Learning Symmetry-Preserving Interatomic Force Fields for Atomistic Simulations

by

Simon Lutz Batzner

B.Sc., University of Stuttgart (2017)

Submitted to the Center for Computational Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computation for Design and Optimization

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Signature redacted

Author .....
Center for Computational Engineering
22 May 2019

Certified by ..... Signature redacted .....
Jeffrey C. Grossman
Professor of Materials Science and Engineering
Thesis Supervisor

Certified by ..... Signature redacted .....
Boris Kozinsky
Associate Professor, Harvard University
Thesis Supervisor

Accepted by ..... Signature redacted .....
Youssef M. Marzouk
Professor, Department of Aeronautics and Astronautics
Co-Director, Center for Computational Engineering

Learning Symmetry-Preserving Interatomic Force Fields for Atomistic Simulations

by

Simon Lutz Batzner

Submitted to the Center for Computational Engineering
on 22 May 2019, in partial fulfillment of the
requirements for the degree of
Master of Science in Computation for Design and Optimization

Abstract

Machine-Learning Interatomic Force-Fields have shown great promise in increasing time- and length-scales in atomistic simulations while retaining the high accuracy of the reference calculations that they are trained on. Most proposed models aim to learn the potential energy surface of a system of atoms as a function of atomic coordinates and species and obtain the forces acting on the atoms as the negative of the gradient of the global energy with respect to the atomic positions. For the time evolution of an atomistic system in molecular dynamics, however, only atomic forces are required. This thesis examines the construction of a direct approach for learning atomic forces, thereby bypassing the need for learning an energy-based model. Predicting atomic forces directly requires the careful consideration of incorporating the symmetries of 3D space into the model. The construction of an efficient, direct, and symmetry-preserving deep learning model that can predict atomic forces in a fully end-to-end fashion is shown. The model's accuracy, its computational efficiency for training as well as its computational efficiency at time of prediction are evaluated. Finally, the approach is used in the simulation of different small organic molecules and the resulting Molecular Dynamics simulations are analyzed.

Thesis Supervisor: Jeffrey C. Grossman
Title: Professor of Materials Science and Engineering

Thesis Supervisor: Boris Kozinsky
Title: Associate Professor, Harvard University

Acknowledgments

I would like to express my deepest gratitude to Prof. Boris Kozinsky, for his guidance and mentorship, for his unwavering support, and most importantly, for giving me the opportunity to learn from him. I look forward to continuing working in the Materials Intelligence Research group.

I would also like to thank Prof. Alexie Kolpak for introducing me to the wonderful world of atomistic simulation and Prof. Jeffrey Grossman for his guidance as well as for his support during uncertain times.

I have greatly enjoyed working both in the Kozinsky and the Kolpak groups and would like to thank my fellow lab members: Spencer and Levi for introducing me to machine-learning interatomic potentials and for patentiently listening to even the wildest ideas that a wide-eyed first year can come up with, Jon and Steven for the many lively discussions about the relative merits of our favorite regression models, and Lixin and Nicola for sharing their postdoctoral wisdom. I'd also very much like to thank Tess Smidt for the many helpful inputs and discussions about Tensor Field Neural Networks.

Finally, I would like to thank my friends and family: my girlfriend Jenica, for her encouragement, her support, and for making my time in Cambridge such a wonderful experience. My two younger brothers, Matthias and Kilian, thank you for being my role models for all these years. And finally, my parents: I cannot put into words how grateful I am for your unconditional support in everything I do. Thank you for allowing me to stay close to home while far away.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Molecular Dynamics

Over the past decades, the atomistic simulation of molecular and condensed matter has become an important third pillar in the chemical and physical sciences, supplementing experiments and theory. The computational study of molecules and materials has enabled scientific breakthroughs in a wide variety of disciplines, including

- the computational design of novel battery materials and catalysts [1, 2]
- drug discovery and the advanced understanding of important biological and chemical processes [3]
- the design of novel energy materials such as thermoelectrics. [4]

Across all these disciplines, there is great need for increasing the accessible time- and length-scales of simulations while retaining high accuracy. The computational methods deployed range from very high accuracy quantum-mechanical and quantum-chemical methods, such as coupled cluster methods [5], to more approximate but also more efficient methods like density functional theory [6], all the way to continuum methods that allow for the simulation of macroscopic structures, like cars and airplanes. For the study of molecules and materials at the atomistic level in particular, the method of molecular dynamics has been very successful [7, 8], taking its place

between density functional theory and coarse-graining or even continuum methods on the axes of both accuracy and efficiency. The core idea behind molecular dynamics simulation is to study the time evolution of a system of atoms under Newtonian dynamics:

$$m_i \frac{d^2\mathbf{r_i}}{dt^2} = \mathbf{F_i}(\mathbf{r_1}, \mathbf{r_2}, ..., \mathbf{r_N}) \tag{1.1}$$

In order to propagate such a system in time, at every step, the forces acting on the atoms are required. Equivalently, since the forces are the negative gradient of the potential energy surface, access to the potential energy of the system also suffices:

$$\mathbf{F_i}(\mathbf{r_1}, \mathbf{r_2}, ..., \mathbf{r_N}) = -\frac{\partial \mathbf{V}(\mathbf{r_1}, \mathbf{r_2}, ..., \mathbf{r_N})}{\partial \mathbf{r_i}} \tag{1.2}$$

These forces can either be obtained from empirical models (often also called *classical* force fields) or from very accurate, but often expensive quantum-mechanical calculations, such as density functional theory, in which the forces are obtained via the Hellman-Feynman theorem.

A typical workflow in a molecular dynamics simulations reads as follows:

- initialize positions and velocities of a set of atoms
- compute instantaneous forces given an atomic configuration
- update the atomic positions based on an integration scheme of the Newton's equations, a second-order differential equation
- equilibrate the system: this typically involves simulating the system under study for a certain time
- record the statistics of interest from the simulation

The most critical open challenge in molecular dynamics simulation is obtaining accurate, yet efficient estimates of the atomic forces. Since the forces determine the

dynamics of the system, getting high-accuracy and faithful forces has been an area of intensive research [9, 10, 11]. As force estimates have to be obtained at every time step, this operation usually constitutes the computationally most expensive component in molecular dynamics simulations. The fundamental trade-off is one between efficiency and accuracy: highly accurate, quantum-mechanical methods tend to come at a very high computational cost and bad scaling, thereby limiting the possible size of the system to be studied and the length of the simulation. More efficient classical force-fields based on simple, parameterized forms are often not flexible enough to capture complicated many-body physics or events such as bond-breaking, but are highly efficient due to their very simple functional form.

Classical potentials are usually designed using a simple functional form that aims to capture the relevant physics of the system. For example, one of the simplest, but fastest empirical potentials that is widely used across disciplines is the Lennard-Jones potential, a pair potential that only requires two parameters to be fitted. In a pair potential, the total energy of the system is expressed as a sum of two-body interactions based on their interatomic distance:

$$E = E_0 + \frac{1}{2}\sum_{i \neq j} V(\mathbf{r_i} - \mathbf{r_j}) \tag{1.3}$$

In the Lennard-Jones potential specifically, the interaction is computed as:

$$V(r) = \epsilon[(\frac{\sigma}{r})^{12} - 2(\frac{\sigma}{r})^{6}] \tag{1.4}$$

where ϵ and σ are parameters to be fitted. Parameters in classical potentials are often adjusted in a way such that they match measured experimental data or data obtained from high-accuracy reference calculations. Figure 1-1 shows the typical form of a Lennard-Jones interaction potential. Other popular classical force fields include:

- Stillinger-Webber potential [12], which includes three-body terms and is therefore able to describe e.g. Silicon with good accuracy

- Embedded Atom Method (EAM) [13], a potential based on the electron density that is particularly well suited for the simulation of metals
- Reactive Force Field (ReaxFF), specifically designed to model the breaking of bonds and therefore the simulation of chemical reactions [9]

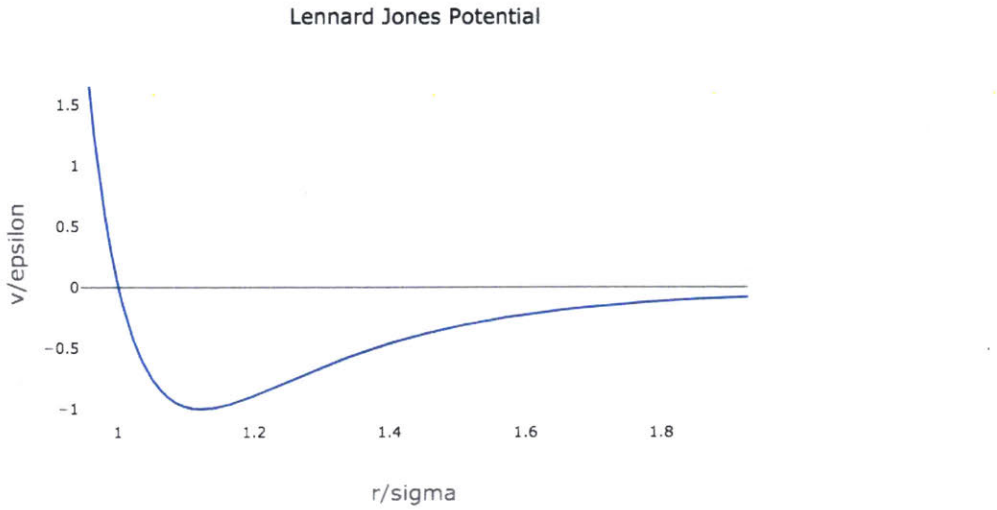

Figure 1-1: The typical form of a Lennard-Jones potential: a short-range repulsive term and a long-range attractive term counteract each other.

At the other end of the trade-off between accuracy and efficiency lies the method of ab-initio molecular dynamics (AIMD) [14], which allows for the study of systems at finite temperature by computing highly accurate forces *on-the-fly* from electronic structure methods. However, since at every time step, a reference DFT calculation has to be computed, the systems that can currently be studied using this method are limited to small numbers of atoms as well as limited time-scales.

1.2 Machine Learning Force Fields

Moving toward the goal of obtaining atomic forces at quantum-mechanical accuracy at a reasonable cost, machine learning has been proposed as a potential path forward

as it provides a computationally efficient way of fitting highly complex and unbiased potential energy surfaces [15, 16]. The workflow across different methods is the following:

1. Obtain a set of high-accuracy reference calculations, typically from density functional theory calculations. It should be noted that, depending on the complexity of the problem, the regression model and the algorithm of how training data are chosen, the number of required training structures to obtain a faithful predictive model can vary greatly [17].

2. Potentially specify a descriptor or *fingerprint*, i.e. a way of transforming the raw atomic coordinates from $\mathbb{R}^3$ to a feature vector. Often, only a finite atomic neighborhood is included in the prediction of atomic quantities (such as atomic energies or atomic forces): i.e. only atoms within a certain cutoff sphere around the current atom of interest contribute to the prediction of the atomic target quantity of interest. This is used to study large systems consisting of many atoms. In such situations, due to the size extensivity introduced by using a local decomposition, one can train on reference structure of sizes accessible by DFT, but predict on much larger structures by iteratively evaluating the trained model.

3. Specify a regression model, that means a learning algorithm that can predict atomic forces or total energies as a function of atomic positions and chemical species, potentially expressed in some feature space

4. Optimize the regression model to fit the set of training structures according to a chosen metric until it can accurately reproduce the energies and/or forces in the specified reference structures

5. Use the trained model in a molecular dynamics simulation and iteratively predict forces as the atomic positions are updated over the course of the simulation.

A variety of different approaches to modeling potential energy surfaces have been

proposed, usually consisting of a different combination of atomistic descriptor and regression model. Three different classes of models have been particularly successful:

- Gaussian descriptors (also called *symmetry functions*) in combination with shallow neural networks, e.g. Behler-Parrinello Neural Networks [18]
- Kernel-based approaches, e.g. the Gaussian Approximation Potentials [16]
- Deep-Learning based approaches, e.g. the SchNet approach [19] or DeepMD [20]

To provide an overview over relevant previous work, this section will outline different approaches taken and discuss their relative merits. An ideal potential / force-field should satisfy a variety of different criteria:

- Accuracy: the potential / force-field should be able to describe a wide variety of atomistic configurations as faithfully as possible.
- Efficiency: the potential should be computationally cheap to evaluate in order to be able to study large length- and time-scales.
- Operate on point clouds: the potential should not be restricted to a grid, such that arbitrary structures can be studied.
- Translational and rotational invariance of total energies with respect to these symmetry operations acting on a set of atoms, invariance with respect to the permutation of two atoms of the same species.
- Translational and rotational equivariance of the atomic forces, equivariance with respect to the permutation of two atoms of the same species.
- Reactivity: the potential should be able to describe the forming and breaking of chemical bonds.
- Transferability: the potential should be able to capture atomistic configurations dissimilar to what it has been trained on.

- Conservation of energy: this is automatically satisfied for any energy-based model that is not a function of time. Note that this would e.g. be violated in an active learning scheme where a set of reference structures is extended over time (e.g. as novel atomic configurations become available during a simulation). However, in such a scheme the potential could be set fixed at some point, meaning no more reference structures are added to the training set, therefore making the potential energy-conserving.

- Meaningful and interpretable descriptors.

- Size extensivity: this is especially important for the simulation of extended systems. In order to increase the accessible length scale of materials simulations using machine learning force fields, one should be able to train on reference structures consisting on n atoms and be able to simulate systems with N » n atoms. This is usually enabled by deploying an atomic decomposition scheme. Often, the global energy of the system is expressed as a sum of atomic energies [18]. It should be noted that these energies are fictitious and do not correspond to observables.

- Uncertainty estimates accompanying the model's predictions: trustable estimates of a model's uncertainty in its prediction are beneficial in a number of ways: firstly, they greatly increase the trustworthiness of a machine learning model's prediction, specifically as the model is queried to predict on data far away from the training set or on physical phenomena that weren't encountered during the construction of the training set (such as e.g. a diffusion event, bond breaking, etc.). Predictions from the Machine Learning model with high uncertainty can then be rejected and a more accurate method such as e.g. a full ab-initio calculation can be queried instead. Furthermore, uncertainty metrics prove useful for building active learning strategies [17]. In such scenarios, there usually exists some low-fidelity model (in the present case e.g. a machine-learning force field) and an "oracle", i.e. a high-accuracy method which can be queried to label arbitrary data. This can then be exploited to greatly reduce

the number of required reference training structures by only adding structures to the data set for which model has given high uncertainty, as shown in [17].

- Linear scaling with the number of atoms.
- Systematic improvement: as more data become available, it should be easy to leverage these to make the force field more accurate.
- Effort of construction: an ideal force field requires little human construction and can be built in an easy and automated fashion.

1.2.1 Behler-Parrinello Networks

Behler-Parinnello networks [15] have been used to model a wide variety of materials systems [21, 22, 23, 24]. They consist of a set of symmetry functions (a Gaussian basis set expansion with a smooth cutoff) and a feed-forward neural network, acting on the atomic positions expressed in the basis of symmetry functions. The symmetry functions consist of radial and angular functions and are used to describe atomic environments in a rotationally invariant fashion. Importantly, the number of symmetry functions is independent of the number of atoms in the current atomic environment [18], a necessary requirement since fully-connected feed-forward neural networks require an input vector of fixed length, whereas the number of atoms in an atomic environment can vary during a simulation. An example of the functional form of radial type of symmetry function is given in [18]:

$$G_i^2 = \sum_{j=1}^{N} e^{-\eta(R_{ij}-R_s)^2} f_c(R_{ij}) \tag{1.5}$$

where R_s and η are hyperparameters that control the position and width of the Gaussians, respectively, and f_c is a cutoff function (often based on a cosine) that aims to capture the physical effect that atoms further from the central atom contribute less to the interaction than those closer to the central atom. In addition, cutoff functions often have a value and slope of 0 directly at the cutoff distance, a requirement for obtaining forces, since these are based on derivatives of the total energy [18].

Size extensivity is ensured by predicting the global energy as a sum of local, atomic energies which are determined by their local atomic environments, typically consisting of atoms inside a cutoff sphere with a radius between 6.5 and 10 Angstroms [18]. Optionally, one can also separately account for long-range, electrostatic terms, which gives a resulting decomposition of the total energy as the sum of a short-range, local term and a long-range term aiming to capture electrostatic interactions [18]:

$$E_{tot} = \sum_{i=1}^{N} E_i + E_{elec}$$

where E_i is the local energy of the i-th atom.

In order to maintain a computationally tractable model, weights are shared across chemical species. i.e. the number of networks trained is equal to the number of distinct chemical species in the system. Finally, atomic forces are necessary to evolve the system of atoms in time. These can be obtained by differentiation of the network output, i.e. the global energy w.r.t the atomic positions. For example, the force with respect to atomic coordinate r_i resulting from the short-range energy can be written as [18]:

$$F_{r_i} = -\frac{\partial E}{\partial r_i} = -\sum_{j=1}^{N_{atoms}} \frac{\partial E_j}{\partial r_i} = -\sum_{j=1}^{N_{atoms}} \sum_{k=1}^{N_{symmetry}} \frac{\partial E_j}{\partial G_{j,k}} \frac{\partial G_{j,k}}{\partial r_i} \tag{1.6}$$

While these networks are shallow and analytical derivatives are available, the computation of derivatives is an additional computational overhead, especially as multiple derivatives of the symmetry functions with respect to the atomic positions have to be computed. Finally, due to the use of a neural network as the regression model, Behler-Parinello neural networks currently do not come with reliable uncertainty estimates. Instead, the use of ensembles has been proposed as a method for approximating the uncertainty in the networks prediction and for obtaining relevant new data to be added to the training set [18].

1.2.2 Kernel-based Methods

Another type of machine-learning interatomic potential that has been very successful at describing PES [16] is based on kernels, i.e. functions that compute a distance or similarity metric between two atomic environments. Based on either Gaussian Process Regression or Kernel Ridge Regression, energies can then be predicted by measuring the similarity of an unseen data point to existing ones. In practice, kernel methods are often preferred over neural networks in the absence of large data sets, as they can give good predictive models even with a limited amount of training data [17]. In addition, in Bayesian methods, such as e.g. Gaussian Processes, a distribution over functions is learned, which gives immediate access to measures of uncertainty, a property often sought after in the development of force-fields. The resulting uncertainty metrics can be exploited to build accurate models from comparably small data sets of expensive reference calculations [17].

1.2.3 Deep-Learning-based Approaches and the SchNet model

The motivation for deep-learning-based approaches for the construction of interatomic force fields lies in their great success in other domains, particularly in the fields of vision and language [25, 26]. The fundamental principle is to learn a model in a fully end-to-end fashion, i.e. not to specify hand-crafted, engineered descriptors, but to instead let the model identify relevant patterns in the data. These models are known to be highly data-intensive, meaning that very often, successful results are only achievable when large training set sizes are available. The computational model underlying deep learning is again a neural network, i.e. multiple layers of weight-matrix multiplications, followed by nonlinear transforomations. Typically, in deep-learning approaches to the construction of force-fields, as opposed to the shallow neural networks used in Behler-Parrinello networks, the layers are not only fully connected, but certain layers exhibit special structure [19].

As they are also based on neural networks, similar to the Behler approach (al-

though usually deeper and more complex [19]), they share some of their properties with Behler-Parrinello neural networks, i.e. they tend to be computationally efficient, often lack interpretability, are very flexible, but do not come with principled uncertainty estimates (however, progress has been made towards this goal, e.g. through the use of Gaussian approximations [27]). Since the method proposed here will be compared to the SchNet approach [19] in terms of its accuracy and computational efficiency, the SchNet model will be explained in further detail in the following section.

SchNet leverages continuous filter convolutional layers, i.e. convolutional layers that can operate on arbitrary points in $\mathbb{R}^3$ to model atomic interactions, instead of grid-bound data (such as e.g. images). By predicting total energies invariant to rotation, translation, and permutation and obtaining forces via the computation of derivatives, this defines an energy-conserving, rotationally equivariant force-field [19]. In the SchNet model, the system is represented via different atomic embeddings, that are repeatedly updated based on pair-wise interaction with other atoms. At each layer l, every atom i is described by its current *state* or *feature* vector, giving the following tuple to describe a set of n atoms:

$$\mathbf{X^l} = (\mathbf{x_1^l}, \mathbf{x_2^l}, ..., \mathbf{x_n^l})$$

where every atom is represented by a vector $\mathbf{x_i^l} \in \mathbb{R}^F$, F being the number of feature maps [19]. Starting from a randomly initialized embedding based solely on the atomic number Z_i (the embedding is however optimized during training), this embedding is refined through continuous-filter interactions with other atoms in the atomic environment, through *atom-wise* layers, and through non-linearities. These different computational units will briefly be described in the following as the neural network used in this thesis shares a number of characteristics with this approach.

Atom-wise layers are based on a simple affine transformation acting on the feature vector of individual $\mathbf{x_i^l}$, not accounting for atomic interactions:

$$\mathbf{x_i^{l+1}} = \mathbf{W^l x_i^l} + \mathbf{b^l}$$

Continuous-filter convolutions enable updating the feature vector based on pairwise interactions with the atoms in its local environment. The filter values are obtained via filter-generating networks: these first express a set of relative distances to the central atom $||\mathbf{r_j} - \mathbf{r_i}||$ (to satisfy rotational invariance) in a basis of shifted Gaussians, and then use a two-layer dense network with a shifted softplus nonlinearity to map these to the corresponding filter values. This gives the following definition of a continuous-filter convolution in which the state vector is updated via element-wise multiplication "$\circ$" based on the filter values [19]:

$$\mathbf{x_i^{l+1}} = (\mathbf{X^l} * \mathbf{W^l})_i = \sum_{j=0}^{n_{atoms}} \mathbf{x_j^l} \circ \mathbf{W^l}(\mathbf{r_j} - \mathbf{r_i}) \tag{1.7}$$

Finally, an atom-wise layer, a convolutional layer, an atom-wise layer, a nonlinearity and a final atom-wise layer are connected in series to define an *interaction block*, which gives the update of the state vector from the l-th to the l+1-th layer. It should also be noted that this happens through a residual learning update, meaning the interaction block models the difference between $\mathbf{x_l}$ and $\mathbf{x_{l+1}}$. Finally, the atomic contributions are summed to obtain the total potential energy of the system. Finally, SchNet is trained to minimize the joint loss function of energy and forces RMSEs:

$$L = \rho||E - \hat{E}||^2 + \frac{1}{n_{atoms}} \sum_{i=1}^{n_{atoms}} ||\mathbf{F_i} - (-\frac{\partial \hat{E}}{\partial \mathbf{R}})|| \tag{1.8}$$

Bibliography

[1] G Ceder, Y-M Chiang, DR Sadoway, MK Aydinol, Y-I Jang, and B Huang. Identification of cathode materials for lithium batteries guided by first-principles calculations. *Nature*, 392(6677):694, 1998.

[2] Jens Kehlet Nørskov, Thomas Bligaard, Jan Rossmeisl, and Claus H Christensen. Towards the computational design of solid catalysts. *Nature chemistry*, 1(1):37, 2009.

[3] Noriaki Okimoto, Noriyuki Futatsugi, Hideyoshi Fuji, Atsushi Suenaga, Gentaro Morimoto, Ryoko Yanai, Yousuke Ohno, Tetsu Narumi, and Makoto Taiji. High-performance drug discovery: computational screening by combining docking and molecular dynamics simulations. *PLoS computational biology*, 5(10):e1000528, 2009.

[4] Giri Joshi, Ran He, Michael Engber, Georgy Samsonidze, Tej Pantha, Ekraj Dahal, Keshab Dahal, Jian Yang, Yucheng Lan, Boris Kozinsky, et al. Nbfesb-based p-type half-heuslers for power generation applications. *Energy & Environmental Science*, 7(12):4070–4076, 2014.

[5] Hendrik J Monkhorst. Calculation of properties with the coupled-cluster method. *International Journal of Quantum Chemistry*, 12(S11):421–432, 1977.

[6] Pierre Hohenberg and Walter Kohn. Inhomogeneous electron gas. *Physical review*, 136(3B):B864, 1964.

[7] Martin Karplus and J Andrew McCammon. Molecular dynamics simulations of biomolecules. *Nature Structural & Molecular Biology*, 9(9):646, 2002.

[8] Kevin Leung and Joanne L Budzien. Ab initio molecular dynamics simulations of the initial stages of solid–electrolyte interphase formation on lithium ion battery graphitic anodes. *Physical Chemistry Chemical Physics*, 12(25):6583–6586, 2010.

[9] Adri CT Van Duin, Siddharth Dasgupta, Francois Lorant, and William A Goddard. Reaxff: a reactive force field for hydrocarbons. *The Journal of Physical Chemistry A*, 105(41):9396–9409, 2001.

[10] Stephen L Mayo, Barry D Olafson, and William A Goddard. Dreiding: a generic force field for molecular simulations. *Journal of Physical chemistry*, 94(26):8897–8909, 1990.

[11] Junmei Wang, Romain M Wolf, James W Caldwell, Peter A Kollman, and David A Case. Development and testing of a general amber force field. *Journal of computational chemistry*, 25(9):1157–1174, 2004.

[12] Frank H Stillinger and Thomas A Weber. Computer simulation of local order in condensed phases of silicon. *Physical review B*, 31(8):5262, 1985.

[13] Murray S Daw and Michael I Baskes. Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals. *Physical Review B*, 29(12):6443, 1984.

[14] Georg Kresse and Jürgen Hafner. Ab initio molecular dynamics for liquid metals. *Physical Review B*, 47(1):558, 1993.

[15] Jörg Behler and Michele Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical review letters*, 98(14):146401, 2007.

[16] Albert P Bartók, Mike C Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Physical review letters*, 104(13):136403, 2010.

[17] Jonathan Vandermause, Steven B Torrisi, Simon Batzner, Alexie M Kolpak, and Boris Kozinsky. On-the-fly bayesian active learning of interpretable force-fields for atomistic rare events. *arXiv preprint arXiv:1904.02042*, 2019.

[18] Jörg Behler. Constructing high-dimensional neural network potentials: A tutorial review. *International Journal of Quantum Chemistry*, 115(16):1032–1050, 2015.

[19] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24):241722, 2018.

[20] Linfeng Zhang, Jiequn Han, Han Wang, Roberto Car, and E Weinan. Deep potential molecular dynamics: a scalable model with the accuracy of quantum mechanics. *Physical review letters*, 120(14):143001, 2018.

[21] Gabriele C Sosso, Giacomo Miceli, Sebastiano Caravati, Jörg Behler, and Marco Bernasconi. Neural network interatomic potential for the phase change material gete. *Physical Review B*, 85(17):174103, 2012.

[22] Nongnuch Artrith, Tobias Morawietz, and Jörg Behler. High-dimensional neural-network potentials for multicomponent systems: Applications to zinc oxide. *Physical Review B*, 83(15):153101, 2011.

[23] Nongnuch Artrith and Jörg Behler. High-dimensional neural network potentials for metal surfaces: A prototype study for copper. *Physical Review B*, 85(4):045439, 2012.

[24] Hagai Eshet, Rustam Z Khaliullin, Thomas D Kühne, Jörg Behler, and Michele Parrinello. Ab initio quality neural-network potential for sodium. *Physical Review B*, 81(18):184107, 2010.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[26] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[27] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

Chapter 2

Methods

2.1 Neural Networks

The computational model underlying deep learning is called a neural network. In the language of Linear Algebra, a neural network can be viewed as a series of matrix-vector multiplications (linear transformations), followed by element-wise non-linear functions, that maps an input vector $\mathbf{x}$ (a numerical reprsentation of the raw or featurized data) to a predicted output vector $\mathbf{y}$ through a series of intermediate representations.

The forward pass through a network, i.e. the evaluation of the predicted output for a given input, where $f(x)$ is a non-linear transformation, can thus be written as follows [1]:

$$h_{1,i} = f(\sum_{a \in x_i} W_{ia} x_a) \tag{2.1}$$

$$h_{2,i} = f(\sum_{b \in h_1} W_{ib} h_{1,b}) \tag{2.2}$$

$$\hat{y}_i = f(\sum_{c \in h_2} W_{ic} h_{2,c}) \tag{2.3}$$

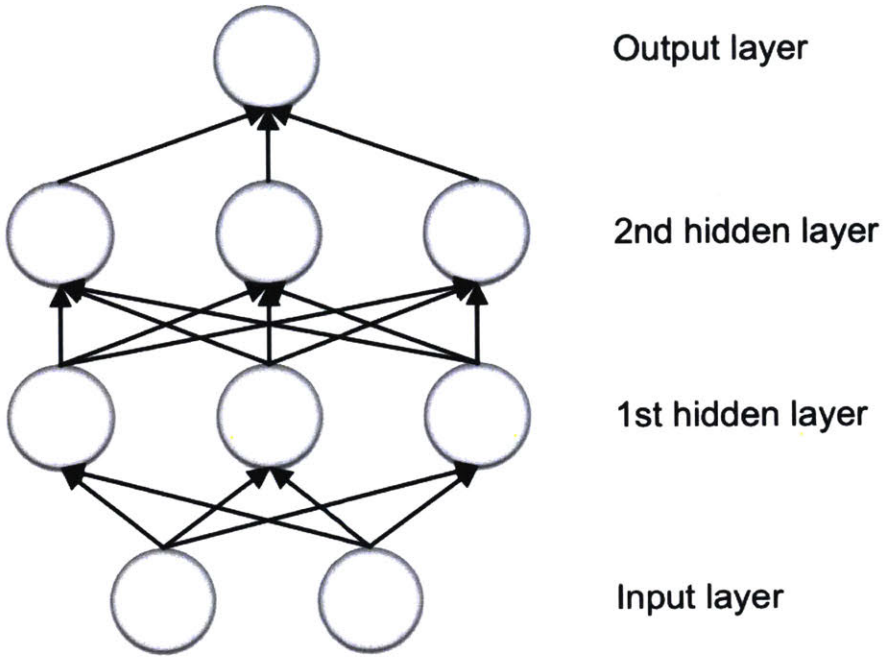

Figure 2-1: A fully-connected feed-forward neural network with an input layer, two hidden layers, and an output layer

Figure 2-1 shows the architecture of a fully-connected, feed-forward neural network: input, intermediate, and output representation are often called layers, which are composed of the most elementary computational unit of a neural networks, individual nodes or neurons.

In order to be able to accurately model the underlying data distribution, the weight matrices of the specified neural network model have to be iteratively updated to match a set of training data, measured by some loss function. When modeling regression problems, i.e. continuous dependent variables, a popular cost function is that based on an L2 loss:

$$L = \sum_{j \in data\ set} ||\mathbf{y_j} - \hat{\mathbf{y}}_\mathbf{j}||^2 \tag{2.4}$$

where y_j is the true target value given an input x_j and $\hat{y_j}$ is the output predicted by the target.

The miminization of this loss function is performed via the backpropagation algorithm [2], an application of the chain rule that allows one to iteratively compute the derivatives of the loss L with respect to the individual weights $W_{i,j}$ by multiplying a series of derivatives.

2.2 Deep Learning

Deep learning has allowed tremendous breakthroughs in a variety of fields, most notably computer vision and speech recognition [3, 4]. Recently, this computational method has also been applied to a variety of problems in the natural sciences, such as e.g. drug discovery [5], astronomy [6], and most important for the scope of this thesis, in the design of new molecules and materials [7, 8, 9]. The paradigm of deep learning is to allow neural networks to abstract representations from data without the need for pre-specifying these features (often called *handcrafting* or *engineering* of features, which is usually a time intensive process, guided by experience and human intuition). By composing several computational units in series, the resulting model is able to capture more and more complex patterns in the data. Analysis of the filters used in a convolutional neural network has shown that e.g. in image analysis, the model learns to recognize more and more complex features as information propagates through the network [10]. Due to the fact that deep neural networks learn patterns in a completely data-driven fashion without engineered features, this often results in very data-intensive algorithms. In atomistic applications, where large data sets of reference Density Functional Theory calculations are often very expensive to obtain, this problem is in some way remedied by the use of a local decomposition, in which every atomic environment comes with an atomic force label. Thereby in a system of e.g. 200 atoms, 1000 DFT calculations would result in 200,000 input-target pairs. In addition, incorporating symmetry into the model can also be seen as reducing the data requirements for machine learning models. It will be shown in this thesis, that as little as 1,000 reference calculations are enough to build force-fields for small

molecules at good accuracy and very low computational cost.

2.3 Tensor Field Neural Networks

Satisfying symmetry requirements is of very high importance in the development of interatomic force-fields. Specifically, if a target property satisfies certain invariance or equivariance properties, then the predictions of the machine learning model should satisfy the same symmetry constraints. For example, when predicting total energies, a scalar quantity that is invariant under translation of the system, rotation of the system, and permutation of atoms of the same chemical species, the predictions of the neural network should also be invariant with respect to these transformations. If symmetry requirements are not satisfied, two atomic configurations that are equivalent under a symmetry operation can result in differing predictions, thereby yielding unphysical results. While symmetry constraints can also be learned from data, e.g. by including rotated data points in the training set (often called data augmentation), this methodology is data inefficient and does not provide provable guarantees that the symmetry requirements are indeed respected. In machine-learning-based force fields, these symmetries are often enforced either by expressing the raw coordinates in a feature space that satisfies the requirements (such as e.g. the symmetry functions used in Behler-Parrinello Neural Networks [11]) or by operating on interatomic distances, as e.g. shown in the SchNet approach [9]:

$$||\mathbf{r_i} - \mathbf{r_j}|| \tag{2.5}$$

When predicting atomic forces directly, the target property is now equivariant (often also called covariant) with respect to rotation and translation of a system of atoms. While translational equivariance can easily be achieved by operating only relative distances between atoms, the requirement of rotational equivariance is very hard to encode. To this end, Thomas, Smidt, et. al have proposed a deep learning algorithm called *Tensor Field Neural Networks* (TFNNs), that specifically enables the construction of a fully rotationally and translationally equivariant neural network

[12]. It should also be noted that a very similar method called *3D Steerable CNNs* has also recently been proposed by Weiler, Geiger et. al [13].

The model in the framework of Tensor Field Neural Network operates on a set of arbitrary points in $\mathbb{R}^3$ (i.e. inputs are not restricted to a grid) with associated features at every point and inputs and outputs geometric tensors in every layer, i.e. scalars, vectors, or higher-order tensors [12]:

$$\mathcal{L}(\mathbf{r}_a, \mathbf{x}_a) = \mathbf{y}_a \tag{2.6}$$

In the following, the core aspects of the model will be introduced. For a more extensive analysis, the reader is refered to [12].

The requirement of a function $\mathcal{L} : \mathcal{X} \to \mathcal{Y}$ being covariant with respect to a symmetry operation can be expressed as follows [12]:

$$\mathcal{L} \circ D^{\mathcal{X}}(g) = D^{\mathcal{Y}}(g) \circ \mathcal{L} \tag{2.7}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are vector spaces, $D^{\mathcal{X}}$ and $D^{\mathcal{Y}}$ are group representations, and g is an element of some group G. Since a composition of two equivariant functions yields an equivariant function, it is enough to build a network out of different types of layers, each of which are equivariant. In the proposed TFNN approach, there are three kinds of layers: point-wise convolutions, self-interaction layers (similar to those introduced in SchNet) and non-linearities. All three individually satisfy the required symmetry constraints and will be discussed in further detail below. In addition, if a function is equivariant to g, $h \in G$, then it is also equivariant to their composition. Therefore, it is enough to show that each layer satisfies the symmetries of translational, permutational, and rotational covariance seperately to prove that the layer (and therefore the network) respects their composition [12].

The layers satisfy translational equivariance by only operating on relative distances

between two atoms:

$$\mathcal{L}(\mathbf{r_i} - \mathbf{r_j}) \tag{2.8}$$

In addition, permutational equivariance is satisfied by never enforcing an order on a list, and instead only acting on a *set* of atoms. Finally, the layers satisfy rotational equivariance in different ways for the three types of layers. This will be discussed in further detail below.

Each layer acts on a set of points with associated vectors in a representation of $SO(3)$. Depending on the type of tensor, these come in different rotation orders l=0 (scalars), l=1 (vectors), l=2 (symmetric, traceless matrices), etc. Representations are expressed in irreducible representation of $SO(3)$, resulting in objects with dimension of $m = 2l + 1$. In practice, these are encoded via an object $V_{acm}^{(l)}$ that is acted on throughout the network [12]. Here:

- l corresponds to tensors of different rotation orders, as described above
- a is a point index (running over all points in the set of atoms)
- c is a channel index, allowing for multiple channels to be associated with each rotation order (for example, a V-object could carry i different l=0 scalars, j different l=1 vectors, and k different l=2 tensors)
- m is the representation index of length $m = 2l + 1$

Figure 2-2 shows a simple example of two atoms associated with features of different order: atomic numbers Z_i (l=0), atomic masses m_i (l=0), and velocities v_i (l=1). The way that individual atoms incorporate interactions with all other atoms is based on point-wise convolutional layers. These follow a design very similar to the SchNet convolutional layers outlined above, however with the additional property of being equivariant. In order to design a fully equivariant layer, the filters are of the following form [12]:

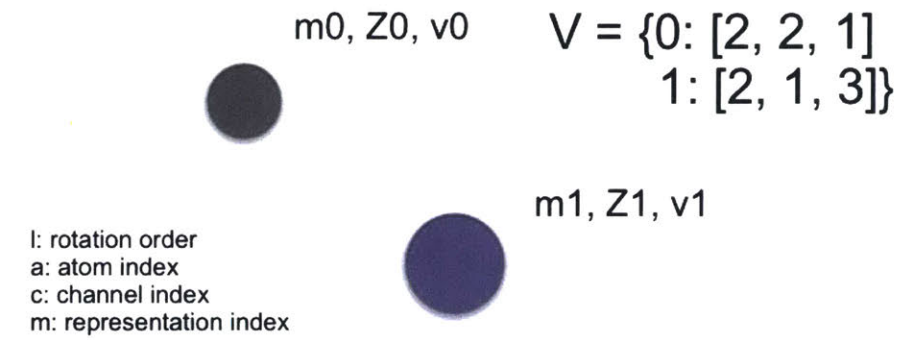

Figure 2-2: Two atoms associated with features of different rotation order. The V object has l=0 and l=1 rotation orders associated with it, the list shows the shape: there are two atoms, two l=0 properties vs one l=1 property, and finally m=1 in the l=0 (scalar) case and m = 3 in the l=1 (vector) case. Figure created based on similar display in [12].

$$F_{cm}^{(l_f,l_i)}(\mathbf{r}) = R_c^{(l_f,l_i)}(r)Y_m^{(l_f)}(\hat{\mathbf{r}}) \tag{2.9}$$

where r is the magnitude of the interatomic distance $\mathbf{r}$, $\hat{\mathbf{r}}$ is the distance vector normalized to unit length, $R_c^{l_f,l_i} : \mathbb{R}_{\geq 0} \to \mathbb{R}$ are radial functions, that contain the learned weights, and $Y_m^{(l_f)}$ are spherical harmonics of order l. Since spherical harmonics are the basis functions of the irredubicle representation of $SO(3)$, they are equivariant to $SO(3)$. In addition, the radial functions used here act as scalars in m and therefore, the final filter as a product of the two is also rotationally equivariant.

Since both the filter and its inputs are expressed in terms of irreducible representations of $SO(3)$, they must be combined in an equivariant fashion. In the framework of TFNNs, this is done via a tensor product of representations, an equivariant operation, which in practice is computed using Clebsch-Gordan coefficients C. This finally yields the following definition of a pointwise convolutional layer [12]:

$$\mathcal{L}_{acm_o}^{(l_o)}(\mathbf{r}, V_{acm_i}(l_i)) = \sum_{m_f,m_i} C_{(l_f,m_f)(l_i,m_i)}^{l_o,m_o} \sum_{b\in S} F_{cm_f}^{l_f,l_i}(\mathbf{r}_{ab})V_{bcm_i}^{(l_i)} \tag{2.10}$$

Similar to SchNet, the convolution layers only update the individual elements

of the feature vectors, and a refinement within a feature vector is obtained by *self-interaction* layers, defined as follows [12]:

$$\sum_{c'} W cc'^{(l)} V^{(l)}_{ac'm} \tag{2.11}$$

The final layer are nonlinearities, which only act on the norm of the representation vector.

2.4 Learning Scheme for obtaining direct forces from TFNNs

In conventional energy-based schemes (i.e. models that output a total energy, e.g. the Behler-Parrinello approach or the SchNet model), equivariant forces are obtained by computing the derivative of the scalar potential energy. However, these derivative come with a series of computational overheads:

- While *running* a molecular dynamics simulation, forces are needed at every time step, which in practice means computing a forward pass through the neural network, followed by an additional backward pass through the network at every time step the simulation.

- While *training* the neural network, first, a forward pass is computed to obtain the potential energy. Subsequently, a first backward pass is computed to obtain the predicted atomic forces. In order to obtain good force estimate, these predicted forces are then added to the loss function, as show in equation 1.8, therefore requiring an additional backward pass.

- Finally, in descriptor-based approaches: an additional set of derivatives is computed, since the atomic coordinates are first transformed into a set of feature functions, resulting in an additional set of derivatives $\frac{\partial G_{j,k}}{\partial r_i}$, as shown in equation 1.6.

To move towards more efficient force-fields that allow for the simulation or larger systems over longer time scales, this thesis explores the direction prediction of atomic forces using Tensor-Field Neural Networks [12]. Through the TFNN framework, the symmetries of translational and most importantly rotational equivariance are satisfied, thus bypassing the need for predicting total energies. The mapping to be learned by the Direct Force Prediction (DFP) algorithm can be written as follows:

$$\mathbf{F}(\{\mathbf{r_i}\}, \{Z_i\}) : \{R^3, Z^+\}^N \rightarrow \{R^3\}^N : \tag{2.12}$$

In this scheme, only one backward pass is required during training and *no backward pass* at time of prediction, since the forces are obtained directly from atomic positions and atomic numbers.

In addition to the aspect of computational efficiency, in the proposed model, atomic coordinates are not expressed in feature functions, but instead the network directly predicts on relative atomic positions and atomic numbers. This follows the spirit of deep learning, where the aim is that the model should extract the relevant features from the data without the need for human feature engineering. Furthermore, since different chemical species simply correspond to different atomic numbers used in the input, the model naturally handles multiple species. In the scope of this thesis, up to four difference chemical species will be handled by a single network (C, H, O, N). An overview of the proposed model is show in figure 2-3.

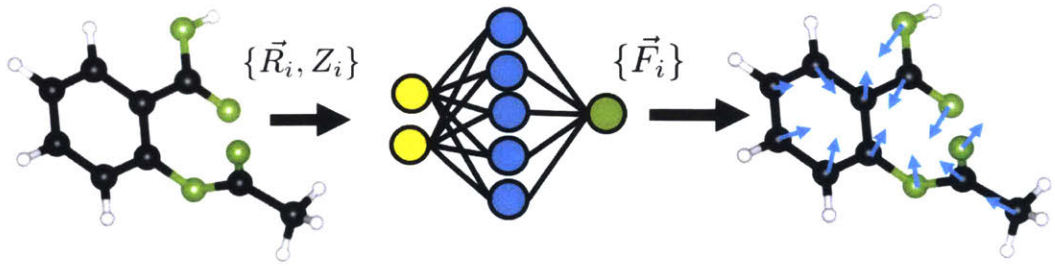

Figure 2-3: The proposed learning scheme: atomic forces, i.e. vector quantities, are being predicted directly from atomic positions and atomic numbers.

Bibliography

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

[2] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[4] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[5] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072*, 2015.

[6] Christopher J Shallue and Andrew Vanderburg. Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(2):94, 2018.

[7] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[8] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[9] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24):241722, 2018.

[10] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[11] Jörg Behler. Constructing high-dimensional neural network potentials: A tutorial review. *International Journal of Quantum Chemistry*, 115(16):1032–1050, 2015.

[12] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[13] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pages 10381–10392, 2018.

Chapter 3

Results

In order to assess the suitability of using the proposed model in molecular dynamics simulations, results regarding the model's accuracy on a data set of small organic molecules, the MD-17 data set [1, 2, 3], are discussed in this chapter. In addition, the model's computational efficiency both during training and at time of prediction is measured and compared to that of the SchNet approach. Finally, MD trajectories using the trained direct prediction model as the underlying force-field are analyzed and discussed.

3.1 The MD-17 data set of small organic molecules

The MD-17 data set contains ab-initio molecular dynamics (AIMD) trajectories of eight organic molecules with up to four different chemical elements (C, H, N, O): benzene, uracil, naphthalene, aspirin, salicylic acid, malonaldehyde, ethanol, and toluene. Each MD trajectory was computed at T = 500K, a time step of 0.5 fs and at the PBE+vdW-TS level of accuracy. The trajectories contain between 133,770 and 993,237 different structures. The results will be compared to the SchNet model to provide a baseline for accuracy and efficiency.

3.2 Accuracy

The model was trained on two different subsets of the full data set: one subset containing 1,000 configurations and a second subset containing 50,000 configurations. In both cases, a validation set of 1,000 structures was used. The validation set is used to give an estimate of how well the model generalizes to unseen data with the purpose of prohibiting overfitting of the model's parameters to the training data. The accuracy on both the training and the validation set was computed and recorded every 5 training epochs. If the validation set accuracy, measured by the mean absolute error of the force components, was better than all previous accuracies, the model's current parameters were saved to disk to overwrite the previous best model.

For the 1,000 structures data set, the model was then trained for a maximum of 100,000 epochs or one week on a Tesla K80 GPU. In addition, an early stopping criterion was used: if there was no improvement in the force MAE on the validation set for 2500 epochs, the training was stopped. A batch size of 5 molecules per batch was used together with a starting learning rate of 0.0005, exponential decay of the learning rate and a network using 15 radial basis functions with means evenly spaced between 0.0 and 2.5. In total, four layers with dimension 30 were used. The full network architecture used was as follows: a self-interaction layer was used for the embedding of the chemical species, followed by a series of convolution-interaction-nonliearity blocks. This pattern was repeated three times until the final layer, in which only a convolution followed by a self-interaction layer and no nonlinearity was used. Thus in total, the network consists of four convolutional layers. Information propagates through the different convolutional layers, which can be viewed as each atom updating its state based on all other atoms around it.

Using the configuration specified above, the results on the 1k data set are shown in table 3.1. The accuracy displayed measures the mean absolute error of the force components on all structures not included in the training and validation set (floored

to the nearest integer multiple of 100). The accuracies are compared to those obtained for the SchNet model, as reported in [4].

Molecule	Direct Prediction	SchNet
Benzene	**0.19**	0.31
Toluene	**0.39**	0.57
Malonaldehyde	1.00	**0.66**
Salicylic acid	**0.78**	0.85
Aspirin	**1.09**	1.35
Ethanol	0.73	**0.39**
Uracil	0.60	**0.56**
Naphthalene	**0.30**	0.58

Table 3.1: Mean absolute erorrs in [kcal/(mol Angstrom)] on the test set, using a training set of 1,000 configurations and a validation set of 1,000 configurations, compared to the SchNet results reported in [4]. Best models in bold.

For the larger data set, consisting of 50,000 training structures, the model was trained for a maximum of 100,000 epochs or six days and six hours on a Tesla K80 GPU. Again, an early stopping criterion was used. A larger batch size of 50 molecules per batch was used together with a larger starting learning rate of 0.001, exponential decay of the learning rate and a network using 15 radial basis functions with means evenly spaced between 0.0 and 2.5. In total, seven layers with dimension 250, 200, 150, 150, 100, 100, 50 composed the networks. The same scheme of a series convolution-self-interaction-nonlinearity blocks were used. Using this training configuration, the results on the 50k data set are shown in table 3.2. Again, the mean absolute error of the force components on all structures not included in the training and validation set was computed (floored to the nearest integer multiple of 10). It can be seen that on both the training subset of 1,000 as well as on the larger training subset of 50,000 structures, the direct prediction approach obtains high accuracy, often outperforming the SchNet accuracies, as they were reported in [4].

Molecule	Direct Prediction	SchNet
Benzene	**0.16**	0.17
Toluene	**0.08**	0.09
Malonaldehyde	**0.08**	**0.08**
Salicylic acid	**0.16**	0.19
Aspirin	**0.24**	0.33
Ethanol	0.06	**0.05**
Uracil	**0.06**	0.11
Naphthalene	**0.06**	0.11

Table 3.2: Mean absolute errors on the test set in [kcal/(mol Angstrom)], using 50,000 training configurations and 1,000 validation configurations, compared to the SchNet results reported in [4]. Best models in bold.

3.3 Efficiency

In addition to reproducing the true atomic forces as accurately as possible, computational efficiency is a second very important aspect in the development of interatomic force fields. To measure the efficiency of the proposed scheme, the wall time of both the prediction as well as the time required to train the networks was measured. Since the underlying force model has to be queried at every time step, the efficiency of prediction easily becomes the bottleneck in molecular dynamics simulations. When developing machine-learning based force fields however, the training process often additionally amounts for a significant portion of the total time required. Therefore, efficient training is of special importance in developing ML-based force fields. In the following, the results from direct prediction are compared to those using the SchNet model. Being an energy-based model, in order to compute the forces, a forward pass has to be computed to obtain the global energy, followed by a backward pass to obtain the atomic forces. During training, a forward pass is required to obtain the global energy, a backward pass is run to get the predicted forces, these are subsequently added to the loss function, and finally, a second backward pass needs to be computed in order to obtain the weight updates based on the loss.

3.3.1 Prediction Efficiency

Tables 3.3 and 3.4 compare the computational efficiency of inference for the two models on two molecules of different sizes, averaged over 500 evaluations, on a CPU and a GPU system, respectively. In both cases, it is observed that the direct prediction approach yields an efficient model for fast prediction of atomic forces. It should be noted that the Direct Prediction approach uses a TensorFlow implementation [5] for the neural network, whereas the SchNet approach is implemented using the PyTorch framework [6] for deep learning. Additional experiments are required to better understand the behavior of the computational efficiency of the two approaches.

Molecule	Direct Prediction	SchNet
Asprin (21 atoms)	**0.0070**	0.091
Benzene (12 atoms)	**0.0041**	0.042
Ethanol (9 atoms)	**0.0038**	0.027

Table 3.3: Efficiency of Inference on CPUs in [sec]

Molecule	Direct Prediction	SchNet
Asprin (21 atoms)	**0.0079**	0.022
Benzene (12 atoms)	**0.0076**	0.022
Ethanol (9 atoms)	**0.0073**	0.024

Table 3.4: Efficiency of Inference on GPUs in [sec]. Both models were evaluated on Tesla K80 GPUs.

3.3.2 Training Efficiency

Table 3.5 shows the efficiency for training for both models in minutes as well as the resulting relative computational speed-up of using direct prediction in comparison to the energy-based SchNet approach. Both models are trained on a training set of 1,000 structures and a validation set of 1,000 structures. The SchNet model is trained using the SchNetPack software package [7]. It should be added that the direct prediction model has approx. 71x fewer parameters than the SchNet model. In addition, different

learning rates, different schemes for updating the learning rate, and different batch sizes are used in the respective models, which influence training: the parameters in the direct prediction model are those that have been also used for obtaining the above accuracy results. For the SchNet model, the architecture and training parameters were chosen based on those given in [7] and the default SchNetPack configurations (table 3.6 details the full training configuration used for training of the SchNet model). As shown above, the direct prediction approach outlined in this thesis uses architectures of different complexity for the smaller and larger training sets. The architecture for the smaller training set of 1,000 structures has fewer weights than that used for the larger training set of 50,000 structures. For measuring training and prediction efficiency in the direct prediction approach, the small network architecture consisting of four 30-element layers was used. Due to these differences, additional experiments regarding the dependence of computational efficiency of both training and prediction on the number of weights in the two networks and the different training settings chosen are required.

Molecule	Direct Prediction	SchNet	Relative Speed-up
Ethanol (9 atoms)	**174**	1893	11
Benzene (12 atoms)	**9**	2957	329
Toluene (15 atoms)	**22**	2118	96
Naphthalene (18 atoms)	**25**	3463	139
Aspirin (21 atoms)	**8**	995	124

Table 3.5: Efficiency of Training in [minutes]. Both models were trained on a training set of 1000 samples and validation set of 1000 samples. The times shown represent the times it took each model to obtain an accuracy within 10% of the worse-performing model. This metric was chosen in order not to measure the final stretch of the training cycle, which usually only gives minor improvements in accuracy, but requires a significant portion of the overall training time. It should be noted that the SchNet model for Aspirin did not converge below a force MAE of 3.22 kcal/(mol Angstrom). All models were trained on Tesla K80 GPUs.

3.3.3 SchNet Training Parameters

The SchNet models were trained using the SchNetPack software [7]. The parameters used for training the SchNet model are shown in table 3.6.

Parameter	Value
Training Set Size	1000
Validation Set Size	1000
Initial Learning Rate	0.0001
Decay Factor	0.5
Minimal Learning Rate	1e-06
Patience	150
Batch Size	100
Rho	0.1
Cutoff	5.0
Features	256
Interactions	6
Number of Gaussian	25

Table 3.6: Training parameters used for SchNet training on MD-17. The parameters were chosen according to those given in [7]. Hyperparameters not specified in [7] were set according to the default settings in SchNetPack.

3.4 Molecular Dynamics Simulations

To gain a better understanding of the learned force-fields, the trained models were finally used as the underlying force models for molecular dynamics simulations of different small molecules. This section will discuss the results of these simulations for the three molecules toluene, benzene, and naphthalene.

In order to run molecular dynamics simulations, the model was coupled to the Atomic Simulation Environment [8], a Python package specifically designed for atomistic simulations. The ASE framework was used in order to make use of the MD capabilities given in this package, including specifically the use of thermostats. A question in the development of force fields that *directly* predict forces is that of energy conservation. In order to investigate the effects of this, no thermostat was used in the

final simulation, but instead the system was evolved using the Verlet integration algorithm. For equilibration of the system, however, a target temperature was chosen, and a Langevin thermostat was used to equilibrate the system in a NVT ensemble at a certain temperature. The thermostat was run in two different stages: first, with a friction coefficient of 1.0 for a duration of 1,000 time steps (0.5 ps), then with a friction coefficient of 0.1 for a duration between 1,000 and 2,000 time steps (0.5 ps to 1 ps). In the period between 1,000 and 2,000 frames, as soon as the temperature of the simulation was within +/- 5% of the target temperature, the thermostat was turned off and the run was continued using Verlet integration and no further control of temperature. All simulations were run with the time step that was used in the simulation of the original AIMD trajectory, $\Delta t = 0.5fs$. During the NVT part of the simulation, the positions and the center of mass were kept constant while the conservation of angular momentum was not explicitly enforced. During the subsequent simulation without a thermostat, however, at every time step, both the center of mass momentum as well as the total angular momentum was set to zero. This procedure was repeated for the molecules toluene, benzene, and napthalene, at target temperatures of T = 100 K, 200 K, and 300 K. It should be noted that due to the outlined procedure, the final simulation at which the temperature was started, did not precisely match these target temperatures. The temperatures and forces shown in the following paragraphs were parsed from the simulation after the system was equilibrated, i.e. they do not include the equilibration phase. The first 1,000 frames after equilibration are shown.

3.4.1 Toluene

Target temperature T = 100 K

Figure 3-1 shows the behavior of the temperature of the simulation as well as the mean magnitude of the absolute atomic forces as a function of time at a target temperature of T = 100 K. It can be seen that while the force magnitudes seem to oscillate around

a constant mean, there is a drift in temperature as the simulation evolves.

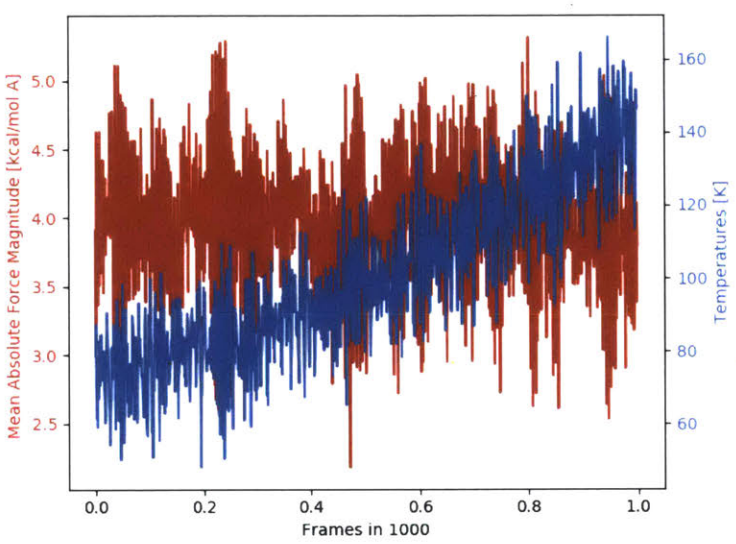

Figure 3-1: Temperatures and mean force magnitudes of a DFP-enabled simulation of toluene with target temperature T = 100 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

Target temperature T = 200 K

Figure 3-2 shows the behavior of the temperature and mean force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T = 200 K. Again, an increase in the temperatures is observed.

Target temperature T = 300 K

Figure 3-3 shows the behavior of the temperatures and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T = 300 K. Here, a much more pronounced increase is observed. This is due to the breaking of a molecular bond during the simulation, at which point, the simulation sharply diverges, resulting in the observed temperature increase. This is accompanied by a matching sharp spike in the forces.

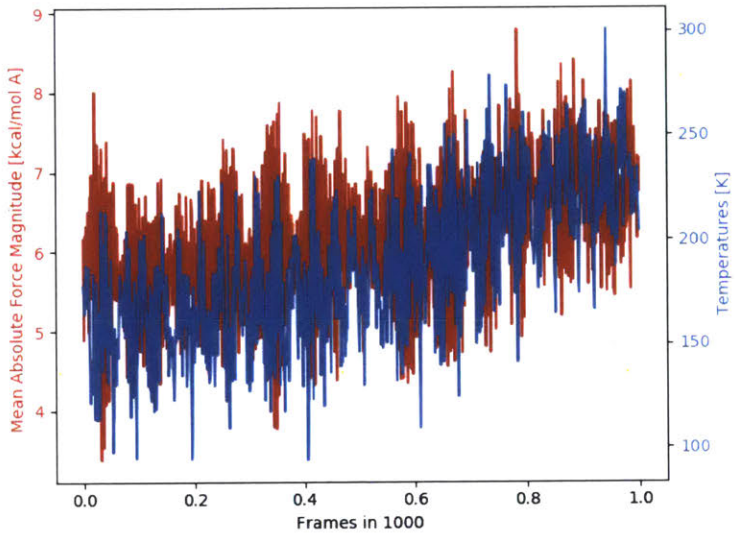

Figure 3-2: Temperatures and mean force magnitudes of a DFP-enabled simulation of toluene with target temperature T = 200 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

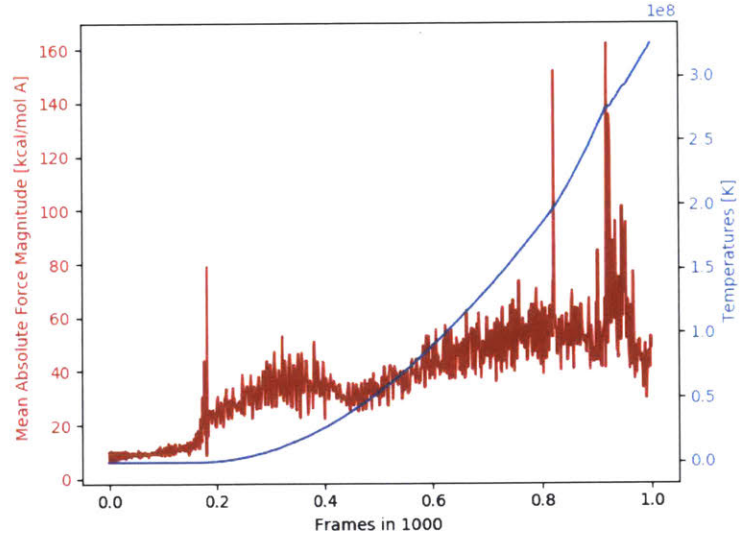

Figure 3-3: Temperatures and mean force magnitudes of a DFP-enabled simulation of Toluene with target temperature T = 300 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

3.4.2 Benzene

Target temperature T = 100 K

Figure 3-4 shows the behavior of the temperature and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=100K for the benzene molecule. The temperatures appear to be more stable in this simulation.

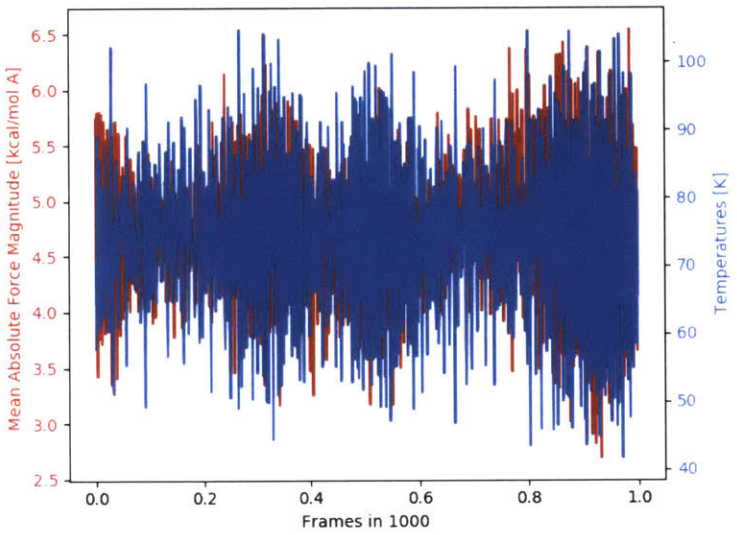

Figure 3-4: Temperatures and mean force magnitudes of a DFP-enabled simulation of benzene with target temperature T = 100 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

Target temperature T = 200 K

Figure 3-5 compares the behavior of the temperatures and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=200K. Again, the temperatures appear to be more stable.

Target temperature T = 300 K

Figure 3-6 shows the behavior of the temperatures and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=300K.

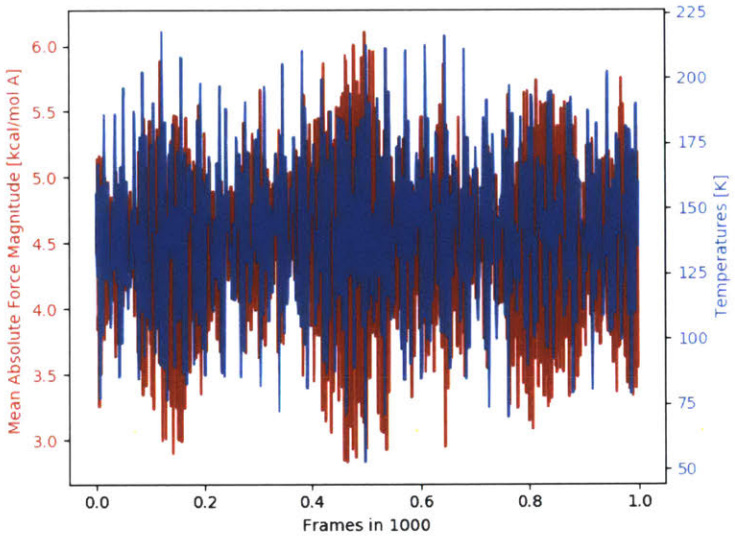

Figure 3-5: Temperatures and mean force magnitudes of a DFP-enabled simulation of benzene with target temperature T = 200 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

Similar to the case of T = 300 K for toluene, the simulation diverges within the first 1,000 steps.

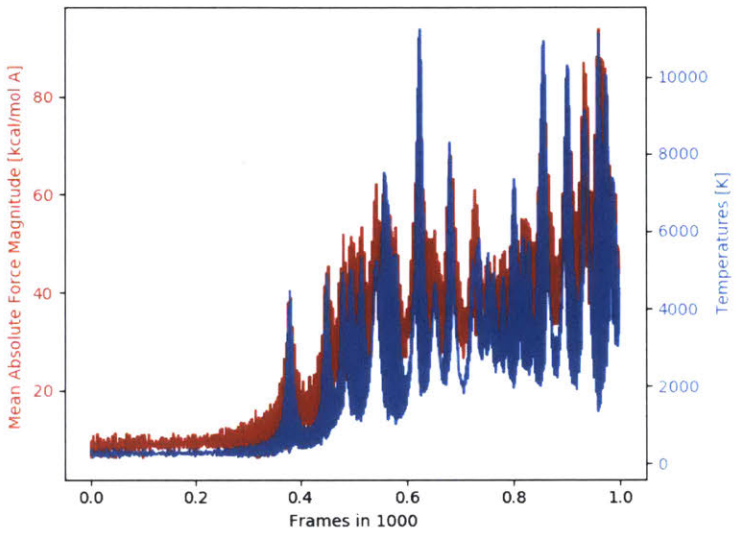

Figure 3-6: Temperatures and mean force magnitudes of a DFP-enabled simulation of benzene with target temperature T = 300 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

3.4.3 Naphthalene

Target temperature T = 100 K

Figure 3-7 shows the behavior of the temperatures and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=100K.

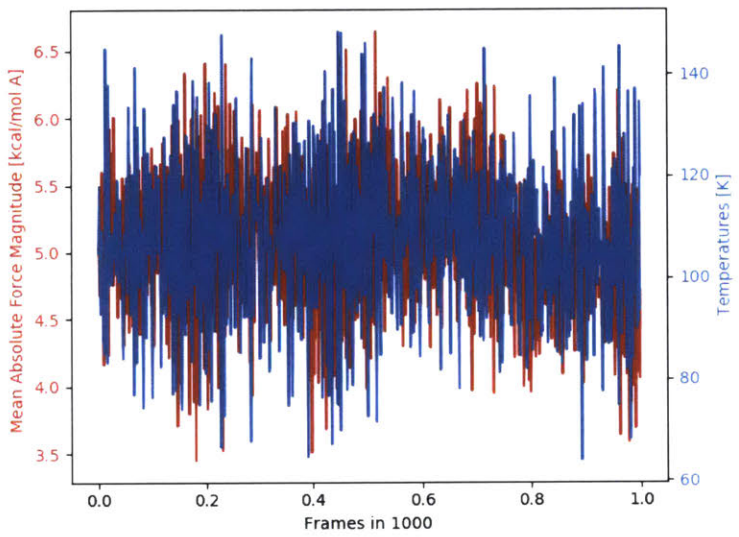

Figure 3-7: Temperatures and mean force magnitudes of a DFP-enabled simulation of naphthalene with target temperature T = 100 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

Target temperature T = 200 K

Figure 3-8 shows the behavior of the temperature and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=200K.

Target temperature T = 300 K

Figure 3-9 shows the behavior of the temperatures and force magnitudes of the simulation vs. the frame of the MD simulation at a target temperature of T=300K.

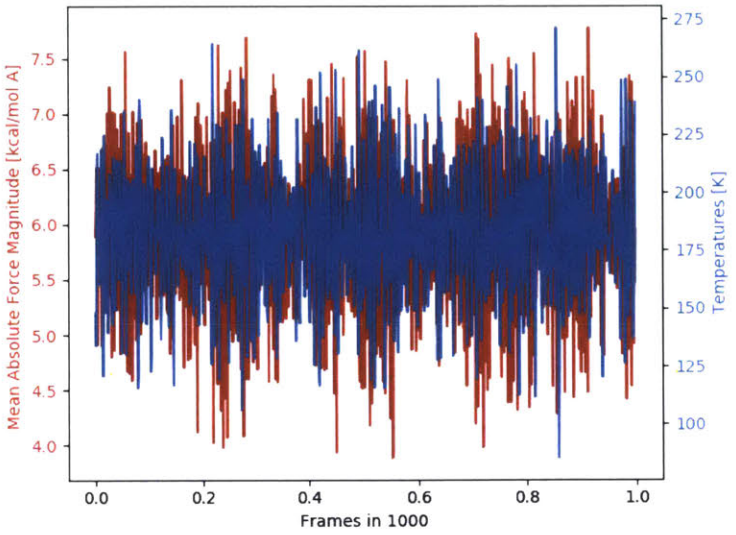

Figure 3-8: Temperatures and mean force magnitudes of a DFP-enabled simulation of naphthalene with target temperature T = 200 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

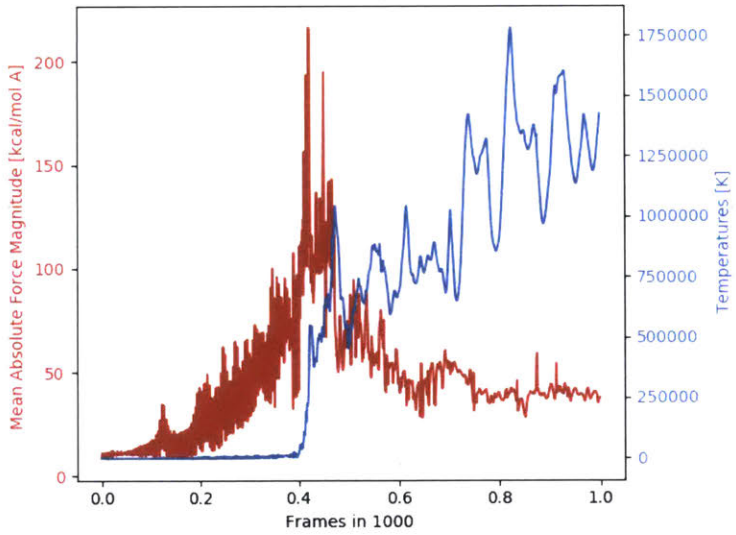

Figure 3-9: Temperatures and mean force magnitudes of a DFP-enabled simulation of naphthalene with target temperature T = 300 K and a molecular dynamics time step of $\Delta t = 0.5$ fs.

The results obtained from deploying the trained force-field model in molecular dynamics simulations of different molecules at different temperatures suggest further study of the divergent behavior. Specifically, the extent to which the model's predictions can be monitored, e.g. via the use of predictive uncertainty estimates or from an ensemble of models, could provide additional insights.

Bibliography

[1] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017.

[2] Stefan Chmiela, Huziel E Sauceda, Klaus-Robert Müller, and Alexandre Tkatchenko. Towards exact molecular dynamics simulations with machine-learned force fields. *Nature communications*, 9(1):3887, 2018.

[3] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8:13890, 2017.

[4] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems*, pages 991–1001, 2017.

[5] 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, Yangqing Jia, 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, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[6] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[7] KT Schu¨tt, Pan Kessel, Michael Gastegger, KA Nicoli, Alexandre Tkatchenko, and K-R Mu¨ller. Schnetpack: A deep learning toolbox for atomistic systems. *Journal of chemical theory and computation*, 15(1):448–455, 2018.

[8] Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer,

Cory Hargus, et al. The atomic simulation environment—a python library for working with atoms. *Journal of Physics: Condensed Matter*, 29(27):273002, 2017.

Chapter 4

Conclusions and Outlook

4.1 Conclusions

In this thesis, it has been shown that the direct prediction of atomic forces provides a computationally efficient route for modeling interatomic force-fields at high accuracy. The proposed model respects the symmetries of rotation, translation as well as permutation, and can obtain high accuracy for different sizes of reference data sets. The approach has been tested on a data set of small organic molecules and shows both high accuracy and computational efficiency.

In addition, the proposed model has been used as the underlying force-field in several different Molecular Dynamics simulations of small molecules. In simulations without the control of a thermostat, it is observed that the trajectories are unstable. Further analysis of this behavior is required.

4.2 Outlook

A more detailed investigation of the influence of the lack of momentum and energy conservation on the resulting Molecular Dynamics simulations is required. In addition, an extension of the model to extended materials is likely to provide additional insights. To this end, the use of a local cutoff should be implemented in order to

enable scaling to large atomistic sytems. Furthermore, the use of an approximate uncertainty metric in the model's prediction should be studied: for example, the use of an ensemble of neural networks, simulatenously running in a Molecular Dynamics simulations could provide a measure of the predictice uncertainty of the model. In such a scenario, the variance in the ensemble's predictions could be used to gauge the trained model's confidence in its prediction and in case the uncertainty is high, a high-accuracy reference method, such as DFT could be queried. Most importantly, however, it will have to be tested, whether the resulting trajectories, e.g. under a NVT ensemble, can reproduce observable properties of the true AIMD simulations, such as e.g. kinetic and thermodynamic properties.