MASARYK
UNIVERSITY

FACULTY OF SCIENCE

Machine learning-assisted atomistic modeling of amorphous materials

Bachelor's Thesis

TOMÁŠ ROTTENBERG

Brno, Spring 2023

MASARYK
UNIVERSITY

FACULTY OF SCIENCE

Machine learning-assisted atomistic modeling of amorphous materials

Bachelor's Thesis

TOMÁŠ ROTTENBERG

Advisor: Mgr. Pavel Ondračka, Ph.D.

Department of Physical Electronics

Brno, Spring 2023

UNIVERSITAS MASARYKIANA BRUNENSIS • FACULTAS RERUM NATURALIUM •

SCIENTIA EST POTENTIA

Bibliographic record

Author:	Tomáš Rottenberg Faculty of Science Masaryk University Department of Physical Electronics
Title of Thesis:	Machine learning-assisted atomistic modeling of amorphous materials
Degree Programme:	Nanotechnology
Field of Study:	Physics
Supervisor:	Mgr. Pavel Ondračka, Ph.D.
Academic Year:	2022/2023
Number of Pages:	9 + 41
Keywords:	machine learning, computational physics, atomistic modeling, molecular dynamics, neural networks, DeePMD

Abstract

We explore the methodology of machine learning-assisted molecular dynamics simulations for quantum chemistry and solid state physics and demonstrate that by using the DeePMD method, it is possible to achieve linear scaling with system size while maintaining computational accuracy comparable to that of *ab initio* calculations. To show this, we use the Deep Potential Molecular Dynamics scheme to calculate various material properties, such as relaxation energy, relaxation volume, and bulk modulus, for select atomic structures and equate the precision of our results with their *ab initio* counterparts. The implementation of our approach uses the DeePMD-kit software suite with the LAMMPS molecular dynamics simulator and leverages active learning to train deep potential neural networks.

Acknowledgements

These are the acknowledgements for my thesis, which can span multiple paragraphs.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Tomáš Rottenberg

Contents

List of Tables

List of Figures

1 Introduction

Molecular dynamics (MD) is a computer simulation method for calculating the physical movements of atoms and molecules. MD simulations work by numerically solving Newton's equations to obtain the trajectories of atoms and molecules. The forces acting on the simulated particles, along with their potential energies, are calculated using interatomic potentials or molecular mechanical force fields. MD is an invaluable tool used in many disciplines, including physics, chemistry, biology, and material science, as it enables running complex experiments virtually, examining their results on atomary scales, which might be unreachable even with costly laboratory equipment.

The accuracy of MD simulations is largely dependent on its model for atomic interactions. There are two prominent approaches to modeling these interactions. The *ab initio* calculations determine the electronic structure of a system from first principles using a quantum mechanical theory. *Ab initio* molecular dynamics (AIMD) is considered to be highly accurate, however, due to treating the electronic degrees of freedom, its computational burden is significant, and its usage is generally limited to smaller systems with hundreads of atoms and time scales of $\sim 100\,\mathrm{ps}$. Applications requiring larger cells and longer simulations are currently accessible only with empirical interatomic potentials, also called force fields (FFs). These models, while having much lower computational requirements, suffer from decreased precision and are not very transferable.

In recent years, there's been a steady surge of breakthrough applications of machine learning (ML) and neural networks. From beating human experts in games like chess and go to almost perfectly solving the problem of protein folding, artificial intelligence seems to be an ideal fit for finding stochastic heuristics and patterns in complicated problems, given a large dataset of learning data. This wave of transition toward machine learning-based algorithms has affected even MD software packages and given rise to a very successful class of machine learning potentials and force fields.

The goal of my thesis is to evaluate these machine learning potentials and demonstrate that they are capable of producing results that are on par with quantum mechanical methods while scaling in a linear manner with the size of a system. My application uses the DeePMD-kit code, which is an implementation of the Deep Potential Molecular Dynamics (DeePMD) protocol. Furthermore, the LAMMPS Molecular Dynamics Simulator was used to run the machine learning potentials on concrete molecular dynamics ensembles. Last but not least, the OpenMX simulator was used to provide quantum mechanical data for training and evaluating the neural network models.

2 Molecular Dynamics

Molecular dynamics simulations is an umbrella term for a class of computational methods used to model and analyze the physical behavior of systems of atoms and molecules. MD allows one to monitor the full time evolution of a system, allowing for deep examination of the dynamics of atomic-level phenomena that cannot be observed directly. Computer simulations applied to condensed matter systems began their development as early as the 1950s, when two of the pillars of molecular simulation were introduced, namely the Monte Carlo (MC) sampling technique and the molecular dynamics method. In 1964, the first realistic MD simulation was developed by Rahman, who came up with a realistic model of liquid Argon. Rahman used the Lennard–Jones pair-wise additive potential and showed, that MD simulations with smooth potentials were possible.

Around the same time, Verlet proposed a stable numerical integration algorithm that is still very popular in modern MD software. We will discuss the Verlet integration algorithm in further chapters of this thesis. Verlet also invented a time-saving algorithm, the Verlet neighbour list.

A great leap forward in the MD methology happened in 1971, when Rahman and Stillinger published an MD study on modeling a realistic system of liquid water, a system composed of molecules, not just individual atoms. The significant results of their work prompted a multinational group of scientists centred around Berendsen at CECAM to try using MD simulations for examining biomolecules. The first MD simulation of a simple protein was due to Karplus and collaborators, and appeared shortly after, in 1977. In 2013, the Nobel Prize for Chemistry was awarded to Warshel, Levitt, and Karplus for their work on computer simulations in biochemistry, which was built upon the efforts of many researchers who had previously worked on simulating biomolecules.

Another inportant development took place in 1980. In this year, Anderson published a paper that described how to extend MD to enable it to sample the isoenthalpic (constant pressure) ensemble. The standard molecular dynamics algorithm was designed to simulate the behavior of a system of particles at constant energy, or in the microcanonical ensemble, because the Newton's equations of motion conserve energy. It was not straightforward to modify the MD algorithm to sample systems under different, more experimentally relevant conditions. Andersen's extensions for sampling the isoenthalpic ensemble inspired the question of whether it was possible to use MD to sample the canonical ensemble as well. Nosé, building on Andersen's work, introduced a new variable that linked the kinetic energy of the atoms to the external temperature, resulting in dynamics that sample the desired ensemble. This approach is known as the Nosé–Hoover thermostat, which is often used in a modified form called the Hoover thermostat.

In 1985, Car and Parrinello published a groundbreaking paper in Physical Review Letters that described a method for combining MD with density functional theory (DFT) calculations of electronic structure. This approach eliminated the need for a potential model, as energy, forces, and stress could be calculated directly from the electronic structure. The Car–Parrinello method allowed for the simulation of processes that involve bond formation or breaking and was the first to demonstrate that it is possible to combine finite temperature simulations with ground-state electronic structure calculations. This method also served as a bridge between the simulation community, which typically has a background in statistical mechanics, and the solid-state physics and quantum chemistry communities, which focus on electronic structure calculations at zero temperature.

During the 1980s and 1990s, the use of molecular simulations in condensed matter research became more widespread, due in part to previous successes in this field and also to the increasing availability and power of computers.

Modern MD methodology is frequently used to refine the three-dimensional structures of proteins and other large molecules, to study atomic-level phenomena that cannot be directly observed, such as thin-film growth and ion implantation, and to investigate the physical properties of nanotechnological devices that cannot yet be manufactured. In 2015, for example, MD simulation has been reported for pharmacophore development and drug design.

2.1 Classical Methods

The classical MD implementation uses the so-called "ball and sticks" model, where atoms and molecules are treated as soft balls and their bonds are represented by elastic sticks. The laws of classical mechanics define the dynamics of the entire system.

Each particle in an MD simulation has its own position vector $\mathbf{r}_i(t) = (x_i(t), y_i(t), z_i(t))$. A particle usually corresponds to an atom, although it may represent any simulable entity of interest that can be conveniently described by an interaction law. By Newton's second law the motion of each particle must obey the following relation

$$\mathbf{F}_i = m_i \frac{\mathrm{d}^2 \mathbf{r}_i}{\mathrm{d}t^2}, \tag{2.1}$$

where m_i is the mass of i-th particle and $\mathbf{F}_i$ is the force acting upon i-th particle. Interaction laws are usually specified by a potential function $U(\mathbf{r}_1, \dots, \mathbf{r}_N)$, which represents the potential energy of N interacting particles as a function of their positions. Given the potential, the force acting upon i-th atom is determined by the gradient with respect to particle displacements

$$\mathbf{F}_i = -\nabla_{\mathbf{r}_i} U(\mathbf{r}_1, \dots, \mathbf{r}_N) = -\left(\frac{\partial U}{\partial x_i}, \frac{\partial U}{\partial y_i}, \frac{\partial U}{\partial z_i}\right). \tag{2.2}$$

Let's briefly talk about the meaning and form of the potential U in MD simulations. Any quantum-chemistry textbook would insist, that in order to appropriately examine a behavior of molecule, we can't just look at its individual atoms. The quantum-mechanical lense reveals, that when atoms bond into molecules, their electron orbitals interact in complex ways, giving rise to non-trivial molecule orbitals. These electronic clouds that span multiple atoms then determine molecule's interactions with other particles. This paints molecules as a very complicated quantum systems, where electrons and nuclei are interacting together in an intricate manner. It turns out, however, that to a very good approximation, known as the Born–Oppenheimer adiabatic approximation and based on the difference in mass between nuclei and electrons, the electronic and nuclear problems can be separated. According to this approximation, we can presume that the electron clouds equilibrate quickly for each instanteneous configuration of the heavy nuclei. The nuclei then move in the field created by the average electron densities. This allows us to consider the concept of a potential energy surface, which controls the movement of the nuclei without taking explicit account of the electrons. Given the potential energy surface, we may use classical mechanics to follow the dynamics of the nuclei. Rather than solving the quantum-mechanical problem, we can solve a classic-mechanical problem, in which the effect of the electrons on nuclei is expressed by en empirical potential. It can be very challenging to identify a potential function that accurately represents an energy surfaces of a system, but doing so greatly simplifies the computational process. Atomic force field models and the classical MD are based on empirical potentials with a specific functional form, representing the physics and chemistry of the systems of interest. The following equation is an example of such a force field, used in biosystem simulations

$$\begin{aligned} U(\mathbf{r}_1,\dots,\mathbf{r}_N) = &\sum_{\text{bonds}} \frac{a_i}{2}(l_i - l_{i0})^2 + \sum_{\text{angles}} \frac{b_i}{2}(\theta_i - \theta_{i0})^2 \\ &+ \sum_{\text{torsions}} \frac{c_i}{2}\left[1 + \cos(n\omega_i - \gamma_i)\right] \\ &+ \sum_{\text{atom pairs}} 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] \\ &+ \sum_{\text{atom pairs}} k\frac{q_i q_j}{r_{ij}}. \end{aligned} \tag{2.3}$$

The covalent character of the system is defined by the first three terms of the system, where the summation indices run over all the bonds, angles and torsions. In contrast, the last two terms are only defined by atom pairs, with $q_i q_j$ being the product of their charges and $r_{ij} = |r_i - r_j|$ is the distance between the two atoms in the pair. The first two terms give energies of deformations of the bond lengths l_i and bond angles θ_i from their respective equilibrium values l_{i0} and θ_{i0} with force constants a_i and b_i. These two terms model the correct chemical structure, but prevent more complicated chemical phenomena like bond breaking. Rotations around the chemical bond are described by the third therm, which is priodic with periodicity determined by n and heights of rotational barriers defined by c_i. The forth term represents the van der Waals repulsive and attractive interatomic forces in the form of the Lennard–Jones 12-6 potential. The last term is the Coulomb electrostatic potential. Some effects due to specific environments can be accounted for by properly adjusting partial charges q_i and effective value of the constant k as well as the van der Waals parameters ε_{ij} and σ_{ij}.

We now have a full mathematical description of the problem at hand. Due to the many-body nature of the problem, it is out of question to solve it analytically, thus it has to be discretized and solved numerically with a computer. First, we need to specify the initial conditions of the system, that is, the initial positions $\mathbf{r}_{i0}$ and initial velocities $\mathbf{v}_{i0}$ of the particles in the system. Then we have to use a numerical integrator to continually make finite time interval steps and find the successive values of positions $\mathbf{r}_i(t)$ and velocities $\mathbf{v}_i(t)$ in the time evolution of the system. A very popular numerical integration algorithm is the Verlet algorithm, which gained its popularity for MD simulations mainly due to its simplicity and stability. The fundamental equation for this algorithm can be obtained from the Taylor series expansions of the position vector $\mathbf{r}_i(t)$; the full equation reads

$$\mathbf{r}_i(t+\Delta t) \doteq 2\mathbf{r}_i(t) - \mathbf{r}_i(t-\Delta t) + \frac{\mathbf{F}_i(t)}{m_i}\Delta t^2 \tag{2.4}$$

Equation (2.4) is accurate up to terms of the fourth power in Δt. Velocities can be calculated from the positions or using alternative leapfrog or velocity Verlet schemes. Precise trajectories correspond to the scenario where the integration step size approaches infinitesimally small values. However, it is preferable to utilize larger time steps in order to sample larger trajectories. In practical applications, the time step Δt is dictated by the fast motions present within the simulation.

2.2 *Ab Initio* Methods

In classical molecular dynamics, the forces acting on the atoms are derived from a potential energy function, which is usually based on empirical or semi-empirical force fields. These force fields are parameterized using experimental data or quantum mechanical calculations, and they provide an approximate description of the interactions between atoms. Classical molecular dynamics is generally not very accurate because it relies on approximations and empirical parameters and can't account for dynamical events and quantum effects such as changes in chemical bonding, the presence of important noncovalent intermediates and tunnelling of protons or electrons. The changes in bonding and the existence of intermediates can be accounted for using first-principles (or *ab initio*) MD. *Ab initio* molecular dynamics (AIMD) is generally more accurate than classical molecular dynamics because it directly calculates the electronic structure of the system. This makes it suitable for studying systems where electronic structure plays a critical role. AIMD is much more computationally expensive, which limits its applicability to relatively small systems and short timescales. The forces and electronic structure are determined by solving the Schrödinger equation, typically using methods from quantum physics, such as the Hartree-Fock method, density functional theory (DFT), or other wavefunction-based methods.

2.3 Density Functional Theory

Over the last three decades, density functional theory has emerged as the leading approach for conducting quantum mechanical simulations of periodic systems. Recently, it has gained popularity among quantum chemists and has become extensively employed for simulating energy surfaces in molecular systems. While its origins can be traced back to the early 1930s, DFT was formally established in 1964 through two theorems developed by Hohenberg and Kohn.

Simulating an atomic system from first principles means solving the quantum many-body problem. If we have N nuclei, we are dealing with a problem of $N + ZN$ electromagnetically interacting particles.

The Hamiltonian of such problem is

$$\begin{aligned}\hat{H} = &-\frac{\hbar^2}{2}\sum_i \frac{\nabla^2_{\mathbf{R}_i}}{M_i} - \frac{\hbar^2}{2}\sum_i \frac{\nabla^2_{\mathbf{r}_i}}{m_e} \\ &- \frac{1}{4\pi\varepsilon_0}\sum_{ij}\frac{e^2 Z_i}{|\mathbf{R}_i - \mathbf{r}_j|} + \frac{1}{8\pi\varepsilon_0}\sum_{i\neq j}\frac{e^2}{|\mathbf{r}_i - \mathbf{r}_j|} \\ &+ \frac{1}{8\pi\varepsilon_0}\sum_{i\neq j}\frac{e^2 Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|}.\end{aligned} \tag{2.5}$$

The mass of the nucleus at $\mathbf{R}_i$ is M_i, the electrons hamve mass m_e and are at $\mathbf{r}_i$. The first term is the kinetic energy operator for the nuclei, the second for the electrons. The last three terms describe the Coulomb interaction between electrons and nuclei, between electrons and other electrons, and between nuclei and other nuclei. It is out of question to solve this problem exactly, therefore we have to introduce some approximations, leadin to the formulation of DFT.

Firstly, we observe that the nuclei are much heavier and consequently move much slower than the electrons. This observation justifies rewriting the Hamiltonian such that

$$\hat{H} = \hat{T} + \hat{V} + \hat{V}_{\text{ext}}. \tag{2.6}$$

Now $\hat{T}$ represents only the kinetic energy of the electron gas, as this approximation assumes the nuclei are effectively frozen at their positions. The term $\hat{V}$ now represents the electron-electron interaction only. The last term $\hat{V}_{\text{ext}}$ is due to the potential energy of the electrons in the (now external) potential of the nuclei. System-specific information is now entirely given by the term $\hat{V}_{\text{ext}}$. This approximation is called the The Born–Oppenheimer approximation.

The quantum many body problem obtained after the first approximation is much simpler than the original one, but still far too difficult to solve. To reduce the equation (2.6) further, we introduce the two Hohenberg—Kohn theorems.

1. *There is a one-to-one correspondence between the ground-state density $\rho(\mathbf{r})$ of a many-electron system (atom, molecule, solid) and the external potential V_{ext}. An immediate consequence is that the ground-state*

expectation value of any observable $\hat{O}$ is a unique functional of the exact ground-state electron density

$$\langle\Psi|\hat{O}|\Psi\rangle = O[\rho]. \tag{2.7}$$

2. *For $\hat{O}$ being the hamiltonian $\hat{H}$, the ground-state total energy functional $H[\rho] \equiv E_{V_{\text{ext}}}[\rho]$ is of the form*

$$\begin{aligned} E_{V_{\text{ext}}}[\rho] &= \langle\Psi|\hat{T}+\hat{V}|\Psi\rangle + \langle\Psi|\hat{V}_{\text{ext}}|\Psi\rangle \\ &= F_{HK}[\rho] + \int \rho(\mathbf{r})V_{\text{ext}}(\mathbf{r})\,\mathrm{d}\mathbf{r}, \end{aligned} \tag{2.8}$$

where the Hohenberg-Kohn density functional $F_{HK}[\rho]$ is universal for any many-electron system. $E_{V_{\text{ext}}}[\rho]$ reaches its minimal value (equal to the ground-state total energy) for the ground-state density corresponding to V_{ext}.

Let's pause and ponder the meaning of these theorems for a bit. Each many-electron system has a unique potential, and each system as a whole is represented by a unique ground-state many particle wave function, as governed by the hamiltonian (2.6) and the Schrödinger equation. From this wavefunction, we can obtain the corresponding electron density using the density operator $\hat{\rho}$ which is defined as

$$\hat{\rho}(\mathbf{r}) = \sum_i^N \delta(\mathbf{r}_i - \mathbf{r}). \tag{2.9}$$

The first theorem of Hohenberg—Kohn tells us, that there exists a one-to-one correspondence between the ground-state density and the external potential. The density contains as much information as the wave function does, meaning all observables can be retrieved from the density only and can be written as functionals of the density.

The equation suggests, that the energy contribution from the external potential can easily be calculated, given the density function $\rho(\mathbf{r})$. An explicit expression for F_{HK} is not known, however, since the term is not dependent on the information of the nuclei and their position, it is a universal functional for any many-electron system and can be found. Moreover, the second theorem states that the ground-state density ρ corresponding to the external potential $V_{\text{ext}}(\mathbf{r})$ minimizes the total energy functional $E_{V_{\text{ext}}}[\rho]$, which allows us to use the variational principle of Rayleigh–Ritz in order to find the ground-state density.

Noting once again, that the the Hohenberg–Kohn functional F_{HK} is only dependent on the electron structur and not on the structure of the nuclei, we can now proceed with finding an expression for F_{HK}. There exist several options for defining the energy of the electron many-body system. The exact solution is simply

$$E_e = T + V, \tag{2.10}$$

where T and V are the exact kinetic and electron-electron potential energy functionals. The Hartree–Fock solution is defined as

$$E_{HF} = T_0 + \underbrace{(V_H + V_x)}_{V}, \tag{2.11}$$

where T_0 is the functional for the kinetic energy of a non-interacting electron gas, V_H stands for Hartree contribution and V_x for exchange contribution. There's also the Hartree solution, which lacks the exchange contribution

$$E_H = T0 + V_H. \tag{2.12}$$

We use this solution as a definition for the exchange energy

$$V_x = E_{HF} - E_H = V - V_H. \tag{2.13}$$

We define the correlation energy as the difference between the exact solution and the Hartree–Fock solution

$$V_c = E_e - E_{HF} = T - T_0. \tag{2.14}$$

We can now rewrite the Hohenberg–Kohn functional

$$\begin{aligned} F_{HK} &= T + V + T_0 - T_0 \\ &= T_0 + V + \underbrace{(T - T_0)}_{V_c} \\ &= T_0 + V + V_c + V_H - V_H \\ &= T_0 + V_H + V_c + \underbrace{(V - V_H)}_{V_x} \\ &= T_0 + V_H + \underbrace{(V_x + V_c)}_{V_{xc}}. \end{aligned} \tag{2.15}$$

The term V_{xc} is called the exchange–correlation energy functional. We can write the energy functional explicitly as

$$E_{V_{\text{ext}}}[\rho] = T_0[\rho] + V_H[\rho] + V_{xc}[\rho] + V_{ext}[\rho]. \tag{2.16}$$

The rewritten energy functional can be interpreted as an energy functional of a non-interacting classical electron gas, subject to the potential due to the nuclei and to the potential due to exchange and correlation effects. This is how the Kohn–Sham Hamiltonian is defined

$$\begin{aligned} \hat{H}_{KS} &= \hat{T}_0 + \hat{V}_H + \hat{V}_{xc} + \hat{V}_{ext} \\ &= -\frac{\hbar^2}{2m_e}\nabla_i^2 + \frac{e^2}{4\pi\varepsilon_0}\int \frac{\rho(\mathbf{r})}{|\mathbf{r}-\mathbf{r}_i|}\,\mathrm{d}\mathbf{r} + \hat{V}_{xc} + \hat{V}_{ext}. \end{aligned} \tag{2.17}$$

The exchange-correlation potential is given by the functional derivative

$$\hat{V}_{xc} = \frac{\delta V_{xc}[\rho]}{\delta \rho}. \tag{2.18}$$

The theorem of Kohn and Sham can now be formulated as follows:

The exact ground-state density $\rho(\mathbf{r})$ of an N-electron system is

$$\rho(\mathbf{r}) = \sum_{i=1}^{N} \phi_i(\mathbf{r})^* \phi_i(\mathbf{r}), \tag{2.19}$$

where the single-particle wave functions $\phi_i(\mathbf{r})$ are the N lowest-energy solutions of the Kohn-Sham equation

$$\hat{H}_{KS}\phi_i = \varepsilon_i \phi_i. \tag{2.20}$$

This theorem enables us to recover the ground-state density by solving N Schrödinger-like equations for non-interacting single particles as opposed to solving conventional Schrödinger equation for system of N interacting electrons, which would require us to resolve a complex system of coupled differential equations. It is important to note, however, that the single-particle wave functions $\phi_i(\mathbf{r})$ are not the wave functions of electrons and they do not have any physical meaning – they are only mathematical artifacts. Only the resultant ground-state density $\rho(\mathbf{r})$ is physically relevant.

Both the Hartree operator V_H and the exchange-correlation operator V_{xc} depend on the density $\rho(\mathbf{r})$, which in turn depends on the ϕ_i which are being searched. However, we can't write down and solve the equations for ϕ_i, until we know the density $\rho(\mathbf{r})$. This chicken and egg scenario is known as the self-consistency problem and requires an iterative procedure with an initial guess of the density function to escape from this paradox. The process is further described in figure 2.1.

2.4 Machine Learning

Until recently, classical molecular dynamics simulations based on empirical force fields and *ab initio* calculations were the only available methods in computational physicists' toolbox. However, the rapid advent of machine learning and artificial intelligence algorithms drastically changed the landscape of physical simulations. When trained on large datasets of atomic configurations and corresponding potential energies and forces, ML models can reproduce the original data accurately, while still retaining the linearly scaling computational cost of classical MD simulations. Although ML methods are often quite data hungry, their accuracy, scalability, and transferability make them a very attractive alternative for various MD applications.

2.5 Deep Potential Molecular Dynamics

The Deep Potential Molecular Dynamics (DeepMD) (1) method uses neural networks for modeling many-body potentials and interatomic forces to drive classical molecular dynamics. The neural network architectures used by the DeepMD method are designed so that they preserve all the natural symmetries in the problem. These models are trained on *ab initio* data and are capable of producing results that are essentially indistinguishable from the original data while scaling linearly with the system size.

One of the most notable challenges in developing an efficient NN schema for molecular dynamics is devising an input format that would preserve the translational, rotational, and permutational symmetry of the system. The raw atomic coordinates from MD simulations cannot

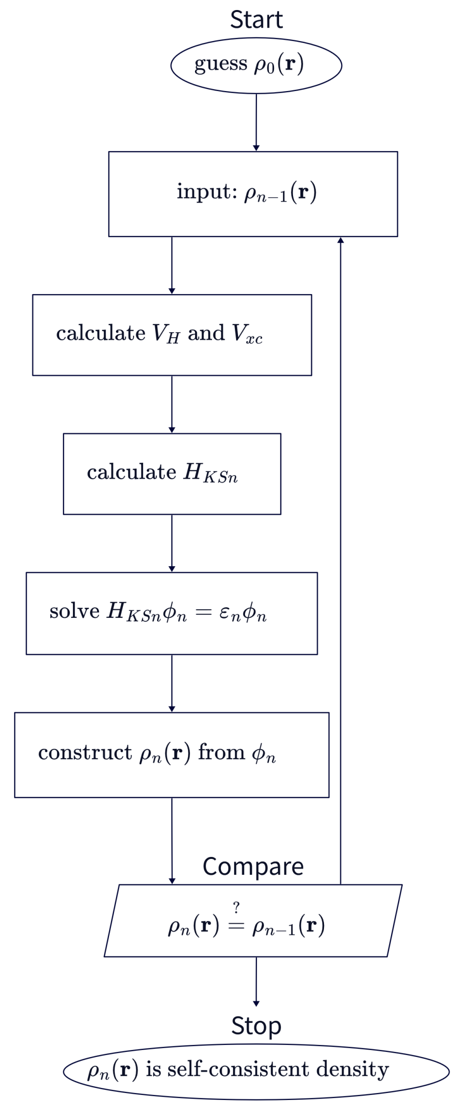
Start
guess $\rho_0(\mathbf{r})$
input: $\rho_{n-1}(\mathbf{r})$
calculate V_H and V_{xc}
calculate H_{KSn}
solve $H_{KSn}\phi_n = \varepsilon_n \phi_n$
construct $\rho_n(\mathbf{r})$ from ϕ_n
Compare
$\rho_n(\mathbf{r}) \overset{?}{=} \rho_{n-1}(\mathbf{r})$
Stop
$\rho_n(\mathbf{r})$ is self-consistent density

Figure 2.1: Flowchart for solving the self-consistent DFT problem.

be used directly, as they do not exhibit these symmetries. Different ML models were proposed to address this problem. For example, the Behler–Parrinello neural network (BPNN) (2) maps the coordinates onto a large set of two- and three-body symmetry functions. Another proposed model, gradient-domain machine learning (GDML) (3), maps the coordinates onto the eigenvalues of the Coulomb matrix. Both of these protocols are successful, but they are also needlessly complicated, with their use-cases being rather limited, as they do not come from a first-principles analysis of the modeling problem, and it is not straightforward to extend them beyond simple systems. The DeepMD methodology attempts to provide a more first principle-based approach to overcome the limitations associated with auxiliary quantities like the symmetry functions or the Coulomb matrix.

Consider a system of N atoms, where the coordinates of these atoms can be represented as a set of position vectors $\{\mathbf{R}_1, \ldots, \mathbf{R}_N\}$, with each $\mathbf{R}_i \in \mathbb{R}^3$. DeepMD decomposes the total system energy E into a sum of energy contributions from individual atoms,

$$E = \sum_i^N E_i, \tag{2.21}$$

where i is an index of an individual atom. Atomic energy E_i is fully determined by the position of the ith atom and by the positions of its near neighbours,

$$E_i = E_{s(i)}(\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}), \tag{2.22}$$

where $N_{R_C}(i)$ denotes the index set of the neighbour atoms of atom i within the cut-off radius R_C. $s(i)$ is the chemical species of atom i. For reasons discussed above, it is less than optimal to use the position vector data $\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}$ when modeling the function $E_{s(i)}$ with DNN. Thus, the DeepMD method introduces a mapping from position vectors to "descriptors" of atomic chemical environment, that better capture the underlying symmetries.

To construct the descriptor for atom i, we first calculate the relative positions of its neighbouring atoms,

$$\mathbf{R}_{ij} = \mathbf{R}_j - \mathbf{R}_i. \tag{2.23}$$

The coordinate of the relative position R_{ij} under the lab reference frame $\{\mathbf{e}^0_x, \mathbf{e}^0_y, \mathbf{e}^0_z\}$ is denoted by $(x^0_{ij}, y^0_{ij}, z^0_{ij})$, such that

$$\mathbf{R}_{ij} = x^0_{ij}\mathbf{e}^0_x + y^0_{ij}\mathbf{e}^0_y + z^0_{ij}\mathbf{e}^0_z. \tag{2.24}$$

Both representations $\mathbf{R}_{ij}$ and $(x^0_{ij}, y^0_{ij}, z^0_{ij})$ preserve the translational symmetry. The rotational symmetry is captured by constructing a local frame of reference and using it to express the coordinates of neighbouring atoms. We first pick atoms with indices $a(i)$ and $b(i)$ from the neighbours $N_{R_C}(i)$ by certain user-specified rules. The local reference frame $\{\mathbf{e}_{i1}, \mathbf{e}_{i2}, \mathbf{e}_{i3}\}$ of atom i is then constructed by

$$\mathbf{e}_{i1} = \mathbf{e}(\mathbf{R}_{ia(i)}), \tag{2.25}$$

$$\mathbf{e}_{i2} = \mathbf{e}\left(\mathbf{R}_{ib(i)} - (\mathbf{R}_{ib(i)} \cdot \mathbf{e}_{i1})\mathbf{e}_{i1}\right), \tag{2.26}$$

$$\mathbf{e}_{i3} = \mathbf{e}_{i1} \times \mathbf{e}_{i2}, \tag{2.27}$$

where $\mathbf{e}(\mathbf{R})$ denotes the normalized vector of $\mathbf{R}$, such that $\mathbf{e}(\mathbf{R}) = \mathbf{R}/|\mathbf{R}|$. The local coordinate (x_{ij}, y_{ij}, z_{ij}) is then calculated from the lab coordinate $(x^0_{ij}, y^0_{ij}, z^0_{ij})$ through the transformation

$$(x_{ij}, y_{ij}, z_{ij}) = (x^0_{ij}, y^0_{ij}, z^0_{ij}) \cdot \mathcal{R}(\mathbf{R}_{ia(i)}, \mathbf{R}_{ib(i)}), \tag{2.28}$$

where

$$\mathcal{R}(\mathbf{R}_{ia(i)}, \mathbf{R}_{ib(i)}) = [\mathbf{e}_{i1}, \mathbf{e}_{i2}, \mathbf{e}_{i3}] \tag{2.29}$$

is the rotation matrix with the columns being the local reference frame vectors. The descriptive information of atom i given by neighboring atom j is then obtained by using either both the radial and angular information or only the radial information

$$\{D_{ij}\} = \begin{cases} \left\{\frac{1}{R_{ij}}, \frac{x_{ij}}{R_{ij}}, \frac{y_{ij}}{R_{ij}}, \frac{z_{ij}}{R_{ij}}\right\}, & \text{full information;} \\ \left\{\frac{1}{R_{ij}}\right\}, & \text{radial-only information.} \end{cases} \tag{2.30}$$

The order of the neighbour indices j in $\{D_{ij}\}$ is fixed by sorting them first by their chemical species and then, within each chemical species, according to their inverse distances to the atom i, i.e., $1/R_{ij}$. The permutational symmetry is naturally preserved in this way. This is the full procedure for constructing the mapping from atomic positions to descriptors, which is denoted by

$$\mathbf{D}_i = \mathbf{D}_i(\mathbf{R}_i, \{\mathbf{R}_j \mid j \in N_{R_C}(i)\}). \tag{2.31}$$

The descriptors $\mathbf{D}_i$ preserve the translational, rotational, and permutational symmetries and are passed to a DNN to evaluate the atomic energies. This process can be mathematically expressed as

$$E_{s(i)} = \mathcal{N}_{s(i)}(\mathbf{D}_i), \tag{2.32}$$

The DNN used by DeepMD method is a feed forward neural network with multiple hidden layers, where each layer transforms the input data $\mathbf{d}_i^{p-1}$ from the previous layer into $\mathbf{d}_i^p$ and passes them as an input to the next layer. The transformation consists of a linear and a non-linear step, i.e.

$$\mathbf{d}_i^p = \varphi\left(\mathbf{W}_{s(i)}^p \mathbf{d}_i^{p-1} + \mathbf{b}_{s(i)}^p\right), \tag{2.33}$$

where φ represents the non-linear function and $\mathbf{W}_{s(i)}^p$ and $\mathbf{b}_{s(i)}^p$ are free parameters of the linear transformation to be optimize by the training process. In order to determine the unknown parameters $\mathbf{W}s^p, \mathbf{b}s^p$ of the linear transformation, a loss function L is minimized during the training process. This loss function is calculated by taking a weighted sum of mean square errors of the predictions made by the DNN. Specifically, the loss function $L(p\epsilon, p_f, p\xi)$ is defined as follows:

$$L(p_\epsilon, p_f, p_\xi) = \frac{p_\epsilon}{N}\Delta E^2 + \frac{p_f}{3N}\sum_i |\Delta \mathbf{F}_i|^2 + \frac{p_\xi}{9N}||\Delta\Xi||^2, \tag{2.34}$$

where ΔE, $\Delta \mathbf{F}_i$ and $\Delta\Xi$ denote root mean square (RMS) error in energy, force, and virial, respectively. The weights p_ϵ, p_f and p_ξ are varying during the learning process. Their dependence on the learning step t is defined as

$$p(t) = p^{\text{limit}}\left[1 - \frac{r_l(t)}{r_l^0}\right] + p^{\text{start}}\left[\frac{r_l(t)}{r_l^0}\right], \tag{2.35}$$

where $r_l(t)$ and r_l^0 are the learning rate at training step t and the learning rate ate the beginning, respectively. The prefactors p^{limit} and

p^{start} are specified by the user configuration. It is easy to observe that as the learning process starts the factor $\frac{r_l(t)}{r_l^0}$ tends to 1 and thus the value of $p(t)$ tends to p^{start}. When the learning process ends, the factor $1 - \frac{r_l(t)}{r_l^0}$ tends to 1 and thus the value of $p(t)$ tends to p^{limit}. By tuning these parameters a user can precisely specify what physical properties of the system should the model be learning to predict at different stages of the learning process. The learning rate $r_l(t)$ function is in our case given by

$$r_l(t) = r_l^0 \cdot d_r^{t/d_s}, \tag{2.36}$$

where d_r and d_s are the decay rate and decay steps, respectively. The decay rate d_r is required to be less than 1.

3 Equations of State

Equations of state (EOS) represent thermodynamic relationships between state variables that characterize the condition of matter under specific physical parameters, including pressure, volume, temperature, and internal energy. State equations usually take the form of

$$f(p, V, T) = 0, \tag{3.1}$$

where p is the pressure, V is the volume, and T the temperature of the system. There exists no single equation of state that would accurately predict the properties of all systems under all possible conditions. As a consequence, a vide variery of state equations have been developed over the years, each modelling a specific class of materials under specific conditions. When modelling a specific system using a particular state equation, it is important to carefully consider the limits of its applicability and its suitability for the system of interest.

3.1 Birch–Murnaghan Equation of State

The Birch-Murnaghan EOS is an empirical isotropic equation that describes the relationship between the volume, pressure, and energy of a solid material under compression. Developed by Francis Birch in 1947, it is based on the Murnaghan equation of state and is widely used in solid-state physics, materials science, and geophysics to examine materials under high pressure conditions.

The derivation of the Birch–Murnaghan EOS is based on the Eulerian finite strain theory. When strains are infinitesimal, the dimensions of a body decrease linearly with pressure. However, this is not the case when strains are finite because matter becomes increasingly incompressible with pressure. The Eulerian finite elastic strain theory treats this phenomenon. When introducing the Eulerian scheme, it is important to note the following points:

- Unlike the Lagrangian scheme, which uses the pre-compression state as a reference, the Eulerian scheme uses the post-compression state.

- Changes are expanded in squared length before and after compression.

We start by considering a cube of side x_0 and volume $V_0 = x_0^3$. We apply a compression u ($u < 0$), so that the length of the side becomes $x = x_0 + u$. The change in squared length is expressed as

$$x^2 - x_0^2 = x^2 - (x-u)^2 = 2ux - u^2. \tag{3.2}$$

The displacement u under uniform compression should be proportional to the length of the cube's side with a proportionality constant c

$$u = cx. \tag{3.3}$$

The constant c is referred to as the strain in the linear elasticity. Combining the above equations we get

$$x^2 - x_0^2 = 2cx^2 - c^2x^2 = 2x^2(c - \frac{1}{2}c^2). \tag{3.4}$$

We obtain the definition for the second-power Eulerian finite strain

$$\varepsilon_{E2} = c - \frac{1}{2}c^2. \tag{3.5}$$

We can use the Eulerian finite strain to express the change in squared length of the side of the cube as

$$x^2 - x_0^2 = 2\varepsilon_{E2}x^2. \tag{3.6}$$

The ration of the pre-compression side length to the post-compression side length can be expressed using the Eulerian finite strain as

$$\frac{x_0}{x} = \sqrt{1 - 2\varepsilon_{E2}}. \tag{3.7}$$

We can use this equation to also express the ratio of the pre-compression volume and the post-compression volume as

$$\frac{V_0}{V} = \left(\frac{x_0}{x}\right)^3 = (1 - 2\varepsilon_{E2})^{\frac{3}{2}}. \tag{3.8}$$

Conversely, we can express the finite strain using the volume ratio

$$\varepsilon_{E2} = \frac{1}{2}\left[1 - \left(\frac{V_0}{V}\right)^{\frac{3}{2}}\right]. \tag{3.9}$$

The ε_{E2} when the body is compressed. To make the strain positive, we define f_{E2} as

$$f_{E2} = -\varepsilon_{E2} = \frac{1}{2}\left[\left(\frac{V_0}{V}\right)^{\frac{3}{2}} - 1\right]. \tag{3.10}$$

The first law of thermodynamics in terms of Helmholtz free energy is given by

$$\mathrm{d}F = -S\mathrm{d}T - p\mathrm{d}V. \tag{3.11}$$

We can simply derive an expression for pressure of isothermal system as

$$p = -\left(\frac{\partial F}{\partial V}\right)_T. \tag{3.12}$$

The Helmholtz free energy of matter may be expressed by a series of the Eulerian finite strain

$$F = a_0 + a_1 f_{E2} + a_2 f_{E2}^2 + a_3 f_{E2}^3 + \cdots . \tag{3.13}$$

The absolute value of F is abitrary, so we can set $a_0 = 0$. Pressure should be zero in an uncompressed state, as should f_{E2}, so from the equation (3.12) we get

$$p_{f_{E2}=0} = -\left(\frac{\partial F}{\partial f_{E2}}\right)_{T,f_{E2}=0}\left(\frac{\partial f_{E2}}{\partial V}\right)_{T,f_{E2}=0} = -a_1\left(\frac{\partial f_{E2}}{\partial V}\right)_{T,f_{E2}=0} = 0. \tag{3.14}$$

Therefore coefficient a_1 in the expansion of F is $a_1 = 0$. We continue by truncating the expansion to

$$F \doteq a_2 f_{E2}^2 + a_3 f_{E2}^3. \tag{3.15}$$

Plugging this into the equation (3.12), we get

$$\begin{aligned} p &= -\left[\frac{\partial}{\partial V}(a_2 f_{E2}^2 + a_3 f_{E2}^3)\right]_T \\ &= -(2a_2 f_{E2} + 3a_3 f_{E2}^2)\left(\frac{\partial f_{E2}}{\partial V}\right)_T \\ &= -2a_2(1+\xi_1 f_{E2}) f_{E2}\left(\frac{\partial f_{E2}}{\partial V}\right)_T, \end{aligned} \tag{3.16}$$

where $\xi_1 = 3a_3/2a_2$. As derived in [4], the parameter ξ_1 is given by

$$\xi_1 = \frac{3}{2}\left(B'_{T_0} - 4\right), \tag{3.17}$$

where B'_{T_0} is the pressure derivative of the isothermal bulk modulus at standard temperature. Refferring to the same paper, the coefficient a_2 is

$$a_2 = \frac{9}{2} B_{T_0} V_0, \tag{3.18}$$

where B_{T_0} is the isothermal bulk modulus at standard temperature. From definition of finite strain we get

$$\left(\frac{\partial f_{E2}}{\partial V}\right)_T = -\frac{1}{3V_0}\left(\frac{V_0}{V}\right)^{\frac{5}{3}}. \tag{3.19}$$

By plugging equations (3.10), (3.19), (3.18), (3.17) into (3.16), we recover the the third-order Birch–Murnaghan EOS of the form

$$p = \frac{3}{2}K_{T_0}\left[\left(\frac{V_0}{V}\right)^{\frac{7}{3}} - \left(\frac{V_0}{V}\right)^{\frac{5}{3}}\right]\left[1 + \frac{3}{4}(B'_{T_0} - 4)\left\{\left(\frac{V_0}{V}\right)^{\frac{2}{3}} - 1\right\}\right]. \tag{3.20}$$

4 Implementation and Methodology

The implementation of the DeepMD protocol employed in this thesis is based on the DeepMD-kit software suite [5]. DeepMD-kit uses TensorFlow [6] to provide an efficient and flexible toolkit for utilizing the DeepMD scheme. DeepMD-kit comes with an interface to the LAMMPS classical molecular dynamics code [7], which was used in this work for computing all the material properties with the trained machine learning models. The models were trained on MD data generated with OpenMX, a DFT-based nanoscale material simulations software [1]. The Python programming language [2] alongside with its multitude of scientific computation packages was used for building the training and evaluation infrastructure. All of the code needed to reproduce the results presented in this thesis is available online [3].

4.1 Preparing the DFT Datasets

As already stated, all DFT trajectories were calculated using OpenMX. Since the computation resources available for this work were quite limited, we choose to only work with simple systems of silicon, as they do not exhibit any complicated quantum mechanical effects and thus should be easier to model and train on. To see how well the models perform on wide variety of learning data, three distinct datasets were prepared – an amorphous silicon dataset, a crystalline silicon dataset, and a mixed dataset containing both the amorphous and the crystalline frames. Each dataset contains a selection of frames taken from OpenMX simulations of silicon systems with distinct volumes. After the simulation frames were generated, they were further distilled down as to not include frames that were too similar to each other. This was done simply by removing frames that were too close to each other in terms of their time step. The detailed parameters for each used dataset are listed in table 4.1.

1. See `https://www.openmx-square.org/`
2. See `https://www.python.org/`
3. See `https://github.com/hacksparr0w/bachelor-thesis`

Table 4.1: The parameters used for the DFT datasets.

	Training		Validation	
Dataset	Samples	Frames	Samples	Frames
Crystalline	c-0.99	200	c-1.01	200
	c-1.03	200		
Amorphous	a-2.15	150	a-2.25	150
	a-2.20	150	a-2.31	150
	a-2.29	150		
	a-2.30	150		
	a-2.35	150		
	a-2.40	150		
Combined	c-0.99	200	c-1.01	200
	c-1.03	200		
	a-2.15	150	a-2.25	150
	a-2.20	150	a-2.31	150
	a-2.29	150		
	a-2.30	150		
	a-2.35	150		
	a-2.40	150		

4.2 Choosing the DNN Hyperparameters

DeepMD provides a bright pallete of hyperparameters that can be used to fine-tune the behavior of a DNN. The following is an exceprt from a sample DNN training input:

```
{
  "model": {
    "descriptor": {
      "type": "se_e2_a",
      "sel": [70],
      "rcut_smth": 1.8,
      "rcut": 6.0,
      "neuron": [5, 10, 20],
      "resnet_dt": false,
      "seed": 260222622
    },
    "fitting_net": {
      "neuron": [20, 20, 20],
      "resnet_dt": true,
      "seed": 260222622
    }
  },
  "learning_rate": {
    "type":"exp",
    "decay_steps": 20000,
    "start_lr": 0.001,
    "stop_lr": 1e-08
  },
  "loss": {
    "type": "ener",
    "start_pref_e": 1,
    "limit_pref_e": 500,
    "start_pref_f": 1000,
    "limit_pref_f": 10,
    "start_pref_v": 0,
    "limit_pref_v": 0
  },
  "training": {
    "numb_steps": 2000000,
    "seed": 10
  }
}
```

Let's briefly discuss the most import configuration parameters and their effect on the training and inference processes.

The `model` configuration section defines the architecture details of a DNN. As per the definition in (2.32), the neural network is simply a function of descriptors that predicts an energy value. The DeepMD protocol implements this function as two separate neural networks, where one is called the descriptor network and is defined by the `model.descriptor` configuration section and the other is called the fitting network and is defined by the `model.fitting_net` configuration section. The descriptor network is fed with the descriptor data as specified by the `model.descriptor.type` option, which makes it possible to control whether the network uses the full radial and angular information as per (2.30). All of the models trained in this work use the full radial and angular information, so the `model.descriptor.type` option is always set to `se_e2_a`. The `model.descriptor.sel` option determines how many neighboring atoms for a given atom type (indices correspond to the indices of the `model.descriptor.type` option) are used for inference. The ideal value for this option can be determined empirically from the training data using the `dp neighbor-stat` utility that comes with the DeepMD-kit software. This value was kept constant for all models in this work. The cutoff radius specified by the `model.descriptor.rcut` and `model.descriptor.rcut_smth` options limits the radius of the local environment constructed for each atom. The neighbours filtered with the `model.descriptor.sel` option must be within the cutoff radius, which is measured in Å. One of the most important hyperparameters that have perhaps the most significant effect on the quality of the model inference are the `model.descriptor.neuron` and `model.fitting_net.neuron` options. These options specify the number of layers and number of neurons in each layer of the neural network. One of the difficulties of using deep neural networks is coming up with a good heuristics for driving the decisions on the apropriate network atchitectures. A great portion of chapter 5 focuses on choosing and evaluating various network architectures. When training the models, it is important to choose a random seed that is used by the internal pseudo-random number generator to initialize the weights and biases of the neural network. This is very important for two reasons. Firstly, we can ensure reproducibility of training results by using the same seeds. Secondly, we can use multiple models with different seeds and calculate inference uncertainties based on their inference results, which will differ the more the uncertain the models are. Comparison of models with different seeds is also discussed in later chapters.

Another important configuration section is the `learning_rate` section. It implements the learning rate function as per (2.36). The values of this configuration were optimized to go along with the empirically chosen number of training steps (the `training.numb_steps` option) and do not differ on a per-model basis.

Last but not least, the `loss` configuration section defines the behavior of the loss function by varying the loss prefactors defined in (2.35). In our case, the loss function is set so that it aggressively optimizes for the force predictions at the beginning of the training process and then gradually shifts the focus to the energy predictions.

The whole training matrix of models used in this work is available in table **??**.

5 Results and Evaluation

6 Conclusion

Bibliography

1. ZHANG, Linfeng; HAN, Jiequn; WANG, Han; CAR, Roberto; E, Weinan. Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics. *Physical Review Letters*. 2018, vol. 120, no. 14. Available from DOI: `10.1103/physrevlett.120.143001`.
2. BEHLER, Jörg; PARRINELLO, Michele. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Phys. Rev. Lett.* 2007, vol. 98, p. 146401. Available from DOI: `10.1103/PhysRevLett.98.146401`.
3. CHMIELA, Stefan; TKATCHENKO, Alexandre; SAUCEDA, Huziel E.; POLTAVSKY, Igor; SCHÜTT, Kristof T.; MÜLLER, Klaus-Robert. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*. 2017, vol. 3, no. 5, e1603015. Available from DOI: `10.1126/sciadv.1603015`.
4. KATSURA, Tomoo; TANGE, Yoshinori. A Simple Derivation of the Birch–Murnaghan Equations of State (EOSs) and Comparison with EOSs Derived from Other Definitions of Finite Strain. *Minerals*. 2019, vol. 9, no. 12. ISSN 2075-163X. Available from DOI: `10.3390/min9120745`.
5. WANG, Han; ZHANG, Linfeng; HAN, Jiequn; E, Weinan. DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics. *Computer Physics Communications*. 2018, vol. 228, pp. 178–184. Available from DOI: `10.1016/j.cpc.2018.03.016`.
6. MARTÍN ABADI; ASHISH AGARWAL; PAUL BARHAM; EUGENE BREVDO; ZHIFENG CHEN; CRAIG CITRO; GREG S. CORRADO; ANDY DAVIS; JEFFREY DEAN; MATTHIEU DEVIN; SANJAY GHEMAWAT; IAN GOODFELLOW; ANDREW HARP; GEOFFREY IRVING; MICHAEL ISARD; JIA, Yangqing; RAFAL JOZEFOWICZ; LUKASZ KAISER; MANJUNATH KUDLUR; JOSH LEVENBERG; DANDELION MANÉ; RAJAT MONGA; SHERRY MOORE; DEREK MURRAY; CHRIS OLAH; MIKE SCHUSTER; JONATHON SHLENS; BENOIT STEINER; ILYA SUTSKEVER; KUNAL TALWAR; PAUL TUCKER; VINCENT VANHOUCKE;

VIJAY VASUDEVAN; FERNANDA VIÉGAS; ORIOL VINYALS; PETE WARDEN; MARTIN WATTENBERG; MARTIN WICKE; YUAN YU; XIAOQIANG ZHENG. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Available also from: `https://www.tensorflow.org/`. Software available from tensorflow.org.

7. THOMPSON, A. P.; AKTULGA, H. M.; BERGER, R.; BOLINTINEANU, D. S.; BROWN, W. M.; CROZIER, P. S.; 'T VELD, P. J. in; KOHLMEYER, A.; MOORE, S. G.; NGUYEN, T. D.; SHAN, R.; STEVENS, M. J.; TRANCHIDA, J.; TROTT, C.; PLIMPTON, S. J. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *Comp. Phys. Comm.* 2022, vol. 271, p. 108171. Available from DOI: `10.1016/j.cpc.2021.108171`.

A An appendix

Here you can insert the appendices of your thesis.