UHASSELT

KNOWLEDGE IN ACTION

Faculteit Wetenschappen

master in materiomics

Masterthesis

Explorative dive into machine learned modeling of interatomic potentials and molecular dynamics of H-vacancy defects in diamond

Eleonora Thomas

Scriptie ingediend tot het behalen van de graad van master in materiomics

PROMOTOR :

Prof. dr. dr. Danny VANPOUCKE

2024
2025

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT

KNOWLEDGE IN ACTION

Faculteit Wetenschappen

master in materiomics

Masterthesis

Explorative dive into machine learned modeling of interatomic potentials and molecular dynamics of H-vacancy defects in diamond

Eleonora Thomas

Scriptie ingediend tot het behalen van de graad van master in materiomics

PROMOTOR :

Prof. dr. dr. Danny VANPOUCKE

Explorative dive into machine learned modeling of interatomic potentials and molecular dynamics of H-vacancy defects in diamond

Eleonora H.J.J. Thomas[1,], E. Aylin Melan[1,2,3], Danny E.P. Vanpoucke [1,2]*

[1] Hasselt University, Institute for Materials Research (imo-imomec), Quantum & Artificial inTelligence design Of Materials (QuATOMs), Martelarenlaan 42, B-3500 Hasselt, Belgium

[2] imec, imo-imomec, Wetenschapspark 1, B-3590 Diepenbeek, Belgium

[3] University of Namur, Namur Institute of Structured Matter (NISM), Rue de Bruxelles 61, 5000 Namur, Belgium

E-mail: eleonora.thomas@student.uhasselt.be

Keywords: machine learning, interatomic potentials, diamond defect, density functional theory, molecular dynamics

Abstract English

Vacancy defects in diamond, such as the nitrogen-vacancy (NV) centers, play an important role for quantum applications, and a deeper understanding of their dynamics is necessary to improve quality and understanding. The controlled creation of vacancy defects in diamond experimentally is time-consuming and expensive. Alternately, these structures can be created computationally, with density functional theory (DFT), which requires a lot of time and computational power. The next solution is sought in machine learning (ML). Specifically, in the creation and implementation of an ML model, based on interatomic potentials, to predict behavior and molecular dynamics (MD). This study establishes a proof of principle for a working ML model trained on *ab initio* DFT calculations. Specifically for hydrogen-vacancy (HV) defects, as a stepping stone towards understanding other defects. Through exploration of ML Interatomic Potentials (MLIP)s, an initial model is made for predicting forces and energy of HV defects in diamond, resulting in the final model having an accuracy of 8.8 meV atom^{-1} (MAE) for the energies. With these final models, MD simulations are run, proving that the MLIPs can be applied to MD, thereby adding a stepping stone for further research into MLIPs for diamond defects for quantum applications.

Abstract Dutch

Vacancy defecten in diamant, zoals nitrogen-vacancy (NV) centers, spelen een belangrijke rol voor kwantum applicaties. Dieper inzicht in hun dynamica is essentieel om kwaliteit en begrip te verbeteren. De experimenteel gecontroleerde creatie van vacancy defecten in diamant is tijdrovend en duur. Alternatief kunnen deze structuren computationeel gecreëerd worden, met density functional theory (DFT), wat ook veel tijd en computationele kosten vergt. De volgende oplossing wordt gezocht in machine learning (ML). Specifiek, in het creëren en implementeren van een ML model, gebaseerd op interatomaire potentialen, om het gedrag en de moleculaire dynamica (MD) te voorspellen. Deze studie bewijst dat het mogelijk is om een werkend ML model te creëren, getraind op *ab initio* DFT calculaties. Specifiek voor de waterstof-vacancy (HV) defecten, als eerste stap richting comprehensie van andere defecten. Door exploratie van ML Interatomaire Potentialen (MLIP)s, is een initieel model gecreëerd voor het voorspellen van de krachten en energieën van HV defecten in diamant, resulterend in een accuraatheid van 8.8 meV $atom^{-1}$ (MAE) voor de energieën. Met deze finale modellen worden MD simulaties gerund, die bewijzen dat de MLIPs toegepast kunnen worden voor MD, en hiermee een startpunt bieden voor verder onderzoek in MLIPs voor diamant defecten voor kwantum applicaties.

1. Introduction

Defects in diamond are known as promising candidates for a variety of applications, such as quantum sensing, magnetometry, and quantum computing.[1,2] Diamond in itself has unique mechanical, physical, chemical, and engineering properties.[3] The introduction of defects into diamond, however, drastically changes these properties, and leaves us with new research opportunities.

A defect is an imperfection in the crystal structure. In the case of diamond, this can be, for example, a missing carbon atom, other elements that have intruded in the structure, or a dislocation in the diamond structure. These defects can arise in several ways, such as during the process of growing diamond. All defects lead to a change in the electronic structure of the system. One type of defect is a color center, which is a point defect where an impurity in the diamond lattice impacts the way light is absorbed by diamond, resulting in a change of color, different from the colorless pristine diamond.[4] Current diamond research focusses mainly on the nitrogen-vacancy (NV) center, which is a color center. Here a carbon atom is replaced by a nitrogen atom, and an additional neighboring carbon atom is removed. These NV centers have

many interesting applications due to their optical and spin properties.[5] The hydrogen-vacancy (HV) center, is another type of defect where a carbon atom is replaced by a hydrogen atom, and an additional neighboring carbon atom is removed.[6] HV centers in itself are not commonly used for quantum applications, since the NV center is a more popular defect with a great variety of applications. But for the creation of diamond containing NV centers, hydrogen (H) is often introduced during the diamond growth. The plasma in which diamond is grown consists of >95% H, a few percentages methane (CH_4, the carbon source) and other gasses (to introduce defects). This means that a lot of H is present on the diamond surface, and since H is an atom that easily diffuses though diamond, it is thus still present in the final diamond samples for quantum applications.[7,8] Another type of defect involving hydrogen, is an interstitial hydrogen defect, where hydrogen atoms occupy non-lattice places in the crystal structure, the 'interstitial sites'.[9,10] An X-vacancy diamond defect, with X representing another atom, is visualized in **Figure 1**.

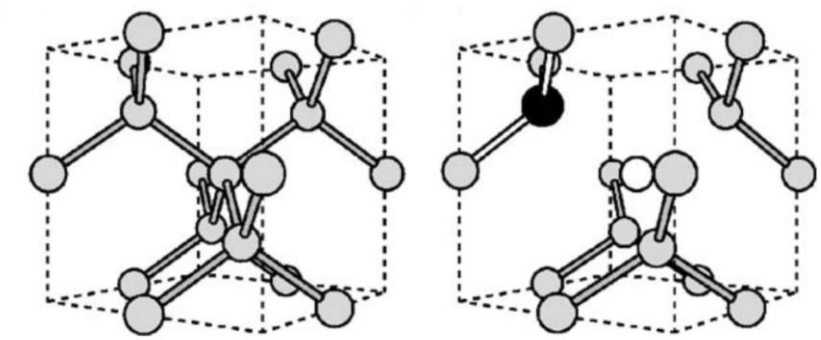

Figure 1. (left) Bulk diamond system without a defect, (right) an X-vacancy in the diamond system. The gray atoms represent the carbon atoms, the black atom represents the X-atom, and the white circle represents the vacancy. Figure from J. P. Goss, *et al.*[11]

When considering actual applications for diamond defect structures, one has to consider the quality of these structures. Experimental uses of NV centers for quantum applications are still dependent on the quality of both the created diamond and the created defects. Unwanted defects, such as oxygen that can contaminate the diamond structure, nitrogen that, while being in the structure, does not become part of an NV center, and hydrogen, a very common element during diamond growth, can lead to a lower (or higher) quality. Each element that gets introduced into the diamond structure, can lead to an impact on the properties. Other color centers, such as silicon (Si)-, germanium (Ge)-, tin (Sn)- and lead (Pb)-vacancies have also been explored with both similar and different (optical) properties to NV centers, and with varying differences in stability and excitation and emission bands.[2,12]

Synthetic diamond can be grown with a variety of techniques, such as High-Pressure and High-Temperature (HPHT), or Chemical Vapor Deposition (CVD). During the growth of synthetic diamond for quantum applications, hydrogen (H) is one of the most common impurities, besides nitrogen (N), during either HPHT or CVD.[7,13] H contributes to the properties of the grown diamond, affecting color, luminescence, growth, stability, *etc*. A detailed understanding of the exact behavior of H in diamond, however, is still needed.[9,14,15]

It is time-consuming and expensive to control the specific insertion of defects into diamond, especially considering the possible contamination with other elements into the structure. The solution, therefore, would be to create these structures computationally and to study the properties theoretically. In order to gain a deeper understanding of the processes present during experimental creation of color centers, it is crucial to study the behavior and dynamics of these defects in diamond.

This defect-behavior can be described by the Schrödinger equation. As this equation cannot be solved exactly, approximations are needed to calculate the desired values. One of such approximations being density functional theory (DFT), a computational quantum mechanical modelling method that is exact for the ground state, but reformulated in terms of the density instead of the wavefunction. DFT uses functionals of the electron density, to formulate the interatomic potentials between the atoms in a system.[16] In contrast, in the case of a (classical) force fields approach, the interatomic potentials are formulated as a simple analytical mathematical equation describing the interaction between atoms. This latter approach is computationally cheaper and can more easily be used to describe the defect in molecular dynamics (MD) simulations, although it has a lower accuracy than DFT.

While DFT can be used to calculate one specific defect in detail, using it for large systems or for many different defects requires a lot of time and computational power. The related energy costs can be directly linked to an increase in CO_2 emissions, and thus a negative impact on the environment. This is why a more efficient method is desirable. The recent popularity of AI and machine learning (ML), and the Nobel Prizes of Physics and Chemistry being awarded to ML,[17] highlights the potential of ML to be used in material science. A machine learning model is trained on an existing dataset, in order to be able to predict a certain output based on the given input. Instead of using extensive DFT calculations for every defect, smaller DFT calculations can be used to create an initial dataset. These can then be used to train an ML model. This ML

model could look at existing DFT calculations, and form patterns between the input (in this case the diamond structure with a specific defect), and the output (the interatomic potentials).[18] When new diamond structures with specific defects are presented to this created ML model, the model, using patterns derived from similar inputs, predicts the interatomic potentials. This uses only a fraction of the computational cost that would be needed to perform the DFT calculations for the new system, while still being able to reach DFT-level accuracies. The potential decrease in computational power shows promising potential for ML to be used as this more efficient method.

Current research already implements ML for predicting and modelling material properties.[19–21] However lack of accuracy and high computational costs,[22] combined with a lack of a generalized model for multiple defects, leads to an opportunity to fill in some knowledge gaps with this work.

As current ML interatomic potential (MLIP) models focus more on small systems and molecules, the next step is to adapt these models for solids, such as a diamond system. The goal in this thesis is to develop the following proof of concept: the possibility to create an MLIP model for a diamond system with defects. Before a general MLIP model can be created for diamond containing various defects, it has to be created for only a few defects: the HV center, and the two H interstitial defect. In order to evaluate the types of MLIP and prove that such a model is possible, the focus in this thesis is only on these two H-related defects in diamond.

2. Theoretical Background

2.1. The hydrogen-vacancy (HV) diamond systems

The systems we will analyze are two diamond cells of $2 \times 2 \times 2$ atoms, with one system containing a HV defect (referred to as HV-diamond), and the other system containing two H interstitial defect (referred to as H2I-diamond). These systems are visualized in **Figure 2**.

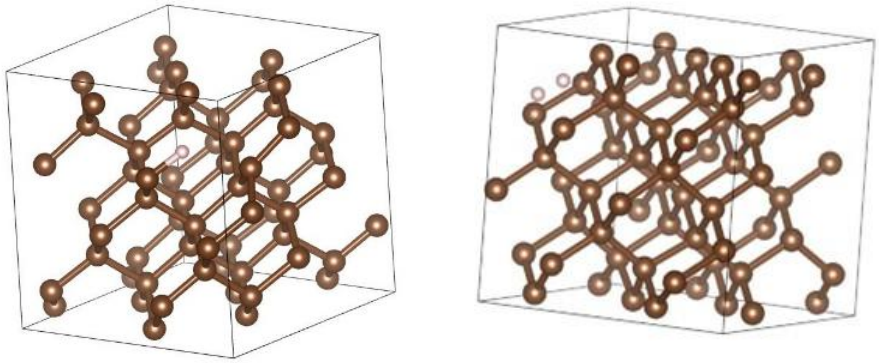

Figure 2. $2 \times 2 \times 2$ diamond systems, (left) HV-diamond (63 atoms), and (right) H2I-diamond (66 atoms). Visualized with VESTA.[23] The brown atoms represent the carbon atoms, while the white atoms represent the hydrogen atoms.

The reason for the limited size of the diamond system, is that larger systems are more computationally expensive, while this thesis is a proof of concept. And secondly, that due to the hardware limitations, models for more than 70 atoms cannot be trained. An actual diamond system used for quantum applications, on the other hand, is much larger. The goal is thus to add periodic boundary conditions (PBCs) to the 'unit cell' from Figure 2, in so far as the MLIPs can deal with PBCs.

The initial systems are defined in POSCARs, a Vienna *Ab initio* Simulation Package (VASP) input file containing the type of system, the lattice geometry, and the starting positions of the atoms.[24,25] The DFT-output is similarly in VASP files, with the most relevant file for the data, being the VASPRUN.xml file, which contains all the output together. Separate parts of the output, for fast evaluations, in our case, can be found in the OSZICAR, OUTCAR, and XDATCAR files.[26]

A more detailed analysis of the DFT outcomes can be found in the supporting information S2. For the following sections, the following details of the four DFT-systems are sufficient: total energies in the range [-580;-545] eV, energies per atom in the range [-9.0;-8.3] eV atom^{-1}, and normalized forces in the range [3;51] eV Å^{-1}. The complete dataset (after filtering), consisted of 47192 data samples, which were divided into training, validation, and testing datasets, with approximately an 80/10/10 (%) split.

2.2. Machine Learning Interatomic Potential overview

For an ML model to go from the input features to the targets, MLIPs need to be constructed. MLIPs are a way to describe the energy and forces of a system as a function dependent on the atomic positions. In other words, an MLIP could be described as a mathematical representation of the Potential Energy Surface (PES). The main difference between ML potentials and DFT calculated potentials, is that ML potentials do not have a description of the electronic structure, (as they do not perform the quantum mechanical electronic structure calculations), and thus works by finding patterns derived from the training data.

Before building an MLIP, certain technical considerations have to be made: The available computational resources, the type of data the model is going to be trained on, the properties it is going to predict, and what physical constraints will be taken into account.

ML models generally work through pattern-finding, instead of a formula-based approach. If a model needs to follow the laws of thermodynamics, or energy conservation, or any of the other physical laws of the universe that describe the systems, these physical constraints need to be added. This is done by the user/developer to ultimately create MLIPs that can accurately predict properties based on scientific principles. This general shape of the to-be-created MLIP, is important to think about before building the actual model.[19,20,27–30]

In general, the steps to create an MLIP are: (1) Creating a reference database containing structural information (input), and compute energies and forces using DFT (ideal output), (2) Create descriptors for the atomic structures from the database, (3) "Learning" the PES using an ML model. These steps are visualized in **Figure 3**.

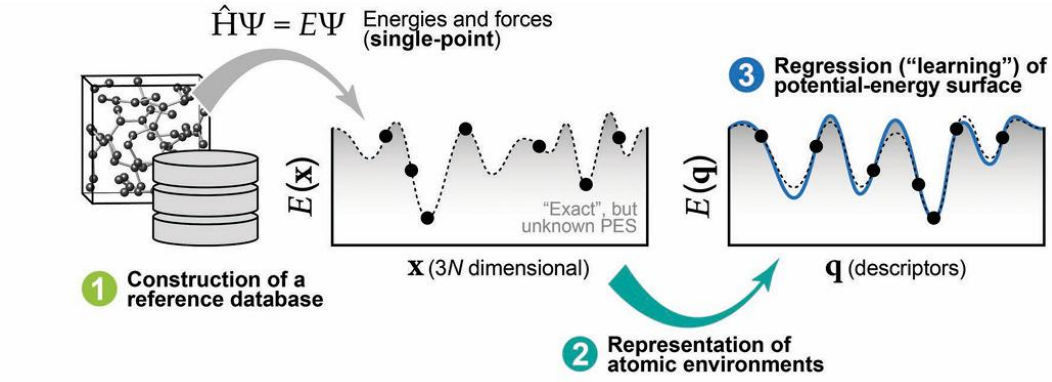

Figure 3. Schematic representation of the steps needed to create an MLIP model. Figure from V. L. Deringer, *et al.*[19]

The reference database (1) needs to contain many geometries. The easiest way to create those consistently is by running an MD (Molecular Dynamics) or MC (Monte Carlo) simulation using the DFT code (more details in section 3.1). The descriptors (2) are a way to translate the atomic positions of the system into something the model can understand. These descriptors are often linked to (and included by) the type of ML model.[28] One such example of a descriptor is SOAP, which stands for *Smooth Overlap of Atomic Positions*.[31] The learning step (3) can be divided in different types of ML models: linear regression, kernel-based methods, and artificial neural networks (NNs).

Linear regression is the simplest of these, makes use of polynomials to fit the interatomic potential, and is in general the fastest type of model, while needing the most guidance during model creation. Examples of this type include SNAP (*Spectral Neighbor Analysis Potential*)[32] and MTP (*Moment Tensor Potentials*).[33] Kernel-based models (such as KRR (*Kernel ridge regression*)[34] and GPR (*Gaussian Process Regression*)[35] types) work by finding similarities between datapoints. Less training data is needed, as long as the atomic structures trained on are similar enough to the to-be-predicted structures. Examples of this type include GAP (*Gaussian Approximation Potential*),[36] and (s)GDML.[37] NNs are the most complex type of model, working similar to neurons. These models have the highest accuracy overall, and are more flexible to differences in input. The downsize is that there are extremely many parameters that need to be adjusted during training, resulting in a higher computational cost than the simpler models. Due to its accuracy, NNs are more studied for MLIPs applications. Examples of this type include SchNet,[38] NequIP,[39] and ANI.[40] Kernel methods are theoretically a nice way to predict properties, as the reference data is incorporated inside the model itself, and is easier to visualize and understand. While NNs are more of a "black-box" method.[41]

The advantage for kernel-based methods is that they need less training data, which leads to less computationally expensive calculations being needed. But where kernel methods can fail due to scaling issues of the sheer size of diamond clusters, NN's prevail, by taking a more generalized approach to linking input properties to the output properties. Another difference is that kernel-based methods use nonparametric ML algorithms, while NN's use parametric ML algorithms. This means that the descriptor for kernel-based methods varies depending on the input data, while the descriptor for NN's stays the same regardless of data, and is thus more scalable.[30] More details about two specific MLIPs, and a general summary of other MLIPs are given in the following subsections.

2.2.1. sGDML

sGDML is a kernel-based model,[37] and stands for *symmetric gradient domain machine learning*. It is based on the gradient domain ML (GDML) model,[42] but with added physical symmetries (the 's'). These symmetries refer to the dynamical (and static) symmetries of the specific system it is trained on, while GDML only looked at spatial and temporal symmetries.[43] sGDML has its basis in the law of energy conservation, and this is, similar to GDML, implemented by taking the atomic gradients of the energies instead of the atomic energies themselves, and only calculating the energies later based on (integration of) the created force field. The reason for this, is that S. Chmiela, *et al.*[42] argue that adding all the atom energies together is an approximation that does not necessarily comply to energy conservation of the system (due to (1) slight inconsistencies between created models and the original calculations, and (2) the fact that quantum systems also have non-local interactions and entanglement), while taking the forces as the main property for the model does lead to energy conservation. The fact that sGDML learns based on the atomic forces, which are the gradients of the PES, is the reason for the "gradient domain" in the name.

The (very) simplified mechanism behind the actual sGDML creation is to (a) take the positions of the system (geometry) and transform these positions into a descriptor, (b) use a kernel function to go from the descriptor to a force field kernel matrix, and (c) transform this matrix into a PES.

a) The descriptor creation uses invariant properties (meaning they do not change even if the system is for example rotated).
b) The creation of the kernel is the part where the similarity between datapoints happens. The kernel can be explained (simplified) by a linear function (**Equation 1**):

   $$K \cdot A = f \quad (1)$$

   Where K is a kernel, A is the to be predicted solution, and f is the atomic force label. Equation 1 is then solved by using a gradient descent method.
c) The PES is ultimately reconstructed by integration of the force field kernel matrix.[41,42]

The goal is to first recreate an existing MLIP model, using an easier dataset, to get the necessary software working, and test my hardware limitations. For this, the workings of sGDML are tested with paracetamol DFT data. Here I follow in the steps of S. Chmiela, *et al.*[37], who created this dataset and trained their model on an Intel Xeon E5-2640 CPU at 2.40 GHz, which is similar to my own laptop (13th Gen Intel(R) Core(TM) i7-13620H at 2.40 GHz). Once this model works,

the next step is to build an MLIP model for our diamond systems. In theory, sGDML models should be possible for systems containing hundreds of atoms,[44] and sGDML in particular is more scalable compared to other kernel-based methods. But in practice, this is still very much limited by the device specifications on which the model is trained.

2.2.2. SchNet

SchNet stands for *Schrödinger Network*, a NN based model that uses continuous-filter convolutional layers in a neural network, and takes quantum-chemical restraints into account.[38] The mechanism behind SchNet is visualized in **Figure 4** and can be simplified into four parts:

1) **Molecular representation**: The atomic positions and nuclear charges of the initial systems are converted into the features (descriptors). This is done by the neural network layers (with 'learned embeddings') which create feature maps.
2) **Atom-wise layers**: These are specifically applied to the separate atoms in the system, and are responsible for recombining the feature maps.
3) **Interaction**: Update the representation of the system based on the positions of all atoms in the system. It is responsible for the interaction between the atoms and the feature maps. This part contains filter-generating networks.
4) **Filter-generating networks**: Continuous-filter convolutions are used to model the interactions between the atoms in the system and to ultimately link the representation of the system (feature map) to the interatomic forces and energy. Here the physical constraints (and chemical knowledge) can be added, such as rotational invariance for the systems.[38,45]

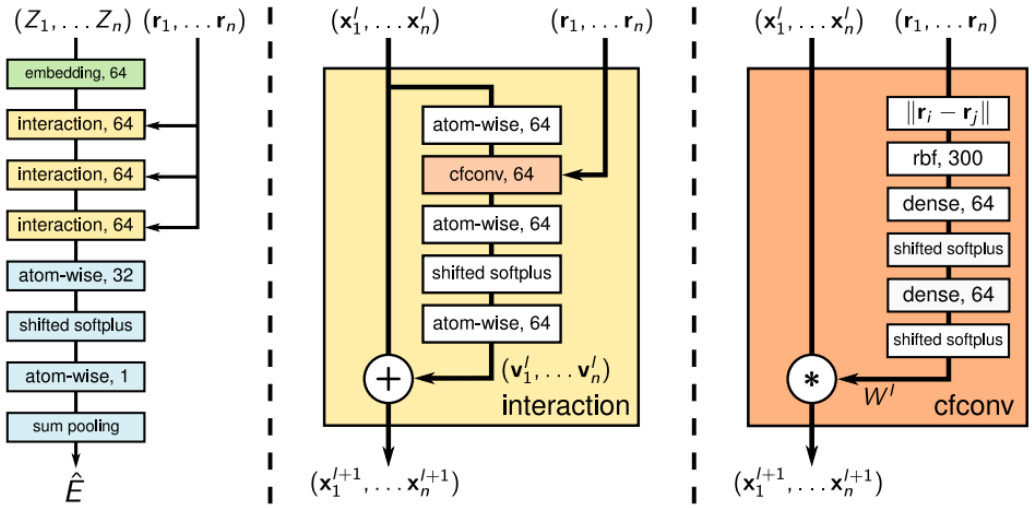

Figure 4. (left) Overview of the SchNet mechanism, (middle) zoomed in 'interaction' block, (right) zoomed in 'continuous-filter convolutions block'. Schematic from K. Schütt, *et al.*[38]

SchNet is focused on predicting the energy of the system, and if the atomic forces need to be predicted, it differentiates the energy (**Equation 2)**.

$$F_i(Z_1, \dots, Z_n, r_1, \dots, r_n) = -\frac{\partial E}{\partial r_i}(Z_1, \dots, Z_n, r_1, \dots, r_n) \quad (2)$$

With F_i being the atomic forces, Z_j and r_j being respectively the nuclear charges and the positions of atom j, and E being the energy. This equation also results in the energy conservation of the systems modeled with SchNet. A practical difference between sGDML and SchNet is that the former uses the total energy of the system, while the latter uses the energy per atom.

2.2.3. Other MLIPs

There are many other MLIPs, and research in this field is rapidly expanding, with many groups making their own classifications of MLIPs. A few note-worthy MLIPs are listed up below.[46]

a) **KLIFF**, stands for *Knowledgebase of Interatomic Models-based Learning-Integrated Fitting Framework*,[47] and is a more general framework for fitting MLIPs. KLIFF combines all the necessary steps to create a MLIP model, and integrates it with KIM (*Knowledgebase of Interatomic Models*).[48] KIM is a project which contains a repository of IPs that can be used to train models. This collaboration with KIM enables exact reproductions, easy testing, easy deployment, and ensures the quality control of KLIFF MLIP models.

b) **GAP**, stands for *Gaussian Approximation Potential*,[36] and works similar to sGDML by using kernel regression. GAP uses SOAP (*Smooth Overlap of Atomic Positions*), which is a local descriptor, while sGDML has a global descriptor. With its use of SOAP, GAP does focus less on the reference data, which means that it works even if the input data is more disconnected. But in contrast to sGDML, GAP is not good in learning long-range interactions.[49]

c) **NequIP**, stands for *Neural Equivariant Interatomic Potential*,[39] which works very similar to SchNet, with the same convolution layers approach and embedding, but has an additional equivariance aspect (extra internal features that are equivariant to rotation and reflection), which changes its learning behavior, and makes it different from SchNet.[39]

d) **GNoME**, stands for *Graph Networks for Materials Exploration*,[50] which looks similar to SchNet with its use of Graph NNs, as it is an expansion of NequIP. GNoME starts with NequIP models and improves them based on their own (larger) dataset. GNoME is an extremely scaled-up version of previous MLIPs, where the focus lies on the prediction of material stability.[50]

e) **GemNet**, stands for *Geometric message passing neural network*,[51,52] which also uses Graph NNs, similar to SchNet. A difference between SchNet and GemNet is that where the former has one type of interaction (Figure 4), the latter has three types of interaction, from which it derives the *geometric message passing* in its name.[51]

2.3. Molecular Dynamics

Richard Feynman claimed the most important statement of scientific knowledge with the highest information-to-words ratio to be: *"All things are made of atoms—little particles that move around in perpetual motion, attracting each other when they are a little distance apart, but repelling upon being squeezed into one another."*[53] This 'atomic hypothesis' brings us to the perpetual motion of atoms. Even in diamond bulk systems, the atoms are always moving. Molecular dynamics (MD), is a way to describe this motion by calculating the trajectories (solving equations of motion) of all parts of the system, per timestep.

There are multiple ways to perform MD, but they are all based on the same principle. From (1) the initial system configuration, to (2) calculating the forces acting on each atom (remember equation 2), to (3) finding the new positions based on the integration of Newton's equations of motion, to (4) repeating step 2 and 3 for each timestep.[54] Step (3) can be performed with multiple algorithms, one example being the Verlet algorithm.[55]

The way that the MDs work in simulations, is by moving each atom based on their forces. The interaction between the system and its environment is specified according to its ensemble. MD simulations can be run in different ensembles (different conditions), that are based on thermodynamics.

a) **A microcanonical ensemble, NVE**, is completely isolated from its environment, with the number of particles (N), the volume (V), and the energy (E) constant. This ensemble is not often used for MDs, because real systems are not completely isolated.
b) In **a canonical ensemble, NVT**, there is a transfer of energy between the system and its environment, but the particles of the system itself stay. The number of particles (N), the volume (V), and the temperature (T) are constant. For NVT, it can be visualized as a system in a heat bath or 'thermostat'.
c) In **the isothermal-isobaric ensemble, NPT**, there is a transfer of heat between the system and its environment, and the volume of the system itself is also variable. The number of particles (N), the pressure (P), and the temperature (T) are constant. This MD is more

computationally costly than NVT, partially due to stress tensors that also need to be calculated.

Lastly, MD simulations can also be used to develop thermodynamic equations, such as the Equation of State (EOS), which shows the relationship between the pressure, the temperature, and the volume of a system.

MDs for all these ensembles can be performed with different methods. The first one being *ab initio* molecular dynamics (AIMD), makes use of DFT. This method has a high accuracy, but is limited to a few atoms over small timescales, and has a high computational cost. This method, with the NPT ensemble, is used to create the reference dataset for this thesis. The second MD method is classical molecular dynamics (CMD), based on empirical force field models, using physics-based potentials (such as the Lennard-Jones potential). When talking about MD, CMD is usually the type that this refers to. This method links atomic interactions to properties on a larger scale, with millions of atoms and longer timescales, but is also less accurate than AIMD. The third MD method is the one based on MLIP. MLIP based MDs try to find a balance between AIMD and CMD, by combining the larger scale and speed of classical MD with the AIMD accuracy. Depending on the specific MLIP, the principle behind their MDs tend to shift closer to either AIMD or CMD.[46,56]

From a more practical point of view, the steps to perform an MD can be visualized, in the case of SchNet-based MLIPs, using schematics from SchNetPack 2.0,[57] as seen in **Figure 5**.

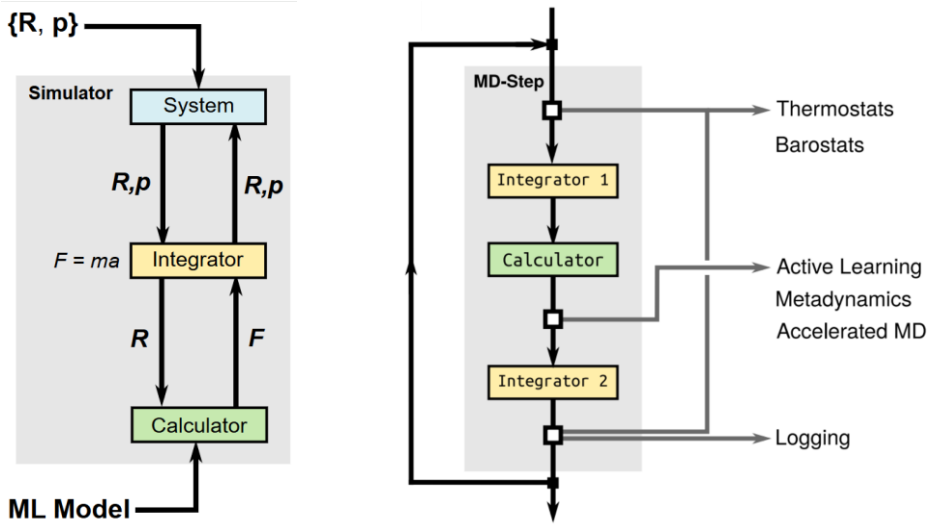

Figure 5. (left) The overview of the workflow of an MD simulator, (right) the overview of a single MD-step. Schematic from SchnetPack 2.0.[57]

The simulator (left) in Figure 5, is the part that performs the MD simulation. The 'calculator' connects to the trained SchNet-based MLIP, as the object that takes the initial structure (positions (*R*) and type of system), and calculates the energies and forces (*F*). The 'integrator' is the part that calculates the new positions (*R*) and momenta (*P*). The 'system' is the object that keeps track of the *R* and *P* of the system (molecule). The MD-step (right) in Figure 5, is a closer view of what happens in one single MD step. Each white square is called a 'simulation hook', because those are the places where customized actions can be 'hooked' to the MD calculation. The thermostats and barostats are to define the temperature and pressure in the MD run (for NVT and NPT, respectively), and the logging is to keep track of the outcomes. For the created MLIP models, the EOS and the NVT MDs will be performed as they are the simplest MDs to calculate for this proof-of-concept MLIP.

3. Results & Discussion

3.1. Database creation

Before an ML model can be made, the data needs to be preprocessed. The input data needs to be of a high enough quality, and large enough to be able to train accurately, validate the training, and test all resulting models. To ensure the quality of the H-defect-diamond calculations, a level of theory with consistent and accurate calculations needs to be used. DFT is a computational quantum mechanical modelling method that uses functionals of the electron density, and is accurate enough to use as a reference for these calculations.

Four NPT calculations (HV-diamond and H2I-diamond, at 500 K and 800 K), were performed using the VASP package on the VSC, by Prof. dr. dr. Vanpoucke. The resulting files, with the calculation details, were delivered as the pre-datasets. From these pre-datasets, the most important value-types to be extracted are the system itself (*i.e.* types of atoms in the system), and per timestep properties (*i.e.* the positions of all the atoms in the system, the total energy, and the forces per atom). Other values of importance are the temperature at which the NPT run is performed, the type of NPT run, the volumes of the system per timestep, *etc*.

For each type of MLIP, different formats of databases are used, due to the differences in the MLIP model infrastructures. This is mostly due to the differences in the creation of the descriptors between models. sGDML uses its own proprietary format, with the database file ending in '.npz'. They do offer conversion scripts, but as those are liable to make errors, it is best to transfer the data directly to their format. A common type of database for MLIPs is the

Atomic Simulation Environment (ASE) database,[58,59] because it is a convenient and compact way to store atoms. SchNet, and the ePotentia model use an ASE database. Additional information for the database creation, such as DFT calculations and database conversion details, are explained in the Experimental/Methods Section of this thesis.

3.2. Model creation

Each model that is created is evaluated using three error-metrics. The Root Mean Square Error (RMSE) (**Equation 3**), the Mean Absolute Error (MAE) (**Equation 4**), and the coefficient of determination (R^2) (**Equation 5**).

$$RMSE = \sqrt{\sum_i^n \frac{\left(y_{pred,i} - y_{true,i}\right)^2}{n}} \tag{3}$$

$$MAE = \frac{\sum_i^n |y_{pred,i} - y_{true,i}|}{n} \tag{4}$$

$$R^2 = 1 - \frac{\sum_i \left(\left(y_{true,i} - y_{pred,i}\right)^2\right)}{\sum_i \left(y_{true,i} - (y_{true,mean})^2\right)} \tag{5}$$

RMSE calculates the standard deviation of the residuals, MAE calculates the average of the residuals, and R^2 calculates the proportion of the variance. This means that RMSE and MAE are dependent on the value ranges of the predicted energy and forces, and have the same units as these energies and forces. The R^2 is independent of those value ranges (without units), and can be used to directly compare all models to each other.

3.2.1. sGDML model

MLIP frameworks do not work immediately after downloading, but instead require a lot of tweaking and setting up a virtual environment near identical to the one the creators of the framework used. This means that it took a while to figure out all the documented and undocumented variables that had an impact on the framework. After extensive trial and error, the sGDML framework was able to be used to recreate an existing paracetamol MLIP model. The next step was to build an MLIP model for the HV-diamond systems.

The training of an sGDML model for larger systems turned out to be limited by the available working memory (RAM) of the used hardware (my laptop only has 16 GB RAM, with even less actually available for the model training). Testing diamond systems of increasing sizes using the sGDML framework on this device, showed a limit of 70 atoms per system. The amount of samples that can be used for training at once was also limited, with the python kernels

crashing when the input was more than 100 samples. To work around this, the model was trained using "data packets". Instead of loading the entire dataset into the working memory of the model, only 100 samples (1% of this particular training dataset) were loaded at once, trained for a small part, and saved to a temporary model file. Then, when there was enough memory free again, the next data packet was opened, and transfer learned on the previous model to create a new "temporary model", until the whole training dataset was processed. This iterative process lowers the quality of the model, as all datapoints cannot be processed at once, but seems to be the only way to train within the current limitations. A graphical representation of this process is shown in **Figure 6.**

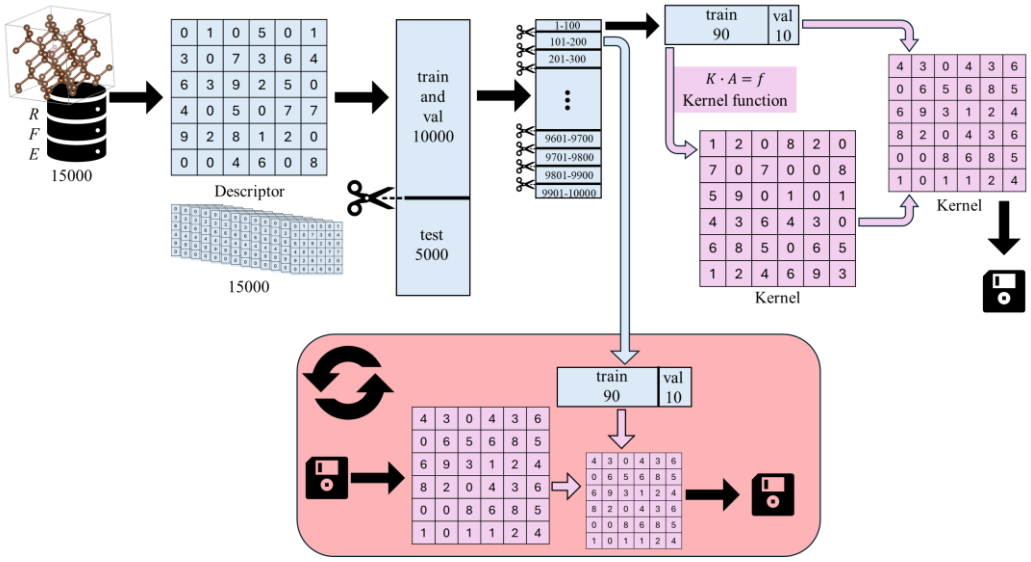

Figure 6. A graphical representation of the "data packets" process. In blue is the part where the data samples are converted to the descriptors and cut into the data packets. In purple is the creation of the kernel. And red (the looping part) shows how each kernel, after being saved, is 'updated' with a new data packet.

To test the quality of this type of model, and lower the overall computational costs, this model was at first only trained and tested on one of the four DFT datasets, namely the HV-diamond at 500 K (which contains a total of 15000 samples). Once the logistics of this training process were streamlined, a small part of the dataset was used as training to test for various hyperparameters. The sGDML model with hyperparameters sigma $= 10$, lambda $= 10^{-10}$, showed the best performance from this small hyperparameter-test. This model was then trained on 10 000 datapoints (of the 15 000) of the complete HV-diamond at 500 K dataset. Instead of only predicting on the test set, this model predicted for the entire dataset, to see if it would also 'remember' the correct values for the training data. The results for both the predicted energy and the predicted normalized forces, are shown in **Figure 7**.

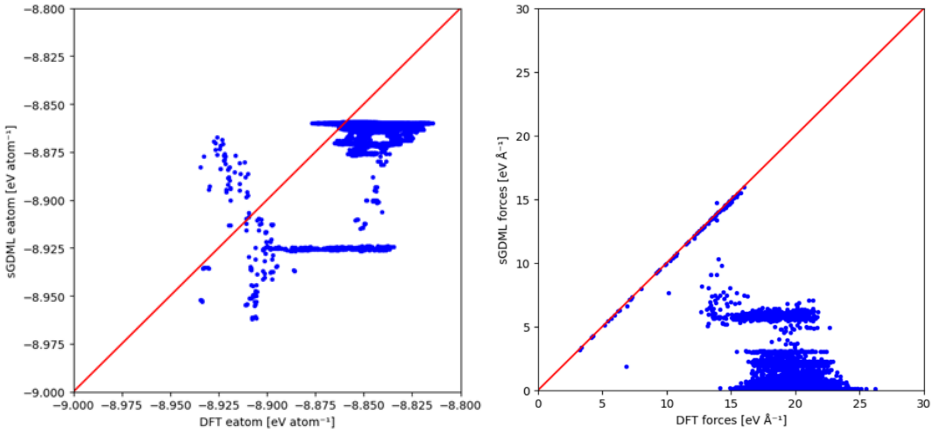

Figure 7. Correlation plots of the predicted energy per atom (left) and normalized forces (right) using the sGDML model with $\mathbf{sigma = 10}$, $\mathbf{lambda = 10^{-10}}$. The red line is what an ideal model would predict.

As can be seen in Figure 7, the sGDML model does not seem to work for the diamond systems within the hardware limitations. The error-metrics for this model are shown in **Table 1**, with negative R^2 values showing a predictive quality even below taking the average value.

Table 1. The RMSE, MAE, and R^2 values of the final sGDML model, for the predicted energies and normalized forces.

Target	RMSE	MAE	R^2
eatom [eV atom^{-1}]	0.021	0.016	-1.946
(normalized) forces [eV Å^{-1}]	18.917	18.741	-143.027

Ultimately the predictions are bad for the sGDML models, and considering that the model cannot even predict correctly for the datapoints it is trained on, a different approach needed to be used. With a clearer understanding of the workings of one type of MLIP model, a different MLIP model was sought.

3.2.2. SchNet model

Similar to sGDML, the SchNet MLIP framework required a lot of tweaking and required setting up a virtual environment. The necessary software was installed and an adapted ASE database was built according to the SchNet specifications. Although SchNet appeared to be the most promising of the NN models, errors outside of my reach (due to device limitations for the software that go back to the python code itself), and the limited hardware (laptop) itself, prevented the creation of a working SchNet model.

A potential solution was found within SchNetPack 2.0, but due to time constraints, this is an exploration for a future project. The basics of SchNet itself did give insight into the workings of an NN MLIP, and NNs (in theory) showed to be the best type of MLIPs for the diamond systems. So an alternative SchNet-based MLIP is sought.

3.2.3. ePotentia model

Due to the confidentiality of most similar existing MLIP models, the limitations of the free-access models, and the time-constraints within a MSc thesis, training an accurate MLIP NN model for a solid system from scratch proved to be a too challenging goal with the available time and resources.

Reaching out to the AI consulting company, ePotentia,[60] where I did my internship during the first year of Materiomics, I was allowed access to their confidential MLIP model infrastructure. They started from the gemnet_oc model (open source),[61] and adapted it within the context of industrial projects, using their own calculator. Similarly, I used their infrastructure and calculator, and started from the gemnet_oc model (so as not to accidentally include confidential data). As this model was already trained on systems containing carbon (C) and hydrogen (H), the MLIP infrastructure existed for further training for a new model (although not trained on systems similar to bulk diamond). Additionally, ePotentia's hardware is dedicated for AI model training, and did not have the same limitations my laptop has.

The input data for this ePotentia model needed to be in ASE format, similar to SchNet, but with different descriptors for the variables. For example 'eatom', for the energy per atom, with the positions, energies, and forces defined slightly different in the database. So an important step here was to convert the DFT diamond data into the correct format (database creation in part 3.1). In order to use the MLIP model to get the predicted values, ePotentia's own calculator was used. The training process consisted of a preprocessing step, and the transfer learning step. The learning happened in epochs, and the specific parameters had to be defined before preprocessing (see Methods section for more details**).**

Initial predictions for the test data, showed that the predicted energy ranges for a diamond system with this initial ePotentia model were (of course) very different than the actual energy values for a diamond system should be (**Figure 8**). To specialize this model for diamond defect systems, transfer learning on the training data was done. Transfer learning is a method in which

an existing ML model is taken, and used as a basis for training a new model. Instead of having to derive all the features and patterns from scratch, the new model has some pre-existing layers to follow in further training. This reduces the computational training cost, and improves the previous model.[62] The initial created models were to gain familiarity with ePotentia's system. Multiple models were made to finetune the necessary parameters, and gain insight into how the model works.

The predictions of the initial model from ePotentia for the test data (consisting of H-defect-diamond bulk systems), are shown in Figure 8. Note that the predicted energy values (y-axis) are in the 10^5 eV/atom range and positive, while the actual DFT energy values (x-axis) have smaller and negative values (Figure 8, left). This is also true for the normalized forces (Figure 8, right).

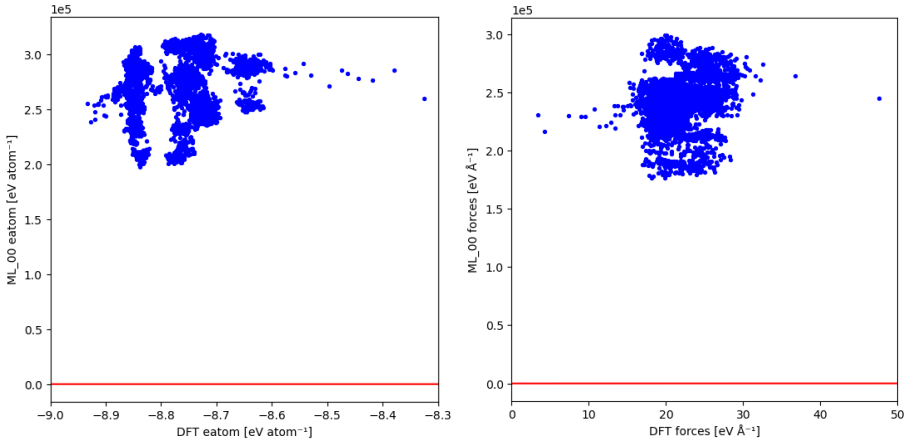

Figure 8. Correlation plots of the predicted energy per atom (left) and normalized forces (right) using the initial 00 model that is not trained on diamond data. Note that for both graphs the x-axis is five orders of magnitude larger than the y-axis. The red line is what an ideal model would predict, but due to the disparity between the axes, it disappears from the range, and appears to fall together with the x-axis).

The next step was then to transfer learn this model, training on DFT data for H-defect-diamond bulk systems, to create new models. Multiple ePotentia-based models were created, with varying parameters, target-importances, and computing times. An overview of the quality of some of these created models can be found in **Table 2**. Model 00 refers to the initial ePotentia model, and thus has wildly varying values from the models that were trained on diamond data.

Table 2. The RMSE, MAE, and R^2 values of the ePotentia-based models, for the predicted energies and normalized forces. Model 00, as the native model, is the only model not trained on the diamond data. The details of the parameters for the other models can be found in the Methods section.

Target	eatom [eV atom^{-1}]			(normalized) forces [eV Å^{-1}]		
Model	RMSE	MAE	R^2	RMSE	MAE	R^2
00	2.667e5	2.654e5	-1.755e13	2.422e5	2.413e5	-6.384e9
02	0.098	0.079	-1.363	3.576	3.015	-0.391
04	0.097	0.078	-1.314	3.391	2.396	-0.251
05	0.041	0.029	0.593	0.897	0.674	0.912
06	0.090	0.073	-1.010	1.082	0.791	0.873
07	0.100	0.080	-1.445	1.185	0.914	0.847
08	0.088	0.072	-0.898	1.517	1.206	0.750
10	0.031	0.023	0.756	0.812	0.604	0.928
12	0.014	0.009	0.953	0.676	0.509	0.950

The performance of the model training for ePotentia was tracked using logs. These can be plotted to see how the model improves itself over each epoch. The RMSE for the predicted values from model 12 are plotted in **Figure 9.**

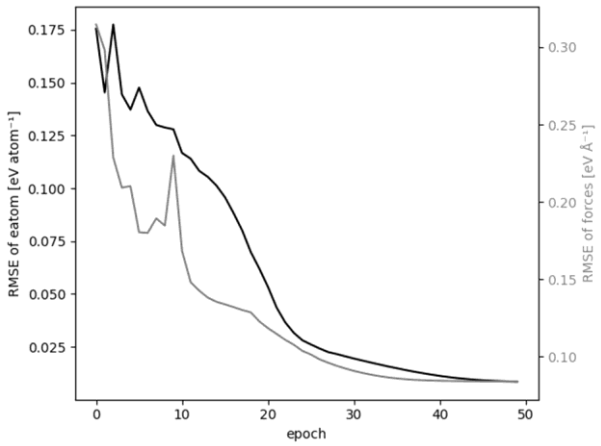

Figure 9. The RMSE of the energy per atom (left axis, black) and the normalized forces (right axis, gray) over the training epochs of the ePotentia-based model 12.

The ePotentia model itself does not look at these literal RMSE values (as those are two values), but instead has one loss function it tries to minimize. This loss function contains the RMSE's values of the two to-be-predicted targets, but attributes weights to them, which is a parameter

that can be changed for every model. For example in models 02, the energy was put as ten times more important than the forces, but the forces were still predicted far better than the energy. This is why for example in model 12, the energy was assigned a higher weight, resulting in a better energy prediction. The predictions of the best and final model 12, for the test data (consisting of H-defect-diamond bulk systems), are shown in **Figure 10.**

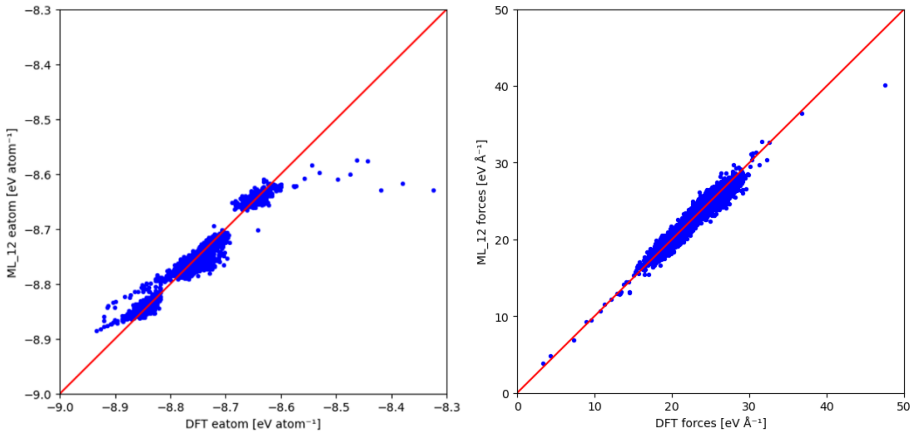

Figure 10. Correlation plots of the predicted energy per atom (left) and normalized forces (right) using model 12, which is the final model trained on diamond data. The red line is what an ideal model would predict.

Model 12 predicts quite accurately for the forces, with an R^2 value of 0.950, and an MAE of 0.51 eV $Å^{-1}$. Similarly for the energies, the predictions have an R^2 value of 0.953, and an MAE of 8.8 meV $atom^{-1}$. For systems with higher energies (DFT run for H2I-diamond, with 800 K), the predictions vary more in quality, as seen by the outlier points that scatter to the right in the left graph of Figure 10. Comparing the energy predictions to GNoME[50], whose model has an initial MAE of 21 meV $atom^{-1}$, model 12 has the more accurate values. Although A. Merchant, *et al.*[50] claim that their final GNoME model predicts energies to an MAE of 11 meV $atom^{-1}$, the MAE of the energy graphed in their paper, lists MAE's of 25 to 50 meV $atom^{-1}$. Still, taking GNoME's best MAE value, 11 meV $atom^{-1}$, and comparing it to model 12, shows that model 12 still surpasses the quality. Model 12 is trained on a less advanced computer than GNoME, with less time (training took less than 18h), fewer data points, and on a different system (bulk diamond). So this type of MLIP still has a lot of opportunities for improvement.

3.3. MD runs

With finally an accurate working model, MD trajectories were calculated, analyzed, and compared to DFT data. Some models refused to work with MD calculations, so a bit of trial

and error was done to find what is and what is not responsible for a successful MD run with these models. One of the problems was that the created pre-processing files for each model should be removed before training a new model. For model 12, some MD runs still did not extrapolate well. This is why the EOS MDs were run with model 10. Due to the computational cost in the prediction of stresses for NPT runs, only EOS and NVT MDs were calculated using the final MLIP models.

3.3.1. EOS

For the EOS MDs, there is a comparison of the energies for varying volumes. The initial systems (and volumes) are the ones defined in the POSCARs for the DFT runs. The volumes were taken between -10% and +10% of the original cell volume. The ASE EOS function was used to plot the EOS graph (**Figure 11**) from the calculated volumes and energies, and to calculate the optimal volume, the minimum energy, and the bulk modulus (**Table 3**).[59]

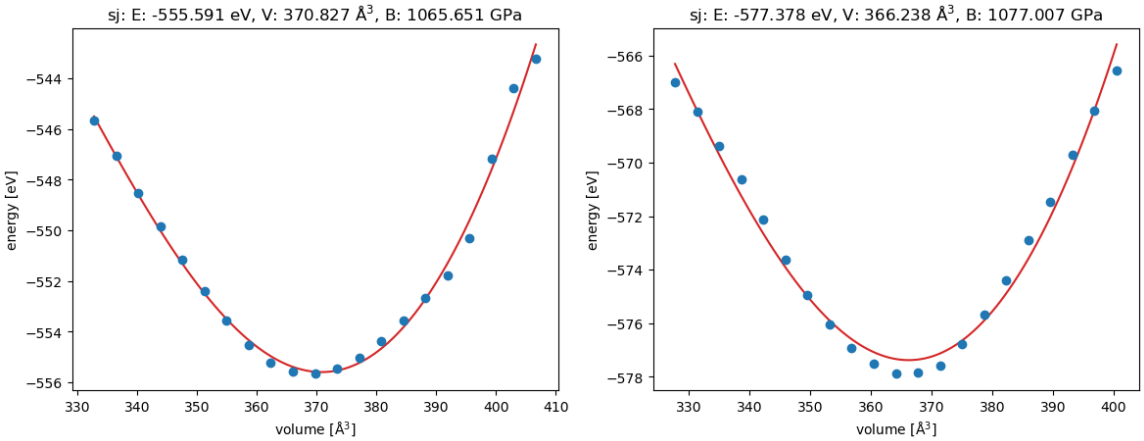

Figure 11. EOS plots for the HV-diamond (left), and the H2I-diamond (right). The ML model used to calculate these graphs is model 10.

In Figure 11, the EOS calculations for both diamond systems can be seen. These graphs show a U-curve, from which the calculated values are shown in Table 3.

Table 3. The optimal volume (*v0*), the minimum energy (*e0*), and the bulk modulus (*B*), with *B* in two different units, of model 10, from the EOS MD.

System	*v0* [Å^3]	*e0* [eV]	*B* [eV Å^{-3}]	*B* [GPa]
Diamond with HV center	370.83	-555.59	6.65	1065.65
Diamond with H2I defect	366.24	-577.38	6.72	1077.01

The bulk modulus (B) is an indication of how much the system resists being compressed. From the EOS, the calculated B for the HV-diamond, and the H2I-diamond is respectively 1066 GPa and 1077 GPa. Referring to literature, the B of diamond (at 4 K) is approximately 443 GPa,[63] and calculated with DFT by M. Hebbache, this value is approximately 463 GPa.[64] These actual values differ from the EOS calculated ones, which may be due to a mistake in the MLIP calculations in regards to EOS, because HV-defects should make the diamond softer, with a lower B. This relatively softer diamond can be seen in that the HV-diamond has a lower B than the H2I-diamond, which has the full crystal structure of diamond without any vacancies.

3.3.2. NVT

For NVT, the Langevin thermostat (from ASE) was used, as this adds a friction and fluctuating force term. The temperature (T) chosen for the NVT runs were 0, 100, 300, 500, 800, and 1000 K. (One T per MD run per system). The calculated total energies for every timestep with NVT are plotted in **Figure 12**.

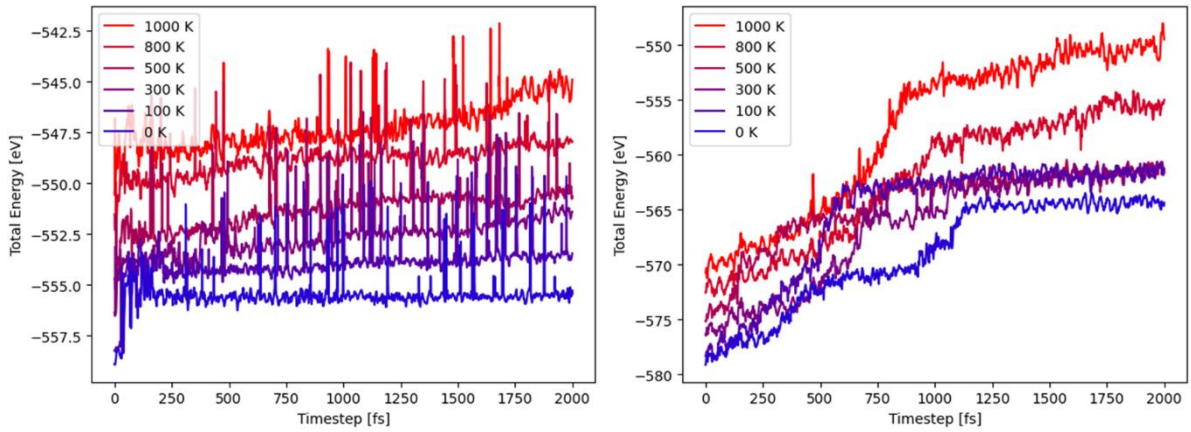

Figure 12. NVT MD plots for the HV-diamond (left) and the H2I-diamond (right). The ML model used to calculate these graphs is model 12. For each system, six NVT MDs are simulated, with temperatures ranging from 0 K (blue) till 1000 K (red).

The NVT MD simulations were run following Langevin dynamics, with a timestep of 1 fs. It can be seen that the higher energies are linked to the higher heat baths (Figure 12). For HV-diamond, the NVT MD stayed relatively constant in total energy, while for the H2I-diamond, the energy increased over time. This can also be seen when visualizing the trajectories in OVITO, where the atoms move faster with increasing temperatures. A comparison of the total, potential, and kinetic energies for one MD run is shown in **Figure 13.**

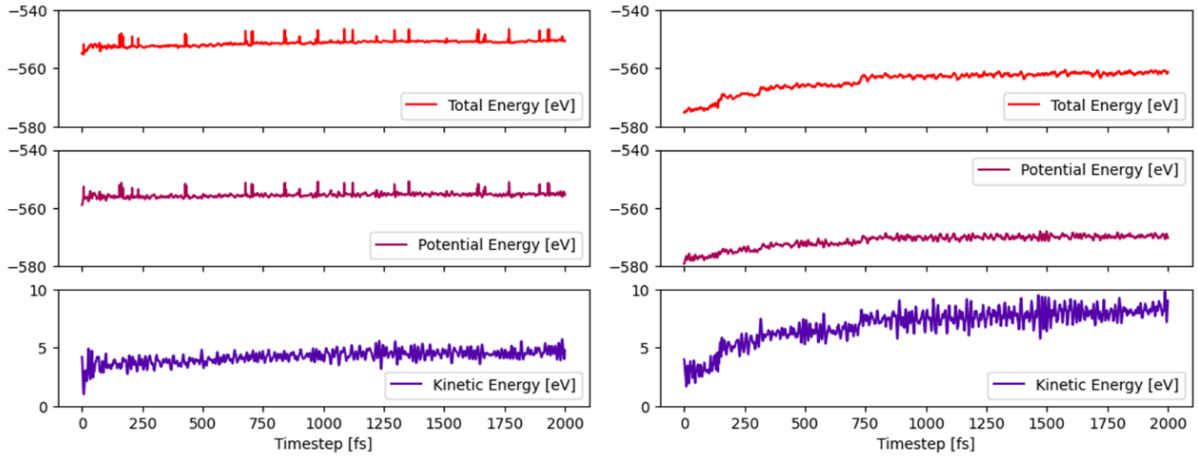

Figure 13. NVT MD plots for the HV-diamond (left) and the H2I-diamond (right). The ML model used to calculate these graphs is model 12. The NVT MDs are simulated for $T = 500$ K.

For both diamond systems, the kinetic energy is smaller than the potential energy, and contributes less to the total energy. This is because diamond is a rigid structure, and has most of its energy stored in its C-C bonds. The C atoms can vibrate, but not move freely. For the H2I-diamond, where the interstitial H defect can move easier, the kinetic energy near the end of the simulation is seen to be higher (almost double) compared to the HV-diamond. More details about the NVT simulations can be found in the Methods section, and in supporting information S5.

3.4. Sustainability

With current climate concerns, sustainability is a recurring topic in every type of research.[65] This thesis itself strives for sustainability by exploring alternatives to computationally extensive DFT calculations, by using MLIP. B. G. Del Rio, *et al.*[66] compared the cost of DFT versus ML in CPU time for organic molecules, polymer chains, and polymer crystals of various sizes, as shown in **Figure 14.** Here the ML based calculations have shown to be a more environmental-friendly option than DFT calculations, especially when scaling systems up to larger sizes.

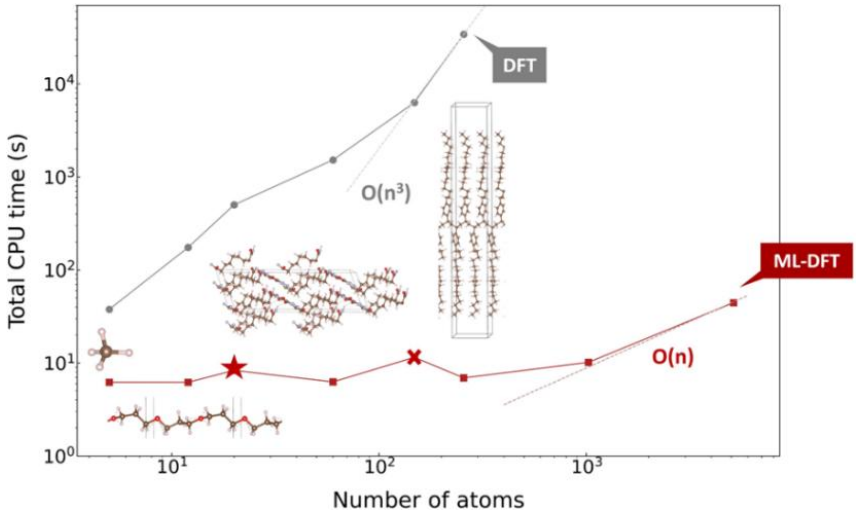

Figure 14. Computational cost (in CPU time), for DFT (gray) and ML-DFT (red), for different system sizes. The systems are organic molecules, polymer chains, and polymer crystals, consisting of predominantly C and H atoms. Graph from B. G. Del Rio, *et al*.[66]

Since ML, and especially training the models, is also computationally costly, methods to try to minimize this resource-use are applied, such as frugal computing.[67] Frugal computing is a way to minimize the computational resources (both hardware and energy) while still keeping the same quality. Frugal modeling is a variation of this, where the focus lies on developing models with minimal computational costs (being resource-conscious), while keeping a similar value. This is finding a balance between the minimum quality needed, and the maximum quality that is possible to achieve. The way that frugal computing is applied in this thesis, is by (1) limiting the amount of times that the DFT values are read (avoiding unnecessary loops), and saving the necessary data to temporary files. (2) working towards training MLIPs on smaller devices, instead of a supercomputer. This is seen by trying sGDML first, especially because this model was supposed to be possible on a resource-limited device. And (3) using transfer learning on existing models, to avoid the initial (larger) training cost.

4. Conclusion

The goal of this thesis was to develop a proof of concept that it is possible to create an MLIP model for a diamond system with defects. Specifically for bulk diamonds with an HV center or an H2I defect. An exploration of the types of MLIP led from kernel-based methods such as sGDML, to NN-based methods such as SchNet, and showed that although MLIPs are possible in theory, and for simple systems, that adapting them for diamond systems requires sufficiently advanced hardware. This is why a final MLIP is created using the infrastructure and an existing model from ePotentia.

The final MLIP (model 12) showed an accuracy of R^2=0.950, and MAE=0.509 eV $Å^{-1}$ for the normalized forces, and R^2=0.953, and MAE=8.8 meV $atom^{-1}$ for the energies. The MAE of 8.8 meV $atom^{-1}$ for the energies competes with the MAE of 11 meV $atom^{-1}$ for GNoME. Still, model 12 has a lot of opportunities for improvement in the future. MD runs using the created MLIP models resulted in EOS and NVT simulations. The EOS (using model 10) specifically showed a larger calculated bulk modulus than expected (1066-1077 GPa, instead of 443-463 GPa). In the end, the MD runs were possible with the MLIP models, and with an improved MLIP model could show a promising future for MD simulations for diamond.

For new models and MDs, the possibility arises for the simulated movement of H in diamond to be compared with experimental values (diffusion), because DFT calculations are also still approximations, and an experimental comparison will increase the accuracy-validation.

During this thesis, sustainability was kept in mind by using frugal computing, besides the fact that the goal of this thesis is to build an initial MLIP model to in the future replace computationally expensive DFT calculations.

For future MLIPs, currently more specialized hardware is needed to train the models accurately enough. Through the rise of new MLIPs however, it could be that in the near future, the software will hopefully be streamlined enough to be able to train larger models with less resources. It is proven that MLIPs are possible for diamond systems with H-related defects. With larger DFT reference data sets and more advanced MLIPs, molecular dynamics for these (and more complicated) systems have the potential to reach far.

5. Experimental/Methods Section

Database creation

The DFT calculated data was provided by Prof. dr. dr. Danny E.P. Vanpoucke, who calculated these systems using VASP,[25] on the VSC (Flemish Supercomputer Center). The MD runs were calculated for the NPT ensemble, with the following parameters defined in the INCAR for the HV-diamond for 500K:

- IBRION = 0, for defining the MD
- ISIF = 3, for specifying the NPT ensemble
- MDALGO = 3, for using the Langevin thermostat
- LANGEVIN_GAMMA = 2.0 2.0, for defining the friction coefficients of Langevin atoms

- TEBEG = 500, for defining the starting temperature in Kelvin
- TEEND = 500, for defining the ending temperature in Kelvin
- POTIM = 2, for a timestep of 2fs
- NSW = 15000, for the number of electronic steps

The two diamond systems themselves are defined in the POSCARs, which can be found in the supporting information S1.

The four DFT created datasets were for:

1) HV-diamond NPT for 500K (15000 steps)
2) HV-diamond NPT for 800K (15000 steps)
3) H2I-diamond NPT for 500K (13922 steps)
4) H2I-diamond NPT for 800K (7431 steps)

This last dataset (4), showed abnormal behavior after step 3275, with a sudden jump to high positive energies. Because of this, only the first 3270 samples from this dataset were taken. More details about these datasets can be found in the supporting information S2.

Database Conversion

To get the needed information from the VASP output to the npz and ASE databases, scripts were written. An overview of the steps is that the atom types (*z*), their cell parameters, positions (*R*), forces (*F*), and energy (*E*), were extracted from the vasprun.xml file, and saved to a temporary file using python in Jupyter Notebooks. For each sample, there was one *E*, but multiple *R* and *F*, since those are defined in three dimensions for every atom. For the MLIP models themselves, this does not matter, but for evaluating the output, similar to *E*, the *F* needed to be normalized. The normalization of *F* was done by taking the square of each force vector of each atom in a molecule, summing them up, and taking the root of this whole summation (**Equation 6**). The resulting value is the normalized force for that specific sample (molecule).

$$normalized\ F = \sqrt{\sum_{i}^{\#\ atoms} {F_{i,x}}^2 + {F_{i,y}}^2 + {F_{i,z}}^2} \quad (6)$$

A script was written to extract the needed values from the vasprun.xml file, and the accuracy of this script was doublechecked for each dataset. Once all the *R, F, E, z, cell, etc.* were extracted from the vasprun.xml file, the new databases were created. (Note: the *F* from here on is an array of all atomic forces. The normalized forces are only used after prediction, to evaluate the

results.) All computational processes regarding database conversion were performed on my own laptop.

First is the creation of the sGDML npz database. The conversion of the data was straightforward, once the structure of the npz file was figured out, and happened in the following steps:

- Choose a dataset
- Load the *R, F, E, z*, and *cell* of all molecules of the dataset
- Define the 'theory' of the structure (necessary for sGDML)
- Create the npz structure

$$\text{np.savez}(f'\{\text{name}\}.\text{npz}', \text{name} = \text{name}, \text{z} = \text{z}, \text{E} = \text{E}, \text{F} = \text{F}, \text{R} = \text{R}, \text{theory} = \text{theory})$$

- For creating the training/validation/test databases, the npz file could not be split easily. So the molecules were first given a number, this number list was split into the train/val/test categories, and each train/val/test npz was created anew by only writing the molecules with the number belonging to their list in the corresponding database.

For creating ASE databases, the Atomic Simulation Environment (ASE) database was used.[58,59] A simplified overview of this process is listed below.

- Choose a dataset and load the *R, F, E, z*, and *cell* of all molecules of the dataset
- Create the (empty) ASE database
- Open this database with "$\text{ase.db.core.connect}$"
- For every molecule in the dataset:
    - Define an ASE atoms object:

$$\text{atoms} = \text{Atoms}(\text{type}, \text{pbc} = \text{True}, \text{positions} = \text{R}[\text{molecule}], \text{cell} = \text{cell\_molecule})$$

        - Type is either 'C62H1' or 'C64H2', depending on the system
        - pbc refers to the periodic boundary conditions
    - Calculate the energy per atom

$$\text{eatom} = \text{E}[\text{molecule}] / \text{nr\_of\_atoms}$$

    - Assign a calculator to the atoms object, in this case a SinglePointCalculator

$$\text{atoms.calc} = \text{SinglePointCalculator}(\text{atoms}, \text{forces} = \text{F}[\text{molecule}], \text{energy} = \text{E}[\text{molecule}])$$

    - Write the atoms object, along with additional necessary data, in the database

$$\text{db.write}(\text{atoms}, \text{data} = \{'\text{energy}': \text{E}[\text{molecule}], '\text{forces}': \text{F}[\text{molecule}], '\text{eatom}': \text{eatom}\})$$

- Repeat this with all datasets
- Split the final database into training/validation/test databases

Computational details of the MLIPs

The MLIP models trained on the infrastructure and framework of ePotentia, had varying parameters and approaches. The basis (initial) model was the gemnet_oc model "gnoc_oc22_oc20_all_s2ef.pt". Some models were directly transfer learned on this model, while others were transfer learned on those new models. Additionally, the training ASE database was adapted along the model trainings, as small mistakes kept cropping up. This is why later models, even if trained similarly as previous models, showed improved accuracies. For training, predictions and MD simulations, the fastatom calculator from ePotentia was used. The training process parameters were defined in four different categories: data, architecture, optimizer, and augmentation.

In 'data', the ASE database was defined (all train/val/test databases), along with the relevant property the model was trained on: eatom. The batch size (number of training samples in one training iteration), was put at 8, since higher batch sizes were too large to be trained. Lastly the mean and standard deviation (stdev) of the database were defined. For all trained models, these values were: $\text{mean} = -8.769570882993687$ and $\text{stdev} = 0.06436164061396021$ (for eatom), as the database consisted of the same training data for all models.

In 'architecture', the MLIP structure was defined. For these MLIP models it was based on the gemnetoc architecture, which is similar to the SchNet MLIP. Here a number of model-specific parameters were defined, such as 2048 features, 256 filters, and 16 interactions.

In 'optimizer', the learning rate was chosen to be 0.001, based on a small learning rate optimization test. The force and stress types were chosen, for which the force type is 'direct', and the stress type is 'none', because the stresses were not trained, due to their higher computational cost. Then the weights for training the forces, energies and stresses were chosen. Since stresses were not trained, the stress weight, ks, was zero. The force weight, kf, and the energy weight, kp, were varied for the different models, with models 01, 02, 04, 05, and 08 choosing $kp/kf = 10$, models 06 and 07 choosing $kp/kf = 15$, and models 10, 11, and 12 choosing $kp/kf = 20$. This increase in energy weight was chosen to prioritize the energy, as the accuracy for the forces was better for initial models.

Next in the 'optimizer' was the number of epochs, training samples, and validation samples. An epoch generally refers to how often the model goes through the entire training database. In

the case of this model, instead of the entire database, the epoch referred to the number of training samples defined in the 'optimizer' part. If no number of samples were defined, the entire training database was taken. For model 05, 06, 08, 10 and 12, the epochs referred to are indeed over the whole training database, but for the other models the number of training samples were either 4096 or 8192, with the corresponding number of validation samples always one fourth of this value. The number of epochs themselves, were chosen between 5 and 50, with the models with higher epochs and samples taking significantly longer to train (around 17h) than the other models (around 3h).

In 'augmentation', one relevant parameter was the number of freezing epochs. A freezing epoch refers to the layers from the pre-trained model, so that an epoch will be trained, without the model forgetting the previous patterns from the first model. Another relevant parameter from 'augmentation' was the initial model the transfer learning was applied to. Models 01, 04, 05, 08, 10, and 12 were trained starting from the gemnet_oc model, while the other models were trained starting from models 01 through 10.

Throughout the training of the models, logs were kept to see how the loss functions (and RMSE) of the models evolve over the epochs. A short overview of these logs, along with some previously listed model parameters, can be found in the supporting information S3.

Computational details of the MDs

Simply predicting the forces and energies with as input a given system, is similar to a static DFT calculation. MD simulations are a bit more complicated, needing additional information regarding the circumstances (*e.g.* the ensemble) of the simulation. The MD trajectories, as they make use of the fastatom calculator and the MLIP model, were performed on ePotentia's infrastructure, but the analysis of these trajectories was done on my own laptop.

The EOS (for each system) was calculated, starting from the system POSCAR, by rescaling the volume within -10% and +10% of the original cell volume, with steps per 1%. From these new resulting *R*, the energies were predicted. The ASE EOS function was then used to plot these volumes and energies as the EOS graph (Figure 11). This same ASE function was then used to calculate the *v0,* the *e0*, and the *B*. The ASE units for the bulk modulus are eV $Å^{-3}$. To get the value in GPa, **Equation 7** was used.

$$B\ [GPa] = B\ [\text{eV Å}^{-3}] \cdot 160.21766208\ [GPa\ \text{eV}^{-1}\ Å^{3}] \quad (7)$$

The NVT MD was calculated with the Langevin thermostat, as this adds a friction and fluctuating force term. The chosen timestep was 1 fs, and the chosen friction coefficient was 0.001. Twelve NVT simulations were run in total, six per system, with temperature (*T*) chosen as 0, 100, 300, 500, 800, and 1000 K. (One *T* per MD run per system). For each simulation, the total, potential, and kinetic energy are given (total energies visualized in Figure 12). To check if the NVT run behaves as expected, OVITO was used to visualize the moving system. Further analysis to quantify the movement of the atoms in these NVT simulations, in order to plot this in a graph, was done in supporting information S5.

Acknowledgements

My profound gratitude goes first to my mentor E. Aylin Melan and my promotor Prof. dr. dr. Danny E.P. Vanpoucke, for their support and knowledge during this thesis. I would also like to thank Prof. dr. dr. Danny E.P. Vanpoucke for providing the reference DFT calculated data on which I trained the models. Next, I am extremely grateful for Michael Sluydts and Catharina Jaeken from ePotentia, for access to their MLIP model, calculator, infrastructure, and their insight and knowledge support during this project. The extra miles they went to help me get an accurately working MLIP model, brought the turning point in this thesis.

References

1. Siyushev, P. *et al.* Photoelectrical imaging and coherent spin-state readout of single nitrogen-vacancy centers in diamond. *Science* **363**, 728–731 (2019).
2. Mzyk, A. *et al.* Diamond color centers in diamonds for chemical and biochemical analysis and visualization. *Anal. Chem.* **94**, 225–249 (2022).
3. Miyoshi, K. Structures and Mechanical Properties of Natural and Synthetic Diamonds. (1998).
4. Smith, J. M., Meynell, S. A., Jayich, A. C. B. & Meijer, J. Colour centre generation in diamond for quantum technologies. *Nanophotonics* **8**, 1889–1906 (2019).
5. Doherty, M. W. *et al.* The nitrogen-vacancy colour centre in diamond. *Physics Reports* **528**, 1–45 (2013).
6. Czelej, K., Zemła, M. R., Śpiewak, P. & Kurzydłowski, K. J. Quantum behavior of hydrogen-vacancy complexes in diamond. *Phys. Rev. B* **98**, 235111 (2018).
7. Chang, C., Liao, Y., Wang, G. Z., Ma, Y. R. & Fang, R. C. CVD Diamond Growth. in *Crystal Growth Technology* 93–141 (Elsevier, 2003). doi:10.1016/B978-081551453-4.50006-3.

8. Maclear, R. D. *et al.* The distribution of hydrogen in polycrystalline CVD diamond. *Diamond and Related Materials* **8**, 1615–1619 (1999).
9. Day, M. C. *et al.* Hydrogen-related defects in diamond: A comparison between observed and calculated FTIR spectra. *Diamond and Related Materials* **143**, 110866 (2024).
10. Lyons, J. L. & Van De Walle, C. G. Surprising stability of neutral interstitial hydrogen in diamond and cubic BN. *J. Phys.: Condens. Matter* **28**, 06LT01 (2016).
11. Goss, J. P., Briddon, P. R., Rayson, M. J., Sque, S. J. & Jones, R. Vacancy-impurity complexes and limitations for implantation doping of diamond. *Phys. Rev. B* **72**, 035214 (2005).
12. Bradac, C., Gao, W., Forneris, J., Trusheim, M. E. & Aharonovich, I. Quantum nanophotonics with group IV defects in diamond. *Nat Commun* **10**, 5625 (2019).
13. Smith, E. M. & Wang, W. Fluid CH4 and H2 trapped around metallic inclusions in HPHT synthetic diamond. *Diamond and Related Materials* **68**, 10–12 (2016).
14. Salustro, S. *et al.* Hydrogen atoms in the diamond vacancy defect. A quantum mechanical vibrational analysis. *Carbon* **129**, 349–356 (2018).
15. Sideras-Haddad, E. *et al.* Hydrogen and oxygen chemistry and dynamics in diamond studied by nuclear microscopic techniques. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* **181**, 419–425 (2001).
16. Giustino, F. *Materials Modelling Using Density Functional Theory: Properties and Predictions*. (Oxford university press, Oxford, 2014).
17. The Nobel Prize in Physics 2024. *NobelPrize.org* https://www.nobelprize.org/prizes/physics/2024/press-release/.
18. Mueller, T., Hernandez, A. & Wang, C. Machine Learning for interatomic potential models. *The Journal of Chemical Physics* **152**, 050902 (2020).
19. Deringer, V. L., Caro, M. A. & Csányi, G. Machine Learning interatomic potentials as emerging tools for materials science. *Advanced Materials* **31**, 1902765 (2019).
20. Deringer, V. L. & Csányi, G. Machine Learning based interatomic potential for amorphous carbon. *Phys. Rev. B* **95**, 094203 (2017).
21. Morrow, J. D., Gardner, J. L. A. & Deringer, V. L. How to validate machine-learned interatomic potentials. *The Journal of Chemical Physics* **158**, 121501 (2023).
22. Zuo, Y. *et al.* Performance and cost assessment of Machine Learning interatomic potentials. *J. Phys. Chem. A* **124**, 731–745 (2020).
23. Momma, K. & Izumi, F. VESTA 3 for three-dimensional visualization of crystal, volumetric and morphology data. *J Appl Cryst* **44**, 1272–1276 (2011).

24. POSCAR - VASP Wiki. https://www.vasp.at/wiki/index.php/POSCAR.
25. VASP - Vienna Ab initio Simulation Package. https://www.vasp.at/.
26. Category:Output files - VASP Wiki. https://www.vasp.at/wiki/index.php/Category:Output_files.
27. Mishin, Y. Machine-learning interatomic potentials for materials science. *Acta Materialia* **214**, 116980 (2021).
28. Wang, G. *et al.* Machine learning interatomic potential: Bridge the gap between small-scale models and realistic device-scale simulations. *iScience* **27**, 109673 (2024).
29. Unke, O. T. *et al.* Machine Learning Force Fields. *Chem. Rev.* **121**, 10142–10186 (2021).
30. Pinheiro, M., Ge, F., Ferré, N., Dral, P. O. & Barbatti, M. Choosing the right molecular machine learning potential. *Chem Sci* **12**, 14396–14413.
31. Bartók, A. P., Kondor, R. & Csányi, G. On representing chemical environments. *Phys. Rev. B* **87**, 184115 (2013).
32. Thompson, A. P., Swiler, L. P., Trott, C. R., Foiles, S. M. & Tucker, G. J. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *Journal of Computational Physics* **285**, 316–330 (2015).
33. Shapeev, A. V. Moment Tensor Potentials: a class of systematically improvable interatomic potentials. *Multiscale Model. Simul.* **14**, 1153–1173 (2016).
34. Murphy, K. P. *Machine Learning - A Probabilistic Perspective*. (MIT Press, Cambridge, 2014).
35. Deringer, V. L. *et al.* Gaussian Process Regression for Materials and Molecules. *Chem. Rev.* **121**, 10073–10141 (2021).
36. Bartók, A. P. & Csányi, G. Gaussian approximation potentials: A brief tutorial introduction. *International Journal of Quantum Chemistry* **115**, 1051–1057 (2015).
37. Chmiela, S., Sauceda, H. E., Poltavsky, I., Müller, K.-R. & Tkatchenko, A. sGDML: Constructing accurate and data efficient molecular force fields using machine learning. *Computer Physics Communications* **240**, 38–45 (2019).
38. Schütt, K. *et al.* SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. in *Advances in Neural Information Processing Systems* vol. 30 (Curran Associates, Inc., 2017).
39. Batzner, S. *et al.* E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat Commun* **13**, 2453 (2022).
40. Smith, J. S., Isayev, O. & Roitberg, A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **8**, 3192–3203 (2017).

41. Blücher, S., Müller, K.-R. & Chmiela, S. Reconstructing Kernel-Based Machine Learning Force Fields with Superlinear Convergence. *J. Chem. Theory Comput.* **19**, 4619–4630 (2023).
42. Chmiela, S. *et al.* Machine learning of accurate energy-conserving molecular force fields. *Sci. Adv.* **3**, e1603015 (2017).
43. Chmiela, S., Sauceda, H. E., Müller, K.-R. & Tkatchenko, A. Towards exact molecular dynamics simulations with machine-learned force fields. *Nat Commun* **9**, 3887 (2018).
44. Accurate global machine learning force fields for molecules with hundreds of atoms | Science Advances. https://www.science.org/doi/10.1126/sciadv.adf0873.
45. Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A. & Müller, K.-R. SchNet – A deep learning architecture for molecules and materials. *The Journal of Chemical Physics* **148**, 241722 (2018).
46. Jacobs, R. *et al.* A practical guide to machine learning interatomic potentials – Status and future. *Current Opinion in Solid State and Materials Science* **35**, 101214 (2025).
47. Wen, M., Afshar, Y., Elliott, R. S. & Tadmor, E. B. KLIFF: A framework to develop physics-based and machine learning interatomic potentials. *Computer Physics Communications* **272**, 108218 (2022).
48. Tadmor, E. B., Elliott, R. S., Sethna, J. P., Miller, R. E. & Becker, C. A. The potential of atomistic simulations and the knowledgebase of interatomic models. *JOM* **63**, 17–17 (2011).
49. Vassilev-Galindo, V., Fonseca, G., Poltavsky, I. & Tkatchenko, A. Challenges for Machine Learning Force Fields in Reproducing Potential Energy Surfaces of Flexible Molecules. *The Journal of Chemical Physics* **154**, 094119 (2021).
50. Merchant, A. *et al.* Scaling deep learning for materials discovery. *Nature* **624**, 80–85 (2023).
51. Gasteiger, J., Becker, F. & Günnemann, S. GemNet: Universal Directional Graph Neural Networks for Molecules. Preprint at https://doi.org/10.48550/arXiv.2106.08903 (2024).
52. Stocker, S., Gasteiger, J., Becker, F., Gunnemann, S. & Margraf, J. T. How Robust are Modern Graph Neural Network Potentials in Long and Hot Molecular Dynamics Simulations?
53. The Feynman Lectures on Physics Vol. I Ch. 1: Atoms in Motion. https://www.feynmanlectures.caltech.edu/I_01.html.

54. Molecular Dynamics: Equations of motion. *CompChems* https://www.compchems.com/molecular-dynamics-equations-of-motion/ (2022).
55. Verlet, L. Computer 'Experiments' on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Phys. Rev.* **159**, 98–103 (1967).
56. Tuckerman, M. E. & Martyna, G. J. Understanding Modern Molecular Dynamics: Techniques and Applications. *J. Phys. Chem. B* **104**, 159–178 (2000).
57. Schütt, K. T., Hessmann, S. S. P., Gebauer, N. W. A., Lederer, J. & Gastegger, M. SchNetPack 2.0: A neural network toolbox for atomistic machine learning. *The Journal of Chemical Physics* **158**, 144801 (2023).
58. A database for atoms — ASE documentation. https://wiki.fysik.dtu.dk/ase/ase/db/db.html.
59. Hjorth Larsen, A. *et al.* The atomic simulation environment—a Python library for working with atoms. *J. Phys.: Condens. Matter* **29**, 273002 (2017).
60. Web hosting, Site/App development and Data consulting - ePotentia. https://www.epotentia.com/ (2019).
61. Gasteiger, J. *et al.* GemNet-OC: Developing Graph Neural Networks for Large and Diverse Molecular Simulation Datasets. Preprint at https://doi.org/10.48550/arXiv.2204.02782 (2022).
62. Wang, J. & Chen, Y. *Introduction to Transfer Learning: Algorithms and Practice*. (Springer Nature Singapore, Singapore, 2023). doi:10.1007/978-981-19-7584-4.
63. Kittel, C. *Introduction to Solid State Physics*. (Wiley, Hoboken, N.J., 2005).
64. Hebbache, M. First-principles calculations of the bulk modulus of diamond. *Solid State Communications* **110**, 559–564 (1999).
65. Alšauskas, O. World Energy Outlook 2024.
66. Del Rio, B. G., Phan, B. & Ramprasad, R. A deep learning framework to emulate density functional theory. *npj Comput Mater* **9**, 158 (2023).
67. Gutiérrez-Finol, G. M., Ullah, A., González-Béjar, M. & Gaita-Ariño, A. A call for frugal modelling: two case studies involving molecular spin dynamics. *Green Chem.* **27**, 3167–3177 (2025).

Supporting Information

S1 POSCARs for the diamond systems

```
C_diamond_PBE_lattice_is_cube_root_of_EO
  3.58854144700000
    2.0000000000000000    0.0000000000000000    0.0000000000000000
    0.0000000000000000    2.0000000000000000    0.0000000000000000
    0.0000000000000000    0.0000000000000000    2.0000000000000000
  H   C
    1   62
Direct
  0.3125000000000000  0.3125000000000000  0.3125000000000000
  0.3918346300979607  0.1196207963298477  0.1196207963298477
  0.1196207963298477  0.3918346300979607  0.1196207963298477
  0.1196207963298477  0.1196207963298477  0.3918346300979607
  0.5053792036701523  0.2331653699020393  0.5053792036701523
  0.2331653699020393  0.5053792036701523  0.5053792036701523
  0.5053792036701523  0.5053792036701523  0.2331653699020393
  0.2530482512562529  0.0032868013424974  0.0032868013424974
  0.7526630980128033  0.0007044196000763  0.0007044196000763
  0.0032868013424974  0.2530482512562529  0.0032868013424974
  0.5052784288900583  0.2490691902858657  0.9962515591634116
  0.2490691902858657  0.5052784288900583  0.9962515591634116
  0.7517254741201427  0.5008578708471561  0.9970936075729284
  0.0007044196000763  0.7526630980128033  0.0007044196000763
  0.5008578708471561  0.7517254741201427  0.9970936075729284
  0.8780714465568096  0.1270097694021928  0.1270097694021928
  0.6287484408365884  0.3759308097141343  0.1197215711099417
  0.3759308097141343  0.6287484408365884  0.1197215711099417
  0.8732745258798573  0.6279063924270716  0.1241421291528439
  0.1270097694021928  0.8780714465568096  0.1270097694021928
  0.6279063924270716  0.8732745258798573  0.1241421291528439
  0.0032868013424974  0.0032868013424974  0.2530482512562529
  0.5052784288900583  0.9962515591634116  0.2490691902858657
  0.7511258841353765  0.2501382210586058  0.2501382210586058
  0.9962515591634116  0.5052784288900583  0.2490691902858657
  0.2501382210586058  0.7511258841353765  0.2501382210586058
  0.7501427660060713  0.7501427660060713  0.2491529460899997
  0.6287484408365884  0.1197215711099417  0.3759308097141343
  0.8738741158646235  0.3748617789413942  0.3748617789413942
  0.1197215711099417  0.6287484408365884  0.3759308097141343
  0.6217131986575026  0.6217131986575026  0.3719517487437471
  0.3748617789413942  0.8738741158646235  0.3748617789413942
  0.8748572339939287  0.8748572339939287  0.3758470539100003
  0.2490691902858657  0.9962515591634116  0.5052784288900583
  0.7517254741201427  0.9970936075729284  0.5008578708471561
  0.9962515591634116  0.2490691902858657  0.5052784288900583
  0.7469285534431904  0.4979902305978072  0.4979902305978072
  0.9970936075729284  0.7517254741201427  0.5008578708471561
  0.4979902305978072  0.7469285534431904  0.4979902305978072
  0.3759308097141343  0.1197215711099417  0.6287484408365884
  0.8732745258798573  0.1241421291528439  0.6279063924270716
  0.1197215711099417  0.3759308097141343  0.6287484408365884
  0.6217131986575026  0.3719517487437471  0.6217131986575026
  0.3719517487437471  0.6217131986575026  0.6217131986575026
  0.8723369019871967  0.6242955803999237  0.6242955803999237
  0.1241421291528439  0.8732745258798573  0.6279063924270716
  0.6242955803999237  0.8723369019871967  0.6242955803999237
  0.0007044196000763  0.0007044196000763  0.7526630980128033
  0.5008578708471561  0.9970936075729284  0.7517254741201427
  0.2501382210586058  0.2501382210586058  0.7511258841353765
  0.7501427660060713  0.2491529460899997  0.7501427660060713
  0.9970936075729284  0.5008578708471561  0.7517254741201427
  0.4979902305978072  0.4979902305978072  0.7469285534431904
  0.2491529460899997  0.7501427660060713  0.7501427660060713
  0.7493508173585397  0.7493508173585397  0.7493508173585397
  0.1270097694021928  0.1270097694021928  0.8780714465568096
  0.6279063924270716  0.1241421291528439  0.8732745258798573
  0.3748617789413942  0.3748617789413942  0.8738741158646235
  0.8748572339939287  0.3758470539100003  0.8748572339939287
  0.1241421291528439  0.6279063924270716  0.8732745258798573
  0.6242955803999237  0.6242955803999237  0.8723369019871967
  0.3758470539100003  0.8748572339939287  0.8748572339939287
  0.8756491826414603  0.8756491826414603  0.8756491826414603
```

POSCAR for HV-diamond

```
C_diamond_PBE
  3.5704
  2.000000000000   0.000000000000   0.000000000000
  0.000000000000   2.000000000000   0.000000000000
  0.000000000000   0.000000000000   2.000000000000
  H   C
  2   64
Direct
    0.3100   0.3100 0.3100
    0.250000000000   0.250000000000   0.250000000000
    0.000000000000   0.000000000000   0.000000000000
    0.500000000000   0.000000000000   0.000000000000
    0.250000000000   0.250000000000   0.000000000000
    0.750000000000   0.250000000000   0.000000000000
    0.000000000000   0.500000000000   0.000000000000
    0.500000000000   0.500000000000   0.000000000000
    0.250000000000   0.750000000000   0.000000000000
    0.750000000000   0.750000000000   0.000000000000
    0.125000000000   0.125000000000   0.125000000000
    0.625000000000   0.125000000000   0.125000000000
    0.375000000000   0.375000000000   0.125000000000
    0.875000000000   0.375000000000   0.125000000000
    0.125000000000   0.625000000000   0.125000000000
    0.625000000000   0.625000000000   0.125000000000
    0.375000000000   0.875000000000   0.125000000000
    0.875000000000   0.875000000000   0.125000000000
    0.250000000000   0.000000000000   0.250000000000
    0.750000000000   0.000000000000   0.250000000000
    0.000000000000   0.250000000000   0.250000000000
    0.500000000000   0.250000000000   0.250000000000
    0.250000000000   0.500000000000   0.250000000000
    0.750000000000   0.500000000000   0.250000000000
    0.000000000000   0.750000000000   0.250000000000
    0.500000000000   0.750000000000   0.250000000000
    0.375000000000   0.125000000000   0.375000000000
    0.875000000000   0.125000000000   0.375000000000
    0.125000000000   0.375000000000   0.375000000000
    0.625000000000   0.375000000000   0.375000000000
    0.375000000000   0.625000000000   0.375000000000
    0.875000000000   0.625000000000   0.375000000000
    0.125000000000   0.875000000000   0.375000000000
    0.625000000000   0.875000000000   0.375000000000
    0.000000000000   0.000000000000   0.500000000000
    0.500000000000   0.000000000000   0.500000000000
    0.250000000000   0.250000000000   0.500000000000
    0.750000000000   0.250000000000   0.500000000000
    0.000000000000   0.500000000000   0.500000000000
    0.500000000000   0.500000000000   0.500000000000
    0.250000000000   0.750000000000   0.500000000000
    0.750000000000   0.750000000000   0.500000000000
    0.125000000000   0.125000000000   0.625000000000
    0.625000000000   0.125000000000   0.625000000000
    0.375000000000   0.375000000000   0.625000000000
    0.875000000000   0.375000000000   0.625000000000
    0.125000000000   0.625000000000   0.625000000000
    0.625000000000   0.625000000000   0.625000000000
    0.375000000000   0.875000000000   0.625000000000
    0.875000000000   0.875000000000   0.625000000000
    0.250000000000   0.000000000000   0.750000000000
    0.750000000000   0.000000000000   0.750000000000
    0.000000000000   0.250000000000   0.750000000000
    0.500000000000   0.250000000000   0.750000000000
    0.250000000000   0.500000000000   0.750000000000
    0.750000000000   0.500000000000   0.750000000000
    0.000000000000   0.750000000000   0.750000000000
    0.500000000000   0.750000000000   0.750000000000
    0.375000000000   0.125000000000   0.875000000000
    0.875000000000   0.125000000000   0.875000000000
    0.125000000000   0.375000000000   0.875000000000
    0.625000000000   0.375000000000   0.875000000000
    0.375000000000   0.625000000000   0.875000000000
    0.875000000000   0.625000000000   0.875000000000
    0.125000000000   0.875000000000   0.875000000000
    0.625000000000   0.875000000000   0.875000000000
```

POSCAR for H2I-diamond

S2 Analyzing the DFT data

There are two diamond systems of 2 × 2 × 2 atoms, the HV-diamond and the H2I-diamond (Figure 2), over four DFT datasets. Histograms (**Figure S2.1**) and the minimum (min), maximum (max), mean, and standard deviation (stdev) (**Table S2.1**), of the total energies, energies per atom (eatom), and normalized forces are shown below per dataset.

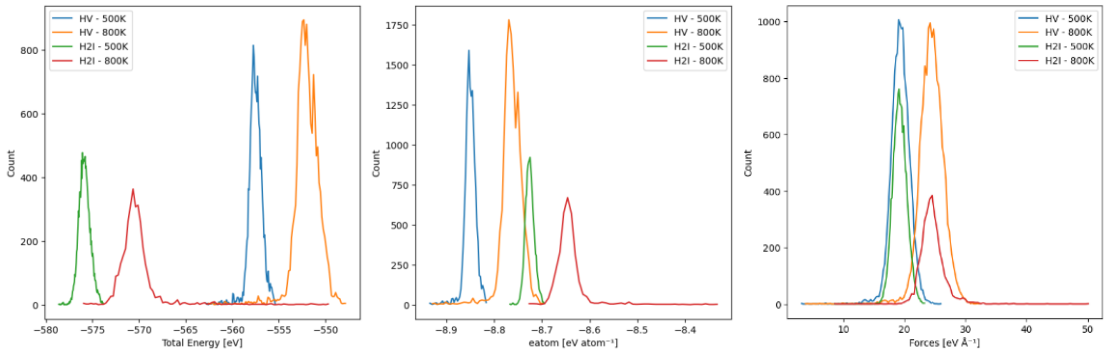

Figure S2.1. Histograms of the total energy (left), eatom (middle), and normalized forces (right) for the four datasets.

Table S2.1. The min, max, mean, and stdev values for the total energy, eatom, and normalized forces for the four datasets.

Dataset	HV – 500 K	HV – 800 K	H2I – 500 K	H2I – 800 K
Total Energy [eV]				
Min	-562.880	-562.542	-578.607	-575.943
Max	-555.303	-547.709	-573.823	-549.431
Mean	-557.484	-551.927	-575.819	-570.086
Stdev	0.768	1.375	0.608	2.333
eatom [eV atom^{-1}]				
Min	-8.935	-8.929	-8.767	-8.726
Max	-8.814	-8.694	-8.694	-8.325
Mean	-8.849	-8.761	-8.725	-8.638
Stdev	0.012	0.022	0.009	0.035
(normalized) forces [eV Å^{-1}]				
Min	3.198	3.832	8.622	8.622
Max	26.208	32.532	23.426	50.497
Mean	19.440	24.569	19.321	24.802
Stdev	1.576	2.052	1.138	2.079

For all the data together, the total energies are in the approximate range [-580;-545] eV, the energies per atom in the approximate range [-9.0;-8.3] eV atom^{-1}, and the normalized forces in the approximate range [3;51] eV Å^{-1}. The complete dataset (after filtering), consisted of 47192 data samples.

Further analysis shows that the volumes of the four systems are in the approximate range [340;375] Å^3. In the histogram (left, **Figure S2.2**), it can be seen that there are two peaks for each system. This can be clarified by looking at a zoomed in part of the volumes over time (right, Figure S2.2), where volume oscillations are visible. As there is a periodicity of the volume over time, one has to be careful if points are extracted with the same number of ionic steps separated from each other. In future MLIP models, when more data is available, this type of selection is needed to break correlation between images.

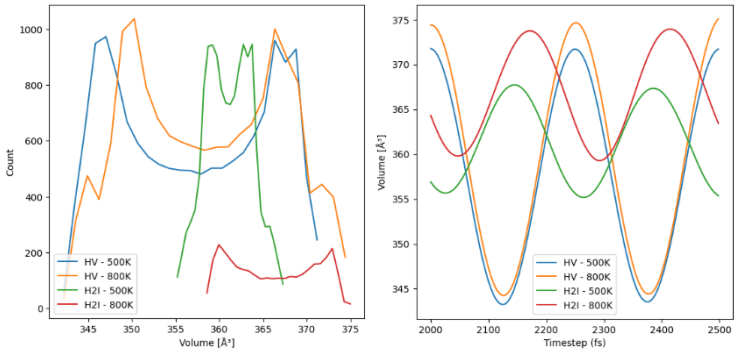

Figure S2.2. (left) Histogram of the volume, (right) zoomed-in plot of the volume over time, for the four datasets.

S3 Extra MLIP details

A short overview of model parameters (**Table S3.1**) and logs of the training progress over epochs (**Figure S3.1**), are shown below. Some evaluations of these models can be found in supporting information S4.

Table S3.1. Some parameters for the ePotentia trained models.

Model	kp	kf	freezing epochs	epochs	# train samples	# val samples	Transfer learned on:
01	10	1	4	6	4096	1024	Oc
02	10	1	4	16	4096	1024	01
04	10	1	4	32	8192	2048	Oc
05	10	1	10	50	/	/	Oc
06	20	1	2	10	/	/	05
07	20	1	4	16	4096	1024	05
08	10	1	1	5	/	/	Oc
10	15	1	10	50	/	/	Oc
11	15	1	10	32	8192	2048	10
12	15	1	10	50	/	/	Oc

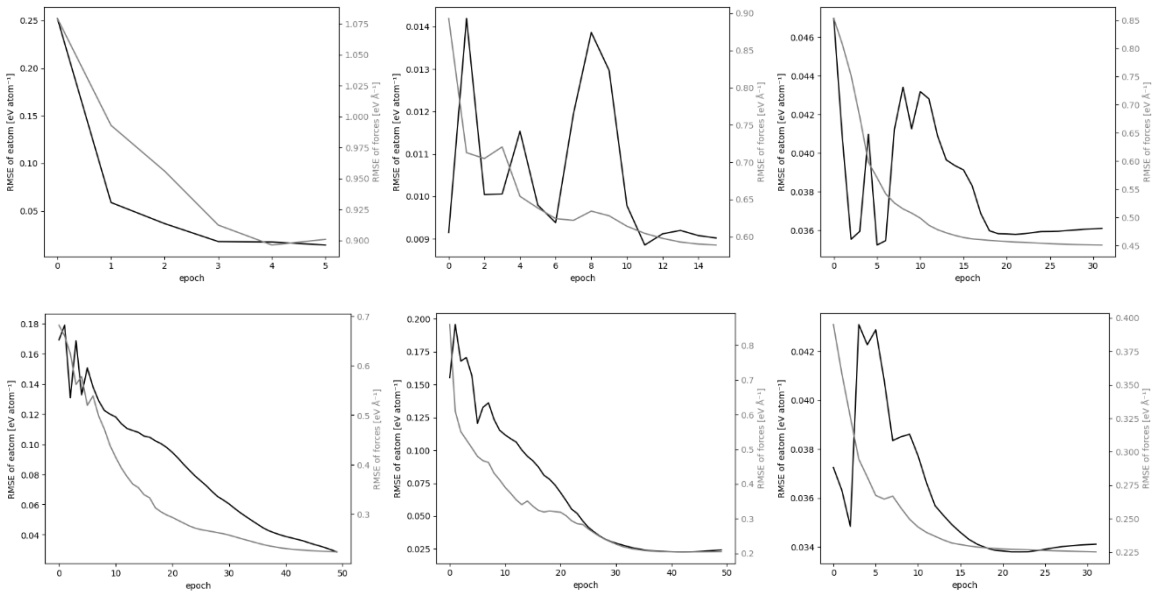

Figure S3.1. The RMSE of the energy per atom (left axis, black) and the normalized forces (right axis, gray) over the training epochs. Logs for models (upper left) 01, (upper middle) 02, (upper right) 04, (lower left) 05, (lower middle) 10, (lower right) 11.

S4 Model evaluations

Some extra model evaluations are shown in **Figure S4.1** and **Figure S4.2**. The datapoints in the test database in those following figures are colored based on their original dataset.

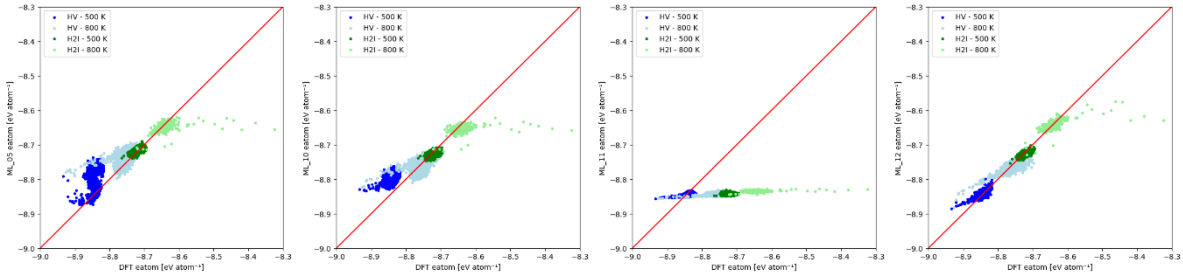

Figure S4.1. Correlation plots of the predicted energy per atom for (left) model 05, (left-center) model 10, (right-center) model 11, (right) model 12. The colors indicate the data source: (dark blue) HV-diamond 500 K, (light blue) HV-diamond 800 K, (dark green) H2I-diamond 500 K, (light green) H2I-diamond 800 K.

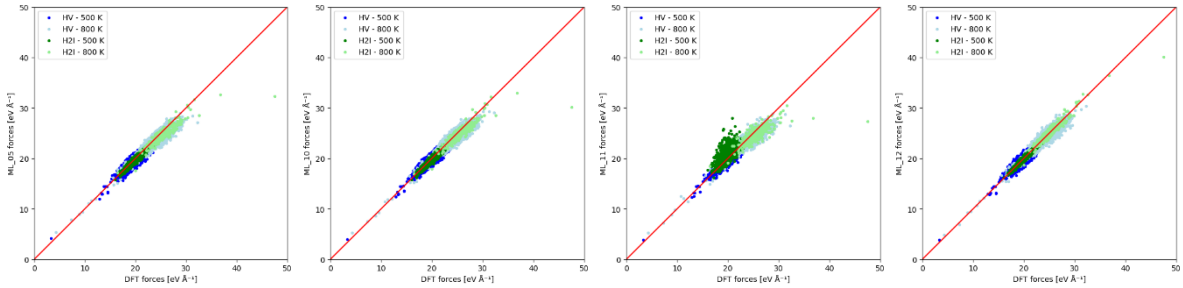

Figure S4.2. Correlation plots of the normalized forces for (left) model 05, (left-center) model 10, (right-center) model 11, (right) model 12. The colors indicate the data source: (dark blue) HV-diamond 500 K, (light blue) HV-diamond 800 K, (dark green) H2I-diamond 500 K, (light green) H2I-diamond 800 K.

S5 NVT MD further analysis

Quantifying the movement of the atoms in the NVT simulations, is done in the following steps:

- Open the trajectories for the chosen NVT simulation
- Write POSCARs for every step of this trajectory
- Convert the coordinates from these POSCARS from direct to cartesian coordinates
- For each timestep:
    - Calculate the center of mass (COM)
    - For each atom:
    - Subtract the COM distance from the atom coordinates
    - Normalize the coordinates into one distance value (using Pythagorean theorem)
- Choose a few atoms to plot their distance from the COM over the time of the simulation

The resulting plots are shown in **Figure S5.1**.

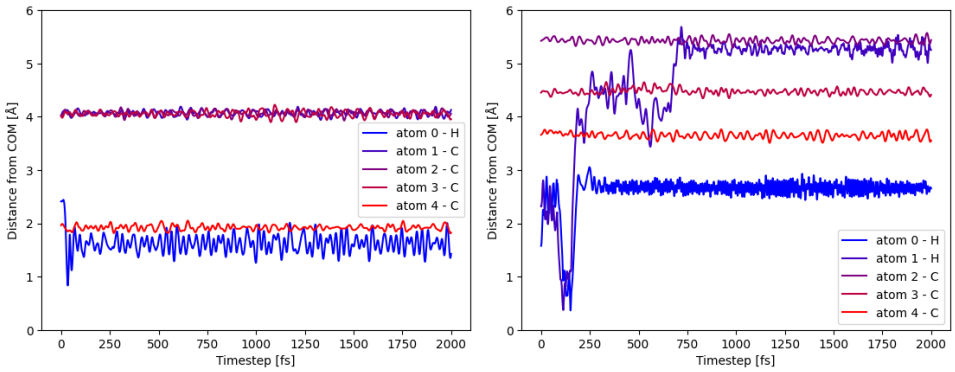

Figure S5.1. The distance from each atom to the COM over time for (left) HV, and (right) H2I. Values from the NVT MD run for $T = 500$ K. Only a few atoms from each system are plotted, so as not to crowd the figures.