Pt(100) surface is constructed with a supercell of (4x4) with 10 layers deep as the convergence was examined with respect to the number of layers. A vacuum of 20 Å are applied to the two surfaces. There are 3 types of arrangements for the adatom on the surface: ”ontop”, ”bridge”, ”hollow”.

The ”hollow” site is the preferred binding site of the Pt adatom on Pt(100) surface. The ”hollow” site can be regarded as fourfold site. Here, the surface diffusion energy barrier from one hollow site to the next nearest hollow site using NEB method. During the NEB simulation, the first layer atoms and the adatoms are permitted to relax. Prior to the NEB simulation, the two equilibrium sites are fully relaxed. The bond-length of the adatom-adsorbent at the ”hollow” site is 2.64 Å (i.e. about 94% of adatom-adsorbent to Pt-Pt bond length). Finally, a diffusion barrier of 0.83 eV for passage over a bridge is predicted (see Fig. 5.4).

5.4 Summary

NEB simulations have been performed to predict the diffusion barriers of Pt adsorption on Pt(111) and Pt(100) surfaces. Since Pt is a fcc metal, it prefers high to low coordination. Hence, the one-fold bonding type (i.e. ”ontop” site) is not considered for both Pt(111) and Pt(100) surfaces. While the ”traditional” method of relaxing the adatom on Pt(111) surface from the ”bridge” site to ”fcc” site fails to reproduce the experimental result, our result using NEB method is consistent with the experimental diffusion barrier. In addition, we were able to reproduce the previously studied results with the ”traditional” method. For Pt(100) surface, the ”hollow” site is the preferred site as the passage over the ”bridge” site takes 0.83 eV.

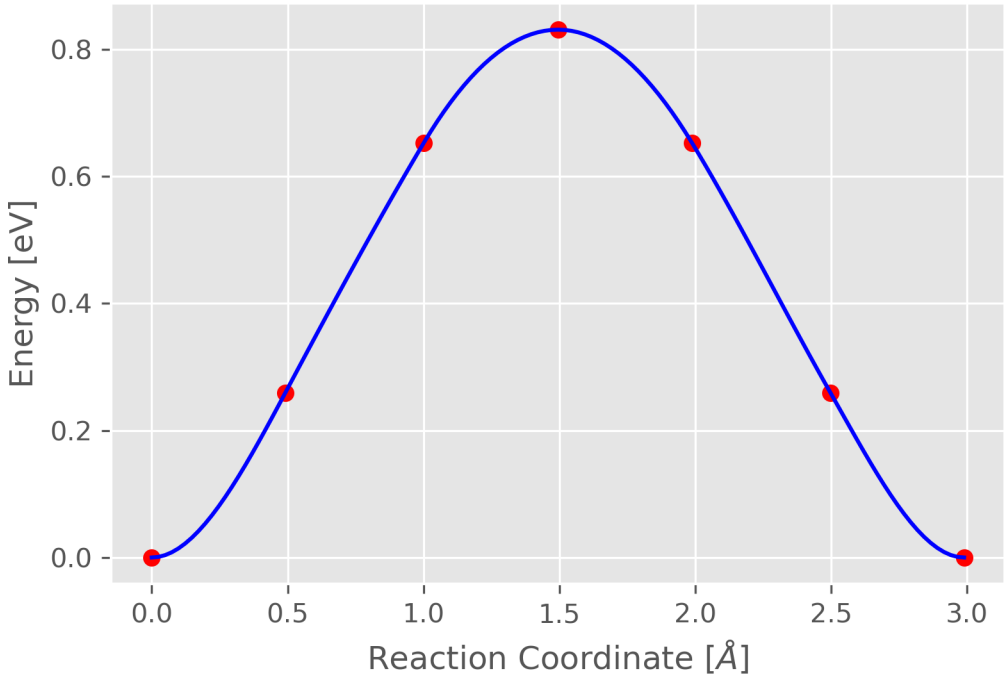


Figure 5.4: The result of NEB simulation of Pt adatom on Pt(111) surface diffused from one fcc site to the other nearest fcc site.

Chapter 6 Conclusions and Future Work

Atomistic simulation with MLP is a useful computational tool to examine materials at microscopic level. MLP has the ability to perceive the atomic environment among crystal structures and thus provides accurate descriptions of the forces governing the atomic motion. As seen in Chapter 5, the diffusion barriers of Pt adsorption on Pt(111) and Pt(100) surfaces have been successfully reproduced using the developed Pt MLP. Moreover, the surface diffusion energy barriers using the (NEB) method have painted a clearer description in comparison to the experimental values. This effort has shown that atomistic simulation at DFT level while retaining low computational cost is truly tractable.

The atomistic simulation with MLP is facilitated through PyXtal_FF, a Python software package that I have developed during my doctoral study. The aim of PyXtal_FF is to promote the application of atomistic simulations with MLPs by providing several choices of atom-centered descriptors and machine learning regressions in one platform. Presently, the PyXtal_FF is equipped with two regression models and four types atom-centered descriptor, and counting. With the PyXtal_FF, we have successfully developed MLPs for SiO_2, crystalline Si, Ni-Mo, HEA NbMoTaW, and Pt nanoclusters. The predictive ability of these MLPs are at the level of DFT accuracy. Since it is written in Python, this platform is constructed in modular approach, enabling convenient effort to add new functionality. We are going to work on interfacing the trained MLP with LAMMPS [121] to enable the large-scale atomistic simulation in the near future.

In the future, we foresee the upcoming developments for PyXtal_FF:

Active Learning. In this thesis, I have shown the development of Si MLP for general purposes. Despite the great predictive ability of the physical properties (see Section 4.2), the training dataset generated from random crystal structures perhaps contains many redundant structures. This can pose several problems such as bias in fairness [160]. Machine learning algorithms can achieve excellent predictability on the data that are well-presented, while it will fail on data that are under-

represented. Due to the issue, active learning has been suggested to rectify the unfairness in dataset [161]. It has been shown that active learning is also provide wonderful pathway in crystal structure predictions as well [94, 162]. Currently, we are developing Bayesian Optimization algorithm coupled with GPR to accelerate crystal structure prediction to find the global minimum structure. The idea of this method is to minimize the number of electronic optimization through the expensive DFT calculations. The dataset generated from this method, instead of random structure generation, can potentially be useful in developing more accurate MLP avoiding unfairness in the dataset.

Additional atom-centered descriptors. Although MLP method is orders of magnitude faster than the DFT method, the bottleneck for MLP is at the construction of atom-centered descriptors. For example, the usage of symmetry functions as the atom-centered descriptors is about two orders of magnitude times slower than the Tersoff potential. The creation of new atom-centered descriptors must satisfy the requirements stated in Section 2.5.3.

Additional MLPs for nanoparticles. In this thesis, I have demonstrated the power of MLP on predicting the diffusion barriers of Pt adsorption on Pt surfaces. In the future, the intention is to extend the capability of Pt MLP on more challenging task such as Pt nanoparticles. It is well known that nanoparticles have many modelling challenges. For example, nanoparticles belonging to the exactly same system may have different properties if they have different sizes. I would eager to learn the behaviors of nanoparticles, in particular carbon monoxide binding on Pt nanoparticles.

Appendix A Additional Cutoff Functions

First, let's define a general notation for x:

$$x = \frac{R}{R_c} \tag{A.1}$$

where R is the distance between two atoms and R_c is the cutoff radius.

A.1 Tanh

$$\begin{aligned} f(x) &= (\tanh(1-x))^3 \\ \frac{df}{dx} &= -3\tanh^2(1-x)(1-\tanh^2(1-x)) \end{aligned} \tag{A.2}$$

A.2 Exponential

$$\begin{aligned} f(x) &= \exp\left\{1 - \frac{1}{1-x^2}\right\} \\ \frac{df}{dx} &= -2x\frac{\exp\left\{1-\frac{1}{1-x^2}\right\}}{(1-x^2)^2} \end{aligned} \tag{A.3}$$

A.3 Polynomial1

$$\begin{aligned} f(x) &= x^2(2x-3)+1 \\ \frac{df}{dx} &= 6x(x-1) \end{aligned} \tag{A.4}$$

A.4 Polynomial2

$$\begin{aligned} f(x) &= x^3(x(15-6x)-10)+1 \\ \frac{df}{dx} &= -30x^2(x-1)^2 \end{aligned} \tag{A.5}$$

A.5 Polynomial3

$$\begin{aligned} f(x) &= x^4(x(x(20x-70)+84)-35)+1 \\ \frac{df}{dx} &= 140(x^3(x-1)^3) \end{aligned} \tag{A.6}$$

A.6 Polynomial4

$$\begin{aligned} f(x) &= x^5(x(x(x(315-70x)-540)+420)-126)+1 \\ \frac{df}{dx} &= -630x^4(x-1)^4 \end{aligned} \tag{A.7}$$

Appendix B Derivatives of Atom-centered Descriptors

B.1 The Derivatives of (w)ACSF

Following the Eq. 2.57 the derivative with respect to an atom m can be written in the following form:

$$\frac{\partial G_i^{(2)}}{\partial \boldsymbol{r}_m} = \sum_{j \neq i} e^{-\eta(R_{ij}-R_s)^2} \left(\frac{\partial f_c}{\partial R_{ij}} - 2\eta(R_{ij} - R_s) f_c \right) \frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} \tag{B.1}$$

For the periodic system, the computation of $\frac{\partial R_{ij}}{\partial \boldsymbol{r}_m}$ is straightforward except that one needs to consider one additional case. When $i = j$, the derivative is always zero.

$$\frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} = \begin{cases} 0 & m \notin [i, j] \\ 0 & m = i = j \\ -\frac{\boldsymbol{r}_{ij}}{Rij} & m = i \text{ (when } i \neq j) \\ \frac{\boldsymbol{r}_{ij}}{Rij} & m = j \text{ (when } i \neq j) \end{cases} \tag{B.2}$$

In Eqs. (2.58, 2.59), the cosine function can be defined as:

$$\cos\theta_{ijk} = \frac{\boldsymbol{r}_{ij} \cdot \boldsymbol{r}_{ik}}{R_{ij} R_{ik}} \tag{B.3}$$

where $\boldsymbol{r}_{ij}$ is the relative position between atom j and atom i.

In the following, the expressions for the derivative with respect to an interacting atom m are:

$$\begin{aligned}
&\frac{\partial G_i^{(4)}}{\partial \boldsymbol{r}_m} = 2^{1-\zeta} \sum_{j \neq i} \sum_{k \neq i,j} e^{-\eta(R_{ij}^2 + R_{ik}^2 + R_{jk}^2)} \\
&\Bigg[\lambda\zeta(1 + \lambda\cos\theta_{ijk})^{\zeta-1} \frac{\partial \cos\theta_{ijk}}{\partial \boldsymbol{r}_m} f_c(R_{ij}) f_c(R_{ik}) f_c(R_{jk}) - \\
&2\eta(1 + \lambda\cos\theta_{ijk})^{\zeta} \left(R_{ij}\frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} + R_{ik}\frac{\partial R_{ik}}{\partial \boldsymbol{r}_m} + R_{jk}\frac{\partial R_{jk}}{\partial \boldsymbol{r}_m} \right) \\
&f_c(R_{ij}) f_c(R_{ik}) f_c(R_{jk}) \\
&+ (1 + \lambda\cos\theta_{ijk})^{\zeta} \left(\frac{\partial f_c(R_{ij})}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} f_c(R_{ik}) f_c(R_{jk}) \right. \\
&\left. + f_c(R_{ij}) \frac{\partial f_c(R_{ik})}{\partial R_{ik}} \frac{\partial R_{ik}}{\partial \boldsymbol{r}_m} f_c(R_{jk}) + f_c(R_{ij}) f_c(R_{ik}) \frac{\partial f_c(R_{jk})}{\partial R_{jk}} \frac{\partial R_{jk}}{\partial \boldsymbol{r}_m} \right) \Bigg]
\end{aligned} \tag{B.4}$$

$$\begin{aligned}
&\frac{\partial G_i^{(5)}}{\partial \boldsymbol{r}_m} = 2^{1-\zeta} \sum_{j \neq i} \sum_{k \neq i,j} e^{-\eta(R_{ij}^2 + R_{ik}^2)} \\
&\Bigg[\lambda\zeta(1 + \lambda\cos\theta_{ijk})^{\zeta-1} \frac{\partial \cos\theta_{ijk}}{\partial \boldsymbol{r}_m} f_c(R_{ij}) f_c(R_{ik}) \\
&- 2\eta(1 + \lambda\cos\theta_{ijk})^{\zeta} \left(R_{ij}\frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} + R_{ik}\frac{\partial R_{ik}}{\partial \boldsymbol{r}_m} + R_{jk}\frac{\partial R_{jk}}{\partial \boldsymbol{r}_m} \right) f_c(R_{ij}) f_c(R_{ik}) \\
&+ (1 + \lambda\cos\theta_{ijk})^{\zeta} \left(\frac{\partial f_c(R_{ij})}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} f_c(R_{ik}) + f_c(R_{ij}) \frac{\partial f_c(R_{ik})}{\partial R_{ik}} \frac{\partial R_{ik}}{\partial \boldsymbol{r}_m} \right) \Bigg]
\end{aligned} \tag{B.5}$$

The derivatives of atomic distances, R_{ij} and R_{ik}, carry the same meaning as in Eq. B.2. The expression of the cosine of triple-atom angle is

$$\frac{\partial \cos\theta_{ijk}}{\partial \boldsymbol{r}_m} = \frac{\boldsymbol{r}_{ik}}{R_{ij}R_{ik}} \cdot \frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{r}_m} + \frac{\boldsymbol{r}_{ij}}{R_{ij}R_{ik}} \cdot \frac{\partial \boldsymbol{r}_{ik}}{\partial \boldsymbol{r}_m} - \frac{\boldsymbol{r}_{ij} \cdot \boldsymbol{r}_{ik}}{R_{ij}^2 R_{ik}} \frac{\partial R_{ij}}{\partial \boldsymbol{r}_m} - \frac{\boldsymbol{r}_{ij} \cdot \boldsymbol{r}_{ik}}{R_{ij}R_{ik}^2} \frac{\partial R_{ik}}{\partial \boldsymbol{r}_m} \tag{B.6}$$

$$\frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{r}_m} = \begin{bmatrix} \delta_{mj} - \delta_{mi} & 0 & 0 \\ 0 & \delta_{mj} - \delta_{mi} & 0 \\ 0 & 0 & \delta_{mj} - \delta_{mi} \end{bmatrix} \tag{B.7}$$

where δ_{mj} is the Kronecker delta between atom m and j.

B.2 The Derivatives of EAD

The expression of the derivative with respect to an interacting atom m is shown in the following:

$$\frac{\partial \rho_i}{\partial \boldsymbol{r}_m} = \sum_{l_x,l_y,l_z=0}^{l_x+l_y+l_z=L} \frac{2L_{\max}!}{l_x!l_y!l_z!}\left[\sum_{j\neq i}^{N} Z_j \Phi\right]\left[\sum_{j\neq i}^{N} Z_j \frac{\partial \Phi}{\partial \boldsymbol{r}_m}\right] \tag{B.8}$$

where the derivative of Φ with respect to an interacting atom m is

$$\begin{aligned}\frac{\partial \Phi}{\partial \boldsymbol{r}_m} = \frac{e^{-\eta(R_{ij}-R_s)^2}}{R_c^{l_x+l_y+l_z}}\Bigg[\Bigg(&\frac{\partial x_{ij}^{l_x}}{\partial \boldsymbol{r}_m} y_{ij}^{l_y} z_{ij}^{l_z} + x_{ij}^{l_x}\frac{\partial y_{ij}^{l_y}}{\partial \boldsymbol{r}_m} z_{ij}^{l_z} + x_{ij}^{l_x} y_{ij}^{l_y}\frac{\partial z_{ij}^{l_z}}{\partial \boldsymbol{r}_m}\Bigg) f_c \\ &+ x_{ij}^{l_x} y_{ij}^{l_y} z_{ij}^{l_z}\left(\frac{\partial f_c}{\partial R_{ij}} - 2f_c\eta(R_{ij}-R_s)\right)\frac{\partial R_{ij}}{\partial \boldsymbol{r}_m}\Bigg)\Bigg]\end{aligned} \tag{B.9}$$

B.3 The Derivatives of Bispectrum

For the SO(4) bispectrum components, we need to calculate the Wigner-D matrices for each neighbor. Here we use a polynomial form of the Wigner-D matrix elements suggested by Boyle [163].

$$D^j_{m',m} = \begin{cases} (-1)^{(j+m)} R_b^{2m}\delta_{-m',m}, & |R_a| < 10^{-15} \\ R_a^{2m}\delta_{m',m}, & |R_b| < 10^{-15} \\ \sqrt{\frac{(j+m)!(j-m)!}{(j+m')!(j-m')!}}|R_a|^{2j-2m} R_a^{m'+m} R_b^{-m'+m}\times & \\ \quad \sum_k \binom{j+m'}{k}\binom{j-m'}{j-m-k}\left(-\frac{|R_b^2|}{|R_a^2|}\right)^k, & |R_a| \geq |R_b| \\ (-1)^{j-m}\sqrt{\frac{(j+m)!(j-m)!}{(j+m')!(j-m')!}} R_a^{m'+m} R_b^{m-m'}|R_b|^{2j-2m}\times & \\ \quad \sum_k \binom{j+m'}{j-m-k}\binom{j-m'}{k}\left(-\frac{|R_a^2|}{|R_b^2|}\right)^k, & |R_a| < |R_b| \end{cases} \tag{B.10}$$

where R_a and R_b are the Cayley-Klein parameters representing the rotation. In the angle-axis representation of rotation the Cayley-Klein parameters representing a rotation about an axis defined

by $r = (x, y, z)$ through an angle ω can be written as:

$$\begin{aligned} R_a &= \cos(\omega/2) + i\frac{\sin(\omega/2)}{r}z \\ R_b &= \frac{\sin(\omega/2)}{r}(y + ix) \end{aligned} \tag{B.11}$$

These polynomials are finite and the coefficients of each term are known. Different from previous works [87, 143] based on a recursive scheme as discussed in Appendix C, we evaluate the Wigner-D matrix elements using Horner's method for the terms in the summation, which allows evaluation of a polynomial of degree n with only n multiplications and n additions.

$$\begin{aligned} P(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \\ &= a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + x a_n))) \end{aligned} \tag{B.12}$$

Using Horner's method is also convenient for the simultaneous computation of the gradient. To obtain the gradient with respect to cartesian coordinates, the chain rule is applied through the Cayley-Klein parameters and their conjugates.

In addition, we make use of the symmetries of the SO(4) bispectrum components discovered by Thompson [143].

$$\frac{B_{j_1 j_2 j}}{2j+1} = \frac{B_{j j_2 j_1}}{2j_1+1} = \frac{B_{j_1 j j_2}}{2j_2+1} \tag{B.13}$$

These symmetries reduce the number of necessary bispectrum components to compute to only the unique components which also greatly reduces the complexity of the gradient calculation. For brevity we denote the two inner sums of the bispectrum component calculation as $Z^{m,m'}_{j_1,j_2,j}$[143]:

$$\sum_{m_1,m_1'=-j_1}^{j_1} \sum_{m_2,m_2'=-j_2}^{j_2} c^{j_1}_{m_1',m_1} c^{j_2}_{m_2',m_2} C^{j j_1 j_2}_{m m_1 m_2} C^{j j_1 j_2}_{m' m_1' m_2'}. \tag{B.14}$$

So that, when utilizing the symmetries in Eq. B.13, the gradient of the bispectrum components

with respect to an atom i can be written as[143]:

$$\begin{aligned}\nabla_i B^{(i)}_{j_1,j_2,j} = &\sum_{m,m'=-j}^{j} \nabla_i \left(c^j_{m',m}\right)^* Z^{m,m'}_{j_1,j_2,j} + \\ &\frac{2j+1}{2j_1+1} \sum_{m_1,m_1'=-j_1}^{j_1} \nabla_i \left(c^{j_1}_{m_1',m_1}\right)^* Z^{m_1,m_1'}_{j,j_2,j_1} + \\ &\frac{2j+1}{2j_2+1} \sum_{m_2,m_2'=-j_2}^{j_2} \nabla_i \left(c^{j_2}_{m_2',m_2}\right)^* Z^{m_2,m_2'}_{j_1,j,j_2},\end{aligned} \tag{B.15}$$

where the gradient of the inner product with respect to one atom is:

$$\nabla_i c^j_{m',m} = \nabla_i \left(f_{\text{cut}}(r_i) D^{*j}_{m',m}(r_i)\right) \tag{B.16}$$

B.4 The Derivatives of Power Spectrum

In calculating the smooth SO(3) power spectrum, three main challenges exist. First, the calculation of the spherical harmonics, and second the radial inner product in Eq. 2.70, and third the gradient of the expansion coefficients in Eq. 2.70. To start, the spherical harmonics can be considered as a subset of the Wigner-D matrices in the z-y-z Euler-angle representation, where the spherical harmonic vector Y_l is a row vector of the corresponding D-matrix D^l with some additional scalar factors as given in the equation below[164]:

$$Y_{lm}(\theta,\phi) = (-1)^m \sqrt{\frac{2l+1}{4\pi}} D^l_{0,-m}(\chi,\theta,\phi),$$

where the Wigner-D matrices are in the z-y-z Euler-angle representation and χ is arbitrary, thus, without loss of generality, we choose $\chi = 0$.

The z-y-z Euler-angle representation represents a rotation about the original z-axis through an angle α, then a rotation about the new y-axis through an angle β, and then a rotation about the new z-axis through an angle γ, which we can parameterize through composing rotations using the Cayley-Klein parameters in the angle-axis representation. For the case of calculating spherical

harmonics and choosing $\chi = 0$, we have a rotation about the original y-axis through an angle θ then a rotation about the new z-axis through an angle ϕ. We represent each of these rotations individually by the Cayley-Klein parameters in the angle-axis representation.

$$R_{a\theta} = \cos\frac{\theta}{2}$$
$$R_{b\theta} = \sin\frac{\theta}{2}$$

$$R_{a\phi} = \cos\frac{\phi}{2} + i\sin\frac{\phi}{2}$$
$$R_{b\phi} = 0$$

To make sense of how to compose rotations represented by the Cayley-Klein parameters it is worthwhile to note that the Cayley-Klein parameters are the matrix elements of the SU(2) representation of rotation. So that the rotation (denoted by $\hat{R}$) can be represented as:

$$\hat{R} = \begin{pmatrix} R_a & R_b \\ -R_b^* & R_a^* \end{pmatrix}$$

Then when composing rotations

$$\hat{R} = \hat{R}_2\hat{R}_1$$

where $\hat{R}_1, \hat{R}_2$ are SU(2) matrices that represent arbitrary rotations. Performing the matrix multiplication we obtain the composition rule for rotations represented by the Cayley-Klein parameters.

$$\begin{aligned} R_a &= R_{a2}R_{a1} - R_{b2}R_{b1}^* \\ R_b &= R_{a2}R_{b1} + R_{b2}R_{a1}^* \end{aligned} \tag{B.17}$$

Then for the case of spherical harmonics the composition rule reduces to:

$$\begin{aligned} R_a &= R_{a\phi} R_{a\theta} = \left(\cos\frac{\phi}{2} + i\sin\frac{\phi}{2}\right)\cos\frac{\theta}{2} \\ R_b &= R_{a\phi} R_{b\theta} = \left(\cos\frac{\phi}{2} + i\sin\frac{\phi}{2}\right)\sin\frac{\theta}{2} \end{aligned} \tag{B.18}$$

Finally, using the composition rule we can then calculate the spherical harmonics using their relationship to the Wigner-D matrices.

$$Y_{lm}\left(R_a, R_b\right) = (-1)^m \sqrt{\frac{2l+1}{4\pi}} D^l_{0,-m}(R_a, R_b) \tag{B.19}$$

The radial inner product $\int_0^{r_{\text{cut}}} r^2 g_n(r) e^{-\alpha r^2} I_l(2\alpha r r_i) dr$ in Eq. 2.70 cannot be solved analytically so we employ numerical integration for this purpose. Chebyshev-Gauss quadrature is used so that the quadrature nodes for the interval $(0, r_{\text{cut}})$ never include $r = 0$ for any N number of nodes in the quadrature; the Chebyshev-Gauss quadrature nodes for the interval $(0, r_{\text{cut}})$ are given by:

$$x_i = \frac{r_{\text{cut}}}{2}\left[\cos\left(\frac{2i-1}{2N}\pi\right) + 1\right] \tag{B.20}$$

Avoiding the removable singularity at $r = 0$ due to I allows for the use of the following recursion relation to compute I at each of the nodes.

$$\begin{cases} I_0(x) & = \frac{\sinh(x)}{x} \\ I_1(x) & = \frac{x\cosh(x) - \sinh(x)}{x^2} \\ \quad \vdots & \\ I_n(x) & = I_{n-2}(x) - \frac{2n-1}{x} I_{n-1}(x) \end{cases} \tag{B.21}$$

The gradient of the smooth SO(3) power spectrum components then follows:

$$\nabla_i p_{nn'l} = \sum_{m=-l}^{+l} (c^*_{n'lm} \nabla_i c_{nlm} + c_{nlm} \nabla_i c^*_{n'lm}) \tag{B.22}$$

where the gradient of the expansion coefficients is obtained through the applying the product rule on Eq. 2.70 and then differentiating under the integral sign (as r_i is independent of r).

$$\begin{aligned}
&\nabla_i c_{nlm} = \\
&\quad 4\pi \nabla_i \left(e^{-\alpha r_i^2} \right) Y^*_{lm}(\hat{r}_i) \int_0^{r_{\text{cut}}} r^2 g_n(r) e^{-\alpha r^2} I_l(2\alpha r r_i) dr + \\
&\quad 4\pi e^{-\alpha r_i^2} \nabla_i \left(Y^*_{lm}(\hat{r}_i) \right) \int_0^{r_{\text{cut}}} r^2 g_n(r) e^{-\alpha r^2} I_l(2\alpha r r_i) dr + \\
&\quad 4\pi e^{-\alpha r_i^2} Y^*_{lm}(\hat{r}_i) \nabla_i \left(\int_0^{r_{\text{cut}}} r^2 g_n(r) e^{-\alpha r^2} I_l(2\alpha r r_i) dr \right)
\end{aligned} \tag{B.23}$$

$$\nabla_i \left(e^{-\alpha r_i^2} \right) = -2\alpha r_i e^{-\alpha r_i^2} \hat{r}_i$$

$$\begin{aligned}
&\nabla_i \left(\int_0^{r_{\text{cut}}} r^2 g_n(r) e^{-\alpha r^2} I_l(2\alpha r r_i) dr \right) = \\
&\qquad 2\alpha \int_0^{r_{\text{cut}}} r^3 g_n(r) e^{-\alpha r^2} I'_l(2\alpha r r_i) dr \hat{r}_i
\end{aligned}$$

We again evaluate the radial integral using Chebyshev-Gauss quadrature and use the following recursion relation for the evaluation of the first derivative of the modified spherical Bessel function.

$$I'_n(x) = \frac{1}{2n+1} [n I_{n-1}(x) + (n+1) I_{n+1}(x)].$$

Computing the gradient of the spherical harmonics is not as trivial as computing the gradient of the Wigner-D functions due to the singularities that exist at the north and south poles of the 2-sphere in Cartesian and spherical polar coordinates. Here we remove those singularities through taking the gradient with respect to the covariant spherical coordinates. The covariant spherical

coordinates are related to Cartesian coordinates by the following relation[164]:

$$
\begin{aligned}
x_{+1} &= -\frac{1}{\sqrt{2}}\left(x + iy\right) \\
x_0 &= z \\
x_{-1} &= \frac{1}{\sqrt{2}}\left(x - iy\right)
\end{aligned}
$$

Then, the gradient of the spherical harmonics with respect to the covariant spherical coordinates is given by[164]:

$$
\begin{aligned}
\nabla_0 Y_{l,m} = &-\frac{l}{r}\sqrt{\frac{(l+1)^2 - m^2}{(2l+1)(2l+3)}} \times Y_{l+1,m} \\
&-\frac{l+1}{r}\sqrt{\frac{l^2 - m^2}{(2l-1)(2l+1)}} \times Y_{l-1,m}
\end{aligned}
\tag{B.24}
$$

$$
\begin{aligned}
\nabla_{\pm 1} Y_{l,m} = &-\frac{l}{r}\sqrt{\frac{(l \pm m + 1)(l \pm m + 2)}{2(2l+1)(2l+3)}} \times Y_{l+1,m\pm 1} \\
&-\frac{l+1}{r}\sqrt{\frac{(l \mp m - 1)(l \mp m)}{2(2l-1)(2l+1)}} \times Y_{l-1,m\pm 1}
\end{aligned}
$$

So that we can obtain the gradient with respect to Cartesian coordinates by transforming the basis vectors back to Cartesian unit vectors[164].

$$
\begin{aligned}
e_x &= \frac{1}{\sqrt{2}}\left(e_{-1} - e_{+1}\right) \\
e_y &= \frac{i}{\sqrt{2}}\left(e_{-1} + e_{+1}\right) \\
e_z &= e_0
\end{aligned}
\tag{B.25}
$$

Appendix C Alternative Expression to compute D

We are aware that two previous works [87, 143] used a different approach to compute the Wigner-D martices [87, 143]. To start, a different set of Cayley-Klein parameters were used,

$$\begin{aligned} R_a &= \frac{1}{\sqrt{r^2 + r^2\cot^2(\omega/2)}}\left(r\cot(\omega/2) + iz\right) \\ R_b &= \frac{1}{\sqrt{r^2 + r^2\cot^2(\omega/2)}}\left(y + ix\right), \end{aligned} \tag{C.1}$$

which can be shown to be identically Eq. B.11. However, when implemented numerically, there exists a singularity at $\omega = 0$ and $\omega = 2\pi$, so we choose to implement Eq. B.11 rather than treating $\omega = 0$ as a separate case, and omitting $\omega = \pi$ altogether. Moreover, they used a recursive scheme to compute the D matrices,

$$\begin{cases} D^j_{mm'} = \sqrt{\frac{j-m}{j-m'}}R^*_a D^{j-1/2}_{m+1/2,m'+1/2} - \sqrt{\frac{j+m}{j-m'}}R^*_b D^{j-1/2}_{m-1/2,m'+1/2}, & m' \neq j \\ D^j_{mm'} = \sqrt{\frac{j-m}{j+m'}}R_b D^{j-1/2}_{m+1/2,m'-1/2} + \sqrt{\frac{j+m}{j+m'}}R_a D^{j-1/2}_{m-1/2,m'-1/2}, & m' \neq -j \end{cases} \tag{C.2}$$

Compared to the polynomial form Eq. B.12, the recursive form requires less floating point operations in general and is more efficient in serial calculations. However, in parallel architectures a polynomial form of the D-matrices is advantageous as no single term depends on another. During our implementation we found that using Numba's automatic parallelization [116], we were able to fuse all loops in the D-matrix calculation to achieve parallelization more so than algorithm when compared to the recursive version. This difference results in an improved scaling of the algorithm.

Bibliography

[1] K Philip and S. Chan, "Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection," in *Proceeding of the Fourth International Conference on Knowledge Discovery and Data Mining*, 1998, pp. 164–168.

[2] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to spam filtering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10 206–10 222, 2009.

[3] C.-L. Huang, M.-C. Chen, and C.-J. Wang, "Credit scoring with a data mining approach based on support vector machines," *Expert systems with applications*, vol. 33, no. 4, pp. 847–856, 2007.

[4] S.-s. Liu and Y.-t. Tian, "Facial expression recognition method based on gabor wavelet features and fractional power polynomial kernel pca," in *International Symposium on Neural Networks*, Springer, 2010, pp. 144–151.

[5] A. Waibel and K.-F. Lee, *Readings in speech recognition*. Elsevier, 1990.

[6] I. Portugal, P. Alencar, and D. Cowan, "The use of machine learning algorithms in recommender systems: A systematic review," *Expert Systems with Applications*, vol. 97, pp. 205–227, 2018.

[7] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in neural information processing systems*, 2014, pp. 487–495.

[8] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.

[9] J. Schmidt, M. R. Marques, S. Botti, and M. A. Marques, "Recent advances and applications of machine learning in solid-state materials science," *npj Computational Materials*, vol. 5, no. 1, pp. 1–36, 2019.

[10] K. Rajan, "Materials informatics," *Materials Today*, vol. 8, no. 10, pp. 38–45, 2005.

[11] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Physical review*, vol. 136, no. 3B, B864, 1964.

[12] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Physical review*, vol. 140, no. 4A, A1133, 1965.

[13] E. J. Heller, "Quantum localization and the rate of exploration of phase space," *Physical Review A*, vol. 35, no. 3, p. 1360, 1987.

[14] S. Piana, K. Lindorff-Larsen, and D. E. Shaw, "Protein folding kinetics and thermodynamics from atomistic simulation," *Proceedings of the National Academy of Sciences*, vol. 109, no. 44, pp. 17 845–17 850, 2012.

[15] *Materials genome initiative*, https://www.mgi.gov, Accessed: 2020-10-21.

[16] S. Curtarolo, G. L. Hart, M. B. Nardelli, N. Mingo, S. Sanvito, and O. Levy, "The high-throughput highway to computational materials design," *Nature materials*, vol. 12, no. 3, pp. 191–201, 2013.

[17] N. Nosengo *et al.*, "The material code," *Nature*, vol. 533, no. 7601, pp. 22–25, 2016.

[18] S. Curtarolo, W. Setyawan, G. L. Hart, M. Jahnatek, R. V. Chepulskii, R. H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, *et al.*, "Aflow: An automatic framework for high-throughput materials discovery," *Computational Materials Science*, vol. 58, pp. 218–226, 2012.

[19] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, *et al.*, "Commentary: The materials project: A materials genome approach to accelerating materials innovation," *Apl Materials*, vol. 1, no. 1, p. 011 002, 2013.

[20] J. E. Saal, S. Kirklin, M. Aykol, B. Meredig, and C. Wolverton, "Materials design and discovery with high-throughput density functional theory: The open quantum materials database (oqmd)," *Jom*, vol. 65, no. 11, pp. 1501–1509, 2013.

[21] V. Yamakov, D. Wolf, S. R. Phillpot, A. K. Mukherjee, and H. Gleiter, "Dislocation processes in the deformation of nanocrystalline aluminium by molecular-dynamics simulation," *Nat. Mater.*, vol. 1, no. 1, pp. 45–49, 2002.

[22] M Terrones, F Banhart, N Grobert, J.-C. Charlier, H Terrones, and P. Ajayan, "Molecular junctions by joining single-walled carbon nanotubes," *Phys. Rev. Lett.*, vol. 89, no. 7, p. 075 505, 2002.

[23] X. Li, Y. Wei, L. Lu, K. Lu, and H. Gao, "Dislocation nucleation governed softening and maximum strength in nano-twinned metals," *Nature*, vol. 464, no. 7290, pp. 877–880, 2010.

[24] M. S. Daw and M. I. Baskes, "Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals," *Phys. Rev. B*, vol. 29, no. 12, p. 6443, 1984.

[25] M. S. Daw, S. M. Foiles, and M. I. Baskes, "The embedded-atom method: A review of theory and applications," *Materials Science Reports*, vol. 9, no. 7-8, pp. 251–310, 1993.

[26] J Tersoff, "New empirical model for the structural properties of silicon," *Phys. Rev. Lett.*, vol. 56, no. 6, p. 632, 1986.

[27] F. H. Stillinger and T. A. Weber, "Computer simulation of local order in condensed phases of silicon," *Phys. Rev. B*, vol. 31, no. 8, p. 5262, 1985.

[28] A. D. MacKerell Jr, D. Bashford, M. Bellott, R. L. Dunbrack Jr, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H Guo, S. Ha, *et al.*, "All-atom empirical potential for molecular modeling and dynamics studies of proteins," *J. Phys. Chem. B*, vol. 102, no. 18, pp. 3586–3616, 1998.

[29] J. Behler, "Constructing high-dimensional neural network potentials: A tutorial review," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1032–1050, 2015.

[30] N. Artrith and J. Behler, "High-dimensional neural network potentials for metal surfaces: A prototype study for copper," *Phys. Rev. B*, vol. 85, p. 045 439, 4 2012.

[31] W. Li, Y. Ando, E. Minamitani, and S. Watanabe, "Study of Li atom diffusion in amorphous Li_3PO_4 with neural network potential," *J. Chem. Phys.*, vol. 147, no. 21, p. 214 106, 2017.

[32] A. P. Bartók, M. J. Gillan, F. R. Manby, and G. Csányi, "Machine-learning approach for one-and two-body corrections to density functional theory: Applications to molecular and condensed water," *Phys. Rev. B*, vol. 88, no. 5, p. 054 104, 2013.

[33] N. Artrith, T. Morawietz, and J. Behler, "High-dimensional neural-network potentials for multicomponent systems: Applications to zinc oxide," *Phys. Rev. B*, vol. 83, no. 15, p. 153 101, 2011.

[34] R. Z. Khaliullin, H. Eshet, T. D. Kühne, J. Behler, and M. Parrinello, "Nucleation mechanism for the direct graphite-to-diamond phase transition," *Nat. Mater.*, vol. 10, no. 9, p. 693, 2011.

[35] J. Behler, R. Martoňák, D. Donadio, and M. Parrinello, "Metadynamics simulations of the high-pressure phases of silicon employing a high-dimensional neural network potential," *Phys. Rev. Lett.*, vol. 100, no. 18, p. 185 501, 2008.

[36] A. P. Thompson, L. P. Swiler, C. R. Trott, S. M. Foiles, and G. J. Tucker, "Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials," *J. Comput. Phys.*, vol. 285, pp. 316–330, 2015.

[37] M. A. Wood and A. P. Thompson, "Extending the accuracy of the snap interatomic potential form," *J. Chem. Phys.*, vol. 148, no. 24, p. 241 721, 2018.

[38] S. Pozdnyakov, A. R. Oganov, A. Mazitov, T. Frolov, I. Kruglov, and E. Mazhnik, "Fast general two-and three-body interatomic potential," *arXiv preprint arXiv:1910.07513*, 2019. arXiv: 1910.07513 [physics.comp-ph].

[39] A. V. Shapeev, "Moment tensor potentials: A class of systematically improvable interatomic potentials," *Multiscale Model. Simul.*, vol. 14, no. 3, pp. 1153–1173, 2016.

[40] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, "Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons," *Phys. Rev. Lett.*, vol. 104, no. 13, p. 136 403, 2010.

[41] A. P. Bartók and G. Csányi, "G aussian approximation potentials: A brief tutorial introduction," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1051–1057, 2015.

[42] J. Behler and M. Parrinello, "Generalized neural-network representation of high-dimensional potential-energy surfaces," *Phys. Rev. Lett.*, vol. 98, no. 14, p. 146 401, 2007.

[43] R. G. Parr and W. Yang, *Density-Functional Theory of Atoms and Molecules (International Series of Monographs on Chemistry)*. Oxford University Press, USA, 1994, ISBN: 0195092767.

[44] P. A. Dirac, "Note on exchange phenomena in the thomas atom," in *Mathematical Proceedings of the Cambridge Philosophical Society*, Cambridge University Press, vol. 26, 1930, pp. 376–385.

[45] D. M. Ceperley and B. J. Alder, "Ground state of the electron gas by a stochastic method," *Physical Review Letters*, vol. 45, no. 7, p. 566, 1980.

[46] S. H. Vosko, L. Wilk, and M. Nusair, "Accurate spin-dependent electron liquid correlation energies for local spin density calculations: A critical analysis," *Canadian Journal of physics*, vol. 58, no. 8, pp. 1200–1211, 1980.

[47] J. P. Perdew and A. Zunger, "Self-interaction correction to density-functional approximations for many-electron systems," *Physical Review B*, vol. 23, no. 10, p. 5048, 1981.

[48] J. P. Perdew and Y. Wang, "Accurate and simple analytic representation of the electron-gas correlation energy," *Physical review B*, vol. 45, no. 23, p. 13 244, 1992.

[49] A. D. Becke, "Density-functional exchange-energy approximation with correct asymptotic behavior," *Physical review A*, vol. 38, no. 6, p. 3098, 1988.

[50] C. Lee, W. Yang, and R. G. Parr, "Development of the colle-salvetti correlation-energy formula into a functional of the electron density," *Physical review B*, vol. 37, no. 2, p. 785, 1988.

[51] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," *Physical review letters*, vol. 77, no. 18, p. 3865, 1996.

[52] N. Ashcroft and N. Mermin, *Solid State Physics*. Fort Worth: Saunders College Publishing, 1976.

[53] F. Jensen, *Introduction to computational chemistry*. John wiley & sons, 2017.

[54] D. J. Chadi and M. L. Cohen, "Special points in the brillouin zone," *Physical Review B*, vol. 8, no. 12, p. 5747, 1973.

[55] H. J. Monkhorst and J. D. Pack, "Special points for brillouin-zone integrations," *Physical review B*, vol. 13, no. 12, p. 5188, 1976.

[56] D. Hamann, M Schlüter, and C Chiang, "Norm-conserving pseudopotentials," *Physical Review Letters*, vol. 43, no. 20, p. 1494, 1979.

[57] N. Troullier and J. L. Martins, "Efficient pseudopotentials for plane-wave calculations," *Physical review B*, vol. 43, no. 3, p. 1993, 1991.

[58] D. Vanderbilt, "Soft self-consistent pseudopotentials in a generalized eigenvalue formalism," *Physical review B*, vol. 41, no. 11, p. 7892, 1990.

[59] P. E. Blöchl, "Projector augmented-wave method," *Phys. Rev. B*, vol. 50, no. 24, p. 17 953, 1994.

[60] G. Kresse and D. Joubert, "From ultrasoft pseudopotentials to the projector augmented-wave method," *Physical review b*, vol. 59, no. 3, p. 1758, 1999.

[61] R Nada, C. R. A. Catlow, R Dovesi, and V. Saunders, "An ab initio hartree–fock study of the ilmenite-structured mgsio3," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 436, no. 1898, pp. 499–509, 1992.

[62] M Catti, A. Pavese, E Apra, and C Roetti, "Quantum-mechanical hartree-fock study of calcite (caco 3) at variable pressure, and comparison with magnesite (mgco 3)," *Physics and Chemistry of Minerals*, vol. 20, no. 2, pp. 104–110, 1993.

[63] D. M. Sherman, "Equation of state and high-pressure phase transitions of stishovite (sio2): Ab initio (periodic hartree-fock) results," *Journal of Geophysical Research: Solid Earth*, vol. 98, no. B7, pp. 11 865–11 873, 1993.

[64] E. Tadmor and R. Miller, *Modeling materials: Continuum, atomistic and multiscale techniques*. Cambridge University Press, Jan. 2011, vol. 9780521856980, ISBN: 9780521856980.

[65] J. E. Jones, "On the determination of molecular fields.—i. from the variation of the viscosity of a gas with temperature," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 106, no. 738, pp. 441–462, 1924.

[66] ——, "On the determination of molecular fields.—ii. from the equation of state of a gas," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 106, no. 738, pp. 463–477, 1924.

[67] J. Tersoff, "New empirical approach for the structure and energy of covalent systems," *Physical review B*, vol. 37, no. 12, p. 6991, 1988.

[68] J Tersoff, "Empirical interatomic potential for carbon, with applications to amorphous carbon," *Physical Review Letters*, vol. 61, no. 25, p. 2879, 1988.

[69] J. Tersoff, "Modeling solid-state chemistry: Interatomic potentials for multicomponent systems," *Physical review B*, vol. 39, no. 8, p. 5566, 1989.

[70] M. S. Daw and M. I. Baskes, "Semiempirical, quantum mechanical calculation of hydrogen embrittlement in metals," *Phys. Rev. Lett.*, vol. 50, no. 17, p. 1285, 1983.

[71] M. S. Daw, "Model of metallic cohesion: The embedded-atom method," *Physical Review B*, vol. 39, no. 11, p. 7441, 1989.

[72] T. M. Mitchell, "Does machine learning really work?" *AI magazine*, vol. 18, no. 3, pp. 11–11, 1997.

[73] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[74] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, "Autonomous vehicles: Challenges, opportunities, and future implications for transportation policies," *Journal of modern transportation*, vol. 24, no. 4, pp. 284–303, 2016.

[75] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques," *arXiv preprint cs/0205070*, 2002.

[76] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[77] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[78] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[79] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in neural information processing systems*, 2012, pp. 341–349.

[80] J. Spiegelberg, J. C. Idrobo, A. Herklotz, T. Z. Ward, W. Zhou, and J. Rusz, "Local low rank denoising for enhanced atomic resolution imaging," *Ultramicroscopy*, vol. 187, pp. 34–42, 2018.

[81] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[82] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[83] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[84] J. Behler, "Atom-centered symmetry functions for constructing high-dimensional neural network potentials," *J. Chem. Phys.*, vol. 134, no. 7, p. 074 106, 2011.

[85] S. Hajinazar, J. Shao, and A. N. Kolmogorov, "Stratified construction of neural network based interatomic models for multicomponent materials," *Phys. Rev. B*, vol. 95, no. 1, p. 014 114, 2017.

[86] M. Gastegger and P. Marquetand, "High-dimensional neural network potentials for organic reactions and an improved training algorithm," *Journal of Chemical Theory and Computation*, vol. 11, no. 5, pp. 2187–2198, 2015.

[87] A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments," *Phys. Rev. B*, vol. 87, no. 18, p. 184 115, 2013.

[88] A. V. Shapeev, "Moment tensor potentials: A class of systematically improvable interatomic potentials," *Multiscale Modeling & Simulation*, vol. 14, no. 3, pp. 1153–1173, 2016.

[89] C. Chen, Z. Deng, R. Tran, H. Tang, I.-H. Chu, and S. P. Ong, "Accurate force field for molybdenum by machine learning large materials data," *Phys. Rev. Mater.*, vol. 1, no. 4, p. 043 603, 2017.

[90] X.-G. Li, C. Hu, C. Chen, Z. Deng, J. Luo, and S. P. Ong, "Quantum-accurate spectral neighbor analysis potential models for Ni-Mo binary alloys and fcc metals," *Phys. Rev. B*, vol. 98, p. 094 104, 9 2018.

[91] W. J. Szlachta, A. P. Bartók, and G. Csányi, "Accuracy and transferability of gaussian approximation potential models for tungsten," *Phys. Rev. B*, vol. 90, no. 10, p. 104 108, 2014.

[92] V. L. Deringer, D. M. Proserpio, G. Csányi, and C. J. Pickard, "Data-driven learning and prediction of inorganic crystal structures," *Faraday discussions*, vol. 211, pp. 45–59, 2018.

[93] V. L. Deringer, C. J. Pickard, and G. Csányi, "Data-driven learning of total and local energies in elemental boron," *Phys. Rev. Lett.*, vol. 120, no. 15, p. 156 001, 2018.

[94] E. V. Podryabinkin, E. V. Tikhonov, A. V. Shapeev, and A. R. Oganov, "Accelerating crystal structure prediction by machine-learning interatomic potentials with active learning," *Physical Review B*, vol. 99, no. 6, p. 064 114, 2019.

[95] L. Himanen, M. O. Jäger, E. V. Morooka, F. F. Canova], Y. S. Ranawat, D. Z. Gao, P. Rinke, and A. S. Foster, "Dscribe: Library of descriptors for machine learning in materials science," *Computer Physics Communications*, vol. 247, p. 106 949, 2020.

[96] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Physical review letters*, vol. 108, no. 5, p. 058 301, 2012.

[97] F. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento, "Crystal structure representations for machine learning models of formation energies," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1094–1101, 2015.

[98] H. Huo and M. Rupp, "Unified representation of molecules and crystals for machine learning," *arXiv preprint arXiv:1704.06439*, 2017.

[99] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld, "Machine learning of molecular electronic properties in chemical compound space," *New Journal of Physics*, vol. 15, no. 9, p. 095 003, 2013.

[100] M. Rupp, R. Ramakrishnan, and O. A. Von Lilienfeld, "Machine learning for quantum mechanical properties of atoms in molecules," *The Journal of Physical Chemistry Letters*, vol. 6, no. 16, pp. 3309–3313, 2015.

[101] J. E. Moussa, "Comment on "fast and accurate modeling of molecular atomization energies with machine learning"," *Physical review letters*, vol. 109, no. 5, p. 059 801, 2012.

[102] M. Gastegger, L. Schwiedrzik, M. Bittermann, F. Berzsenyi, and P. Marquetand, "Wacsf—weighted atom-centered symmetry functions as descriptors in machine learning potentials," *J. Chem. Phys.*, vol. 148, no. 24, p. 241 709, 2018.

[103] L. Zhang, J. Han, H. Wang, W. Saidi, R. Car, and E Weinan, "End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems," in *Advances in Neural Information Processing Systems*, 2018, pp. 4436–4446.

[104] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, "Schnet–a deep learning architecture for molecules and materials," *J. Chem. Phys.*, vol. 148, no. 24, p. 241 722, 2018.

[105] Y. Zhang, C. Hu, and B. Jiang, "Embedded atom neural network potentials: Efficient and accurate machine learning with a physically inspired representation," *J. Phys. Chem. Lett.*, vol. 10, no. 17, pp. 4962–4967, 2019.

[106] G. Imbalzano, A. Anelli, D. Giofré, S. Klees, J. Behler, and M. Ceriotti, "Automatic selection of atomic fingerprints and reference configurations for machine-learning potentials," *The Journal of Chemical Physics*, vol. 148, no. 24, p. 241 730, 2018.

[107] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[108] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[109] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[110] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[111] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, 3. MIT press Cambridge, MA, 2006, vol. 2.

[112] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[113] H. Yanxon, D. Zagaceta, B. Tang, D. Matteson, and Q. Zhu, "Pyxtal_ff: A python library for automated force field generation," *Machine Learning: Science and Technology*, 2020.

[114] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, *et al.*, "The atomic simulation environment—a python library for working with atoms," *Journal of Physics: Condensed Matter*, vol. 29, no. 27, p. 273 002, 2017.

[115] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The numpy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[116] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A llvm-based python jit compiler," in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 2015, pp. 1–6.

[117] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035.

[118] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[119] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[120] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[121] S. Plimpton, "Fast parallel algorithms for short-range molecular dynamics," *J. Comp. Phys.*, vol. 117, no. 1, pp. 1 –19, 1995.

[122] A Togo and I Tanaka, "First principles phonon calculations in materials science," *Scr. Mater.*, vol. 108, pp. 1–5, 2015.

[123] Y. Hinuma, G. Pizzi, Y. Kumagai, F. Oba, and I. Tanaka, "Band structure diagram paths based on crystallography," *Comput. Mater. Sci.*, vol. 128, pp. 140 –184, 2017.

[124] *Matscipy*, https://gitlab.com/libAtoms/matscipy.

[125] K. Lee, D. Yoo, W. Jeong, and S. Han, "Simple-nn: An efficient package for training and executing neural-network interatomic potentials," *Computer Physics Communications*, vol. 242, pp. 95–103, 2019.

[126] G. Kresse and J. Furthmüller, "Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set," *Phys. Rev. B*, vol. 54, pp. 11 169–11 186, 16 1996.

[127] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," *Phys. Rev. Lett.*, vol. 77, pp. 3865–3868, 18 1996.

[128] Y. Zuo, C. Chen, X. Li, Z. Deng, Y. Chen, J. Behler, G. Csányi, A. V. Shapeev, A. P. Thompson, M. A. Wood, *et al.*, "Performance and cost assessment of machine learning interatomic potentials," *J. Phys. Chem. A*, vol. 124, no. 4, pp. 731–745, 2020.

[129] S. Fredericks, D. Sayre, and Q. Zhu, "Pyxtal: A python library for crystal structure generation and symmetry analysis," 2019. arXiv: 1911.11123 [cond-mat.mtrl-sci].

[130] G. Kresse and J. Furthmüller, "Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set," *Phys. Rev. B*, vol. 54, no. 16, p. 11 169, 1996.

[131] V. L. Deringer and G. Csányi, "Machine learning based interatomic potential for amorphous carbon," *Phys. Rev. B*, vol. 95, p. 094 203, 9 2017.

[132] V. L. Deringer, C. J. Pickard, and G. Csányi, "Data-driven learning of total and local energies in elemental boron," *Phys. Rev. Lett.*, vol. 120, no. 15, p. 156 001, 2018.

[133] S.-D. Huang, C. Shang, P.-L. Kang, and Z.-P. Liu, "Atomic structure of boron resolved using machine learning and global sampling," *Chem. Sci.*, vol. 9, no. 46, pp. 8644–8655, 2018.

[134] S. Hajinazar, J. Shao, and A. N. Kolmogorov, "Stratified construction of neural network based interatomic models for multicomponent materials," *Phys. Rev. B*, vol. 95, p. 014 114, 1 2017.

[135] N. Kuritz, G. Gordon, and A. Natan, "Size and temperature transferability of direct and local deep neural networks for atomic forces," *Phys. Rev. B*, vol. 98, no. 9, p. 094 109, 2018.

[136] A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments," *Phys. Rev. B*, vol. 87, p. 184 115, 18 2013.

[137] J. D. Schall, G. Gao, and J. A. Harrison, "Elastic constants of silicon materials calculated as a function of temperature using a parametrization of the second-generation reactive empirical bond-order potential," *Phys. Rev. B*, vol. 77, no. 11, p. 115 209, 2008.

[138] A. P. Bartók, J. Kermode, N. Bernstein, and G. Csányi, "Machine learning a general-purpose interatomic potential for silicon," *Phys. Rev. X*, vol. 8, p. 041 048, 4 2018.

[139] A. O. Lyakhov, A. R. Oganov, H. T. Stokes, and Q. Zhu, "New developments in evolutionary structure prediction algorithm uspex," *Comput. Phys. Commun.*, vol. 184, no. 4, pp. 1172–1182, 2013.

[140] H. Niu, L. Bonati, P. M. Piaggi, and M. Parrinello, "Ab initio phase diagram and nucleation of gallium," *Nature Communications*, vol. 11, no. 1, pp. 1–9, 2020.

[141] K. T. Schutt, P. Kessel, M. Gastegger, K. A. Nicoli, A. Tkatchenko, and K.-R. Muller, "Schnetpack: A deep learning toolbox for atomistic systems," *J. Chem. Theory Comput.*, vol. 15, no. 1, pp. 448–455, 2019.

[142] C. Chen, Z. Deng, R. Tran, H. Tang, I.-H. Chu, and S. P. Ong, "Accurate force field for molybdenum by machine learning large materials data," *Phys. Rev. Materials*, vol. 1, p. 43 603, 2017.

[143] A. P. Thompson, L. P. Swiler, C. R. Trott, S. M. Foiles, and G. J. Tucker, "Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials," *J. Comp. Phys.*, vol. 285, pp. 316–330, 2015. eprint: 1409.3880.

[144] J.-W. Yeh, S.-K. Chen, S.-J. Lin, J.-Y. Gan, T.-S. Chin, T.-T. Shun, C.-H. Tsau, and S.-Y. Chang, "Nanostructured high-entropy alloys with multiple principal elements: Novel alloy design concepts and outcomes," *Advanced Engineering Materials*, vol. 6, no. 5, pp. 299–303, 2004.

[145] O. N. Senkov, G. Wilks, J. Scott, and D. B. Miracle, "Mechanical properties of $Nb_{25}Mo_{25}Ta_{25}W_{25}$ and $V_{20}Nb_{20}Mo_{20}Ta_{20}W_{20}$ refractory high entropy alloys," *Intermetallics*, vol. 19, no. 5, pp. 698–706, 2011.

[146] X.-G. Li, C. Chen, H. Zheng, and S. P. Ong, "Unravelling complex strengthening mechanisms in the NbMoTaW multi-principal element alloy with machine learning potentials," *arXiv preprint arXiv:1912.01789*, 2019.

[147] P. Haas, F. Tran, and P. Blaha, "Calculation of the lattice constant of solids with semilocal functionals," *Physical Review B*, vol. 79, no. 8, p. 085 104, 2009.

[148] G. Mills and H. Jónsson, "Quantum and thermal effects in h 2 dissociative adsorption: Evaluation of free energy barriers in multidimensional quantum systems," *Physical review letters*, vol. 72, no. 7, p. 1124, 1994.

[149] G. Henkelman and H. Jónsson, "Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points," *The Journal of chemical physics*, vol. 113, no. 22, pp. 9978–9985, 2000.

[150] G. Henkelman, B. P. Uberuaga, and H. Jónsson, "A climbing image nudged elastic band method for finding saddle points and minimum energy paths," *The Journal of chemical physics*, vol. 113, no. 22, pp. 9901–9904, 2000.

[151] W Windl, M. Bunea, R Stumpf, S. Dunham, and M. Masquelier, "First-principles study of boron diffusion in silicon," *Physical review letters*, vol. 83, no. 21, p. 4345, 1999.

[152] R. Stumpf, C.-L. Liu, and C. Tracy, "Retardation of o diffusion through polycrystalline pt by be doping," *Physical Review B*, vol. 59, no. 24, p. 16 047, 1999.

[153] B. P. Uberuaga, M. Leskovar, A. P. Smith, H. Jónsson, and M. Olmstead, "Diffusion of ge below the si (100) surface: Theory and experiment," *Physical review letters*, vol. 84, no. 11, p. 2441, 2000.

[154] M. Villarba and H. Jónsson, "Diffusion mechanisms relevant to metal crystal growth: Pt/pt (111)," *Surface science*, vol. 317, no. 1-2, pp. 15–36, 1994.

[155] ——, "Atomic exchange processes in sputter deposition of pt on pt (111)," *Surface science*, vol. 324, no. 1, pp. 35–46, 1995.

[156] T. Rasmussen, K. W. Jacobsen, T. Leffers, O. B. Pedersen, S. Srinivasan, and H. Jonsson, "Atomistic determination of cross-slip pathway and energetics," *Physical review letters*, vol. 79, no. 19, p. 3676, 1997.

[157] P. J. Feibelman, J. Nelson, and G. Kellogg, "Energetics of pt adsorption on pt (111)," *Physical Review B*, vol. 49, no. 15, p. 10 548, 1994.

[158] G. Boisvert, L. J. Lewis, and M. Scheffler, "Island morphology and adatom self-diffusion on pt (111)," *Physical Review B*, vol. 57, no. 3, p. 1881, 1998.

[159] A. Okiji, H. Kasai, and K. Makoshi, *Elementary Processes in Excitations and Reactions on Solid Surfaces: Proceedings of the 18th Taniguchi Symposium Kashikojima, Japan, January 22–27, 1996*. Springer Science & Business Media, 2012, vol. 121.

[160] J. Buolamwini and T. Gebru, "Gender shades: Intersectional accuracy disparities in commercial gender classification," in *Conference on fairness, accountability and transparency*, 2018, pp. 77–91.

[161] G. Sivaraman, A. N. Krishnamoorthy, M. Baur, C. Holm, M. Stan, G. Csányi, C. Benmore, and Á. Vázquez-Mayagoitia, "Machine-learned interatomic potentials by active learning: Amorphous and liquid hafnium dioxide," *npj Computational Materials*, vol. 6, no. 1, pp. 1–8, 2020.

[162] M. K. Bisbo and B. Hammer, "Efficient global structure optimization with a machine-learned surrogate model," *Physical Review Letters*, vol. 124, no. 8, p. 086 102, 2020.

[163] M. Boyle, "Angular velocity of gravitational radiation from precessing binaries and the corotating frame," *Phys. Rev. D*, vol. 87, p. 104 006, 10 2013.

[164] D. A. Varshalovich, A. N. Moskalev, and V. K. Khersonskii, *Quantum Theory of Angular Momentum*. Singapore: World Scientific, 1988.

Curriculum Vitae

Howard Yanxon

University of Nevada, Las Vegas
Department of Physics and Astronomy
Las Vegas, NV, 89154

E-mail: hg.yanxon@gmail.com
https://orcid.org/0000-0002-5078-1074

Education

Aug '16 – Dec '20: **PhD in Physics**, University of Nevada, Las Vegas
Jan '13 – May '15: **BS in Physics**, University of Nevada, Las Vegas

Professional Experience

Jan '18 – present: **PhD Candidate**
Dr. Qiang Zhu
Physics and Astronomy Department, University of Nevada, Las Vegas

Jun '19 – Jan '20: **Internship**
Dr. Brandon Wood
Lawrence Livermore National Laboratory

Aug '16 – May '19: **Graduate Teaching Assistant**
Physics and Astronomy Department, University of Nevada, Las Vegas

Aug '18 – Sep '18: **Internship**
Dr. Brandon Wood
Lawrence Livermore National Laboratory

Jan '15 – Dec '17: **Visiting Scholar**
Dr. Andrew Cornelius and Dr. Ravhi Kumar
Argonne National Laboratory

Jan '16 – Dec '17: **Graduate Researcher**
Dr. Andrew Cornelius and Dr. Ravhi Kumar
Physics and Astronomy Department, University of Nevada, Las Vegas

May '17 – Aug '17: **Internship**
Dr. Krzysztof Gofryk
Idaho National Laboratory

Code Contributions

PyXtal_FF - A Python package for developing machine learning of interatomic force field. Source: https://github.com/qzhu2017/PyXtal_FF

PyXtal_ml - A Python package for machine learning modelling of materials properties. Source: https://github.com/qzhu2017/PyXtal_ml

Publications

1. **Yanxon, H.**; Zagaceta, D.; Tang, B.; Matteson, D.; Zhu, Q. *PyXtal_FF: a Python Library for Automated Force Field Generation.* Machine Learning: Science and Technology, 2020.

2. **Yanxon, H.**; Zagaceta, D.; Wood, B. C.; Zhu, Q. *Neural network potential from bispectrum components: A case study on crystalline silicon.* The Journal of Chemical Physics, 153(5), 054118, 2020.

3. Zagaceta, D.; **Yanxon, H.**; Zhu, Q. *Spectral neural network potentials for binary alloys.* Journal of Applied Physics, 128(4), 045113, 2020.

4. Mkrtchyan, V.; Kumar, R.; White, M.; **Yanxon, H.**; Cornelius, A. *Effect of pressure on crystal structure and superconductivity of* $NbSe_xTe_{2-x}$ *(x=2, 1.5)* . Chemical Physics Letters, 692, 249-252, 2018.

Funding/Scholarships

Science Graduate Student Research (SCGSR) Program
UNLV GPSA Funding
The Donna Weistrop and David B. Shaffer Scholarship
The McNair Post-Baccalaureate Scholarship
Russell L. Brenda Frank Endowed Scholarship

Conferences/Workshops

2020 **Virtual**: XSEDE HPC Workshop Summer Boot Camp
2019 **Albuquerque, USA**: LAMMPS Workshop and Symposium
2019 **Las Vegas, USA**: GPSA Research Forum
2018 **Honolulu, USA**: Workshop of the IUCr Commission on High Pressure

2018 **Holderness, USA**: Gordon Research Conference
2017 **Beijing, China**: The 26th International Conference on High Pressure Science and Tech.
2017 **Naperville, USA**: Stewardship Science Academic Programs (SSAP) Symposium

References

Qiang Zhu: **PhD Advisor**
Phone: (702) 895-1707
E-mail: qiang.zhu@unlv.edu

Ashkan Salamat: **Professor**
Phone: (702) 895-1716
E-mail: salamat@physics.unlv.edu

Bernard Zygelman: **Professor**
Phone: (702) 895-1321
E-mail: bernard@physics.unlv.edu

Brandon Wood: **Staff Scientist**
Phone: (925) 422-8391
E-mail: brandonwood@llnl.gov