Karlsruhe Institute of Technology

Machine Learning Methods for Direct Simulations of Charge and Exciton Transfer

Zur Erlangung des akademischen Grades eines

DOKTORS DER NATURWISSENSCHAFTEN

(Dr. rer. nat.)

von der KIT-Fakultät für Chemie und Biowissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

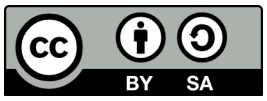
von

Mila Vladimirova Krämer (geb. Andreeva)

aus Sofia (Bulgarien)

1. Referent: Prof. Dr. Marcus Elstner
2. Referent: Prof. Dr. Wolfgang Wenzel
Tag der mündlichen Prüfung: 19. Juli 2021

Karlsruher Institut für Technologie
Fakultät für Chemie und Biowissenschaften
Kaiserstraße 12
76131 Karlsruhe



Abstract

Computer simulations are an invaluable tool elucidate processes occurring on the atomistic scale. The more precise the description of a system, however, the more computationally demanding the calculation becomes. This is especially valid for calculations where quantum mechanics (QM) methods must be used to calculate the electronic structure of large systems and extensive sampling along molecular dynamics (MD) trajectories is needed to obtain the quantities under investigation. Propagation processes where charges or molecular excitations travel through a system require especially costly methods due to the need to solve the time-dependent Schrödinger equation. The description of these processes is essential for the design and development of novel organic semiconductors, as well as for the understanding of biochemical processes involved in photosynthesis or vision. To make such simulations feasible, a menagerie of approximate methods such as semi-empirical density-functional tight binding (DFTB) have been developed, which reduce computational costs by replacing parts of the calculation with parametrized functions. Using DFTB and a scheme to reduce the complexity of the system to essential electronic parameters, direct simulations of charge transfer can be performed, but significant limitations to the treatable system size remain. In the last decade, data-driven machine learning (ML) methods have become widely used in computational chemistry and promise to upset the long-established compromise between speed and accuracy.

In this work, I present several ML methods for use in simulations of charge and exciton transfer, trained and evaluated on organic semiconductors, which are also applicable in other contexts such as biological light-harvesting systems. When trained to several thousand reference values from DFTB for structures sampled from MD simulations, these models can replace DFTB during propagation simulations and reduce the computational costs of the simulation. A first method using kernel ridge regression (KRR) was trained on charge and exciton transfer couplings and reproduced hole transfer mobilities in anthracene to within 8 % of the DFTB reference and 30 % of experimental values. As this method could not provide the forces necessary for the description of relaxation processes and calculation of non-adiabatic coupling vectors (NACVs), I developed a second method on the basis of neural networks (NN) which was able to provide these quantities. First tests of these models in computer simulations showed that the quality of the forces is crucial for the stability of the simulations, as even infrequent occurrences of outlier predictions can cause simulations to crash. With careful management of the training data set, this challenge could be overcome, resulting in models ready to be applied in charge transfer propagation. A third type of models I developed is able to simultaneously predict different types of excitonic couplings at a small fraction of the computational costs of the LC-TD-DFTB reference methods. The models presented here can be easily applied for further systems as they can be automatically trained and included in multi-scale workflows.

Zusammenfassung

Computersimulationen sind ein wertvolles Hilfsmittel um Prozesse auf atomaren Skalen zu beleuchten. Je genauer die Beschreibung des Systems, desto aufwändiger die Rechnung, was in besonderem Maße für Rechnungen gilt, in denen quantenmechanische (QM) Methoden notwendig sind, um die Elektronenstruktur großer Systeme zu beschreiben und die zu untersuchenden Systemeigenschaften aus langen Molekulardynamik (MD)-Simulationen berechnet werden. Propagationsprozesse, in denen sich Ladungen oder molekulare Anregungen durch ein System fortbewegen benötigen besonders aufwändige Methoden, da dafür die zeitabhängige Schrödingergleichung gelöst werden muss. Allerdings sind ebendiese Prozese essenziell für die Entwicklung organischer Halbleiter und das Verständnis lichtaktivierter biochemischer Prozesse. Um derartige Simulationen zu ermöglichen, wurden approximative Methoden wie die semiempirische Dichtefunktionalbasierte *tight binding* (DFTB) entwickelt, die den Rechenaufwand verringern indem Teile der Rechnung durch empirisch parametrisierte Funktionen ersetzt werden. Wird DFTB zusammen mit einem Schema zur Reduktion der Komplexität des Systems auf wesentliche elektronische Parameter verwendet, können direkte Simulationen von Ladungstransfer durchgeführt werden, wenn auch mit eingeschränkter Systemgröße. In den letzten zehn Jahren haben die datengetriebenen Methoden des machinellen Lernens (ML) ihre Anwendungen in der Quantenchemie gefunden und versprechen, den Kompromiss zwischen Geschwindigkeit und Genauigkeit aufzubrechen.

In dieser Arbeit stelle ich mehrere ML Methoden vor, die für Ladungs- und Excitonentransfersimulationen anstelle von DFTB verwendet werden können, die ich auf organischen Halbleitern trainiert und evaluiert habe, die aber auch in anderen Systemen (z.B. aus der Biochemie) anwendbar sind. Diese Methoden werden auf mehreren tausend DFTB-Referenzpunkten trainiert die für MD generierte Strukturen berechnet werden. Eine erste Methode basiert auf dem Formalismus der Kernelregression und war in der Lage die für Ladungs- und Excitonentransfersimulationen notwendigen Parameter zu lernen und die Lochtransfermobilitäten in Anthracen bis auf 8 % der Referenz und 30 % des experimentellen Werts zu reproduzieren. Da diese Methode nicht in der Lage war, die für molekulare Relaxationsprozesse und die Berechnung von nichtadiabatischen Kopplungen notwendigen Gradienten zur Verfügung zu stellen, entwickelte ich zu diesem Zweck eine zweite Methode auf Basis Neuronaler Netze. Erste Tests dieser Netze in Simulationen zeigten dass selbst vereinzelt vorkommende falsche Vorhersagen katastophale Folgen für die Simulationen haben können. Durch Gestaltung des Trainingssatzes gelang es mir, Modelle zu erzeugen, die für die Anwendung in Simulationen bereit sind. Eine dritte Art von Modell kann mehrere Arten von Excitontransferkopplungen mit nur einem Bruchteil der Kosten der Referenzmethode berechnen. Die hier vorgestellten Methoden können einfach für neue Systeme angewandt werden, da sie automatisiert trainiert und in multiskalen-Simulationsprozesse eingebaut werden können.

Thanks

Thank you, Marcus Elstner for the warm welcome to the TCB group and for fostering a great discussion and research environment, as well as for your frequent encouragement and valuable guidance. To Philipp Dohmen, thank you for our fruitful and enjoyable discussions about quantum chemistry, programming and machine learning, and for providing me with fresh data and simulation results. Thank you, Weiwei Xie and Daniel Holub, for the pleasant and productive collaboration, and to you, Tomáš Kubař for letting me pester you with programming questions. To everyone in the TCB group and the members of RTG 2450, thank you for the help, advice, expertise, motivation, discussions, shared successes and frustrations, pizza and cake. A round of thank-yous also to all my neighbors in the 6th floor of building 30.45 for a great time in the "Girls' Tower", and especially to you, Beatrix Bold, for being a great office partner. Also, Anatole von Lilienfeld, thank you for hosting me in Basel for a while, and thanks to your group for being generally awesome.

Finally, thank you to the usual suspects – my husband, family, and friends – for unwavering support and welcome distraction. You're the best!

Contents

Part I.

Introduction and Background

1. Introduction

Throughout most of humanity's history, processes such as the burning of wood to coal or the fermentation of sugar to alcohol have been useful and fascinating, but opaque. Attempts to explain these macroscopically observable transformations of matter began with the development of the various elemental theories used in the ancient world, which transitioned into the alchemical experiments of the medieval period. After the formalization of scientific methods, chemistry emerged as its own field, but was highly interconnected to physics, geology, biology, medicine and engineering.

For a long time, progress in chemistry was limited by scientists' inability to look into matter, determine what it was made of and watch it interact. The discovery of electrons and nuclei, the exploration of the periodic table and the quantum-mechanical description of the states and interactions of atoms provided valuable tools for understanding the microscopic behavior during chemical reactions. The development of experimental methods for measuring or visualizing molecular structures such as molecular spectroscopy or X-ray diffraction enabled new insights into the processes occurring on an atomic scale. Today, methods are available for obtaining images with atomic resolution, or measuring spectra of reaction mixtures in femtosecond intervals.

However, these elaborate experimental setups put severe constraints on the reaction conditions, making the investigation of e.g. biochemical reactions quite difficult. To 'look into the beaker' via computer simulations can avoid these constraints and has other advantages compared to experimental studies. For drug and materials discovery, for example, computer simulations can filter out large amounts of candidates with unfavorable properties without the need to invest in synthesis, and thereby make the process cheaper and faster. Using one of many available mathematical formalisms for describing the behavior of atoms and molecules and given a starting condition, a chemical reaction can be simulated and visualized without the need for an experiment. While these simulations can be exceedingly precise, they are also quite computationally demanding. Three main factors determine computational costs of simulations:

First is the level of detail in which atoms or molecules are treated. Molecular mechanics (MM) methods treat atoms as hard spheres, the bonding interactions between them are modeled as harmonic springs, and dynamics are calculated using Newton's equations of motion. These interactions are inexpensive to calculate and conceptually simple, but rely on empirically determined parameters, which must be adjusted for a given system to give good results. MM methods can describe molecular motions quite well, especially in rigid or macromolecular structures, but struggle at capturing the complexities of bonds breaking and forming and are thus of very limited use for simulations of reactivity. On the other hand, quantum-mechanical (QM) methods precisely describe the interactions between nuclei and electrons, with very few approximations and assumptions. Chemical reactivity naturally emerges from QM simulations, and a QM description is essential for describing

the interaction of molecules with light. All QM methods rely on solving some form of the Schrödinger equation to obtain results for the states of the system. The resulting calculations are highly complex, and although some approximations can be made, the computational cost is orders of magnitude higher than for an MM description of the system. However, there is a plethora of methods which build on the QM formalism but introduce approximations which avoid computationally costly steps in the calculation by replacing them with simple, parametrized functions. The parameters for these *semiempirical* methods are chosen such as to reproduce the results obtained with a higher-level method as closely as possible. Semiempirical methods have proven to be a useful tool for simulations of chemical reactivity, as they provide a quantum-mechanical foundation describing the making and breaking of bonds. They are orders of magnitude faster than true *ab initio* methods, but also far slower than MM models. Density-functional tight binding (DFTB)[1, 2, 3, 4, 5] is a semiempirical method based on density functional theory (DFT) which is very fast due to extensive parametrization of all integrals which would otherwise be calculated during the calculation. It is well-suited for calculations involving organic and biological molecules and molecular dynamics simulations.

The second main factor for the computational cost of an atomistic simulation is the size of the systems to be simulated. All simulation methods become more costly, the more particles (e.g. atoms, electrons) are to be calculated. However, the increase in computational cost scales differently for different methods: While the costs of MM methods usually scale almost linearly with the number of particles, QM methods usually scale cubically or worse, limiting the applicability of QM methods to large systems. This especially concerns biological systems with hundreds of thousands of atoms, but is also an issue in materials science, where the properties to be calculated for a given material only emerge in the bulk phase.

The third main factor is the length of time to be simulated. While photochemical processes can be completed within picoseconds or less, molecular motions can be quite slow, especially in large or stiff molecules. Chemical or enzymatic reactions frequently occur on the timescales of seconds. In molecular dynamics simulations, the development of the system over time is calculated one step at a time, by starting from an initial structure and using energies and forces calculated with the method of choice to calculate the positions of the atoms at the next point in time. The length of these time steps cannot be arbitrarily large, as large atomic displacements can result in physically unreasonable collisions. For QM methods, maximal reasonable step sizes are fractions of a femtosecond. While for MM methods the step size can be a bit larger, even a nanosecond of simulation time can require hundreds of thousands or even millions of calculations of energies and forces. In this context, even a small change in the computational cost for an individual step can greatly affect the cost for the entire simulation.

The drastic increase in the available computational power in the last few decades has meant that ever longer, more precise simulations of ever larger systems have become possible. With a gargantuan investment of supercomputing resources, even an all-atom MM simulation of an entire bacterial cytoplasm [6, 7] was made possible. However, even in less extreme cases, atomic-resolution simulations still need countless hours of supercomputer time, inspiring development of further fast yet precise methods.

One valuable technique is the combination of QM and MM methods within a single simulation, where the MM method is used to simulate the large, non-reactive environment (e.g. a membrane, a protein backbone, water), while the QM method only describes the behavior of the small, reactive part of the system (e.g. the catalytic center of an enzyme). These *hybrid QM/MM* methods have been successfully used in both biochemistry and materials science applications. However, even if a fast semiempirical method is used as the QM part of such an approach, the size of the subsystem relevant for the reactivity can be large enough that this approach is not feasible.

Improvements in reactive force fields, more clever approximations for the physics of bond formation, better implementations of MD algorithms or the continuing miniaturization and falling costs of computational processing power have slowly and steadily been expanding the horizon of systems treatable by simulation. However, in the last decade, one technology has upset the field and promises to accelerate this progress by leaps and bounds. The data-driven, statistically-motivated methods which form the field of machine learning (ML) began gaining traction in both applied and natural sciences after a few breakthrough developments in the mid to late 2000s.

The core idea behind ML methods is to extract patterns from a large set of example data points using statistical methods. An ML *model* then uses these patterns to predict properties for data points which were not part of the reference data set. Examples for widespread use of ML models outside research contexts are techniques for image recognition, fraud detection or risk assessment, as well as suggestion algorithms in social media and advertising or self-driving cars. Mathematically, an ML model is nothing more than a very flexible function heavily relying on parameters which are adjusted during model *training* so that the resulting model best reproduces the input–target relationships learned from the training set. Compared to the 'old-school' empirical models which have always been used in the sciences for modeling functional relationships, ML models are distinguished by the fundamental approach to model design:

Non-ML mathematical models in e.g. physics are constructed with a lot of domain knowledge motivating the design decisions such as the functional form of the model. Parameters are universal or system-specific constants and are usually assigned some physical meaning. This also holds for parameters of empirical and semiempirical quantum chemistry methods such as the specific amount of exchange contribution included in a hybrid DFT functional. In contrast, the functional forms underlying ML models are deliberately chosen to be as universally applicable and non-specific as possible. The trainable parameters of ML methods are not a priori connected to any factor a non-ML model would include in its description of the problem, hindering interpretation of ML models and explanation of the reasoning behind a given prediction in many cases. Techniques exist for analyzing ML models to explain a given prediction or constructing models which are interpretable to begin with, but such analysis or design is not always possible.

The flexibility of ML models allows them to be applied to problems, in which we do not understand the dependencies and functional relationships between the variables well enough to construct a specific model. The use of ML models is not limited to such problems, however. While they have large numbers of parameters, the functional form of the models

is usually quite simple. Therefore, ML can also be used in cases where domain-specific models exist, but are too computationally costly to evaluate.

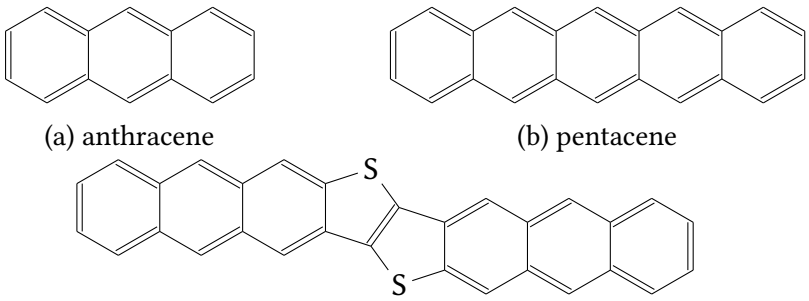
In recent years, it is this latter application which has led to the widespread adoption of ML methods by theoretical chemists and physicists[8]. They can complement the menagerie of classical and quantum mechanical methods available for atomistic calculations or simulations with the low computational costs they incur on prediction and the potential for high accuracy if trained on the right data set. I want to highlight three distinct areas in which ML methods can be applied in computational chemistry contexts. This list is far from exhaustive, and new approaches and applications continue to be published every day.

First is the straightforward application of the models to solve a problem for which a (costly) reference method exists. Here, the model is provided with inputs (usually structural information about a system) and reference values, and can calculate the target value (e.g. energy, macroscopic observables, ...). Models have been constructed to calculate energies and forces for MD simulations[9], aid in catalyst design [10], drug discovery [11, 12], or materials discovery [13, 14, 15] and have been trained to a wide variety of molecular electronic properties [16].

A second application is to train an ML model to only handle part of the calculation necessary to obtain the desired result. ML models can be used to provide intermediate results, such as exchange-correlation functionals for DFT [17, 18] or solutions of the Schrödinger equation [19]. They can also be used to bridge the gap between fast but approximate methods and *ab initio* QM calculations by learning the difference between the values for a given property provided by the two methods. This Δ-ML approach[20] can drastically increase the accuracy of results obtained by semiempirical models at an added cost that is negligible to that which would be incurred by use of a more accurate method[21]. It also reduces the risk of catastrophic predictions, as in cases where the ML model predicts badly, the approximate method forms a baseline for the quality of the result.

Third, ML models can be used to predict the parameters used in semi-empirical methods [22]. This can aid in parametrization [23] or make previously static parameters geometry-dependent to mitigate some of the method's inherent approximations[24]. This approach has the advantage that the prediction is used for parameters which were empirically fitted in the first place. The predicted parameters are then used in a robust theoretical framework.

However, the statistical nature of the ML models also results in some caveats and limitations in their applicability: While the flexibility of the functions constituting the model is a boon for their ability to approximate arbitrarily complex functional dependencies, the flip side of that flexibility is that models can behave very erratically when they are asked for predictions outside the scope of their training. Care must be taken to thoroughly investigate trained models' behavior, as well as assess to which extent the model is reliably able to generalize. This is especially important when a model's predictions are coupled to its inputs, as is the case in ML-driven molecular dynamics simulations where the forces predicted by a model for a given geometry are used to generate the geometry it faces in the next step. Techniques for estimating the uncertainty of each prediction exist can be used to recognize when a model is not predicting well and fall back on more accurate methods.



(a) anthracene (b) pentacene

(c) DATT

Figure 1.1.: Three organic semiconductors used in this work.

Another limitation especially in the context of quantum chemistry is the computational cost of obtaining high-quality training data in large enough amounts to be able to train an ML model. If a model is trained for a very specific purpose, the cost of obtaining the training data for the model may outweigh the benefits gained through its application.

In this work, I focus on the first of these approaches, experimenting with ML models which are able to predict electronic properties of organic materials, more specifically molecular organic semiconductors (OSCs). These molecules are usually large and fairly rigid aromatic polycycles such as anthracene, pentacene or Dianthra[2,3-b:2′,3′-f]thieno[3,2-b]thiophene (DATT)[25] (Figure 1.1).

In contrast to the inorganic semiconductors which have been at the base of the computational revolution for the last 60 years, organic semiconductors have only begun seeing widespread use in the last decade. They are lightweight, easy to fabricate and process, their properties can be tuned via their molecular structure, and they allow for new applications such as flexible or transparent electronics. OSCs are now in use for photovoltaics [26, 27, 28], organic LEDs [29, 30], or organic field effect transistors [31, 32].

The potential number of candidate organic molecules for these applications is astronomical, so efficient techniques are needed to search chemical space, find promising molecules and systematically improve upon known compounds. Experimental high-throughput screening and automated synthesis protocols can be augmented through the use of computer simulations. If the suitability of candidate materials for a given task as quantified by an observable property can be calculated in a simulation, the unsuitable candidates can be quickly discarded and only the most promising candidates can be analyzed further.

However, for OSCs, the properties in question are a system's ability to transport electrons and, for LED or photovoltaics applications, its ability to form and transport *excitons*, quasiparticles which represent molecular excitations. Calculating these properties to a reasonable degree of accuracy requires electronic structure methods and thus the costly solution of the Schrödinger equation, a cost that is compounded by the fact that these are properties of the bulk material, requiring simulations with hundreds of thousands of atoms. While there are model theories which circumvent these issues by describing the propagation of charges in a more coarse-grained manner, these theories only apply in one of two limiting cases:[33] If the charge is localized on individual molecules and moves by hopping from molecule to molecule by overcoming an activation barrier, the transfer is said to be *hopping-like* and can be modeled using kinetic theories using empirically determined

parameters. If instead the charge is fully delocalized, the transfer becomes *band-like* and can be modeled using band theories developed for inorganic materials. OSCs, however, frequently fall in the middle between these two regimes, where neither of the assumptions holds and transfer must be simulated by solving the time-dependent Schrödinger equation and performing non-adiabatic molecular dynamics. Various schemes have been developed to obtain charge transfer properties from computer simulations[34], but they are all limited by the computational costs resulting from the large system size, long simulation times, and need for electronic structure calculations in each simulation step. For optoelectronic properties, the situation is even more complex, as the nature of exciton dynamics requires excited-state calculations and is even more computationally demanding.

There are methods which reduce the complexity of the direct simulations of charge or exciton transfer, which decouple the relevant electronic degrees of freedom from the system at large[35, 36, 37], decompose the system into individual molecular fragments and reduce the number of parameters which describe the electronic structure in order to make solving the time-dependent Schrödinger equation far easier. The parameters necessary for the simulation are the energies of every *site* where the charge or exciton can be located and the coupling matrix elements between these states. For every individual molecule, these parameters can be calculated during a simulation using an *ab initio* or semiempirical method, in the case of charge transfer even without the need for a time-dependent theory. However, the extensive sampling along a molecular dynamics trajectory requires the repetitive calculation of these properties, and the large costs incurred by the size of the individual OSC molecules and the entire crystalline or amorphous system limits the systems it is reasonably possible to simulate.

In this work, I demonstrate how machine learning models can be used to push the limits of these simulations further. The computationally costly step in these simulation schemes is the QM calculation of the electronic structure of every molecular fragment in the system. This step is repeated over and over again, in every step, for overall very similar structures. The entire electronic structure is then reduced to a set of site energies of the relevant orbitals for every fragment and the pairwise couplings between these states. These quantities, collected in the form of a Hamiltonian, then form the input for the time-dependent Schrödinger equation, solving which is only a small computational cost compared to the individual electronic structure calculations. Therefore, by training ML models which can calculate good approximations for the site energies and couplings (and any other quantities required for the specific simulation) without the need to obtain the electronic structure for every fragment, the method can be sped up significantly.

I have developed several techniques for training different types of machine learning models for the prediction of charge and exciton transfer properties of organic semiconductors. I train both kernel ridge regression and neural network models for this application, and evaluate the ML models not only in terms of static metrics on a reference data set, but also show that they can indeed be used to drive direct charge and exciton transfer propagation simulations from which experimental observables can be calculated.

This work is structured as follows: In chapter 2 I summarize the quantum chemistry methods used in the context of this work, both the reference methods for data generation and the fundamentals of molecular dynamics simulations. Subsequently, in chapter 3 I provide an introduction to the terminology and toolbox of machine learning methods and

explain how both kernel ridge regression and neural network models are designed, trained, and evaluated.

In chapter 4 I present my work on training kernel ridge regression models for use in charge and exciton transfer simulations on reference data for the anthracene crystal obtained from the DFTB method. When provided with the geometry of a fragment or fragment pair, these models are able to accurately predict the site energies and couplings. Additionally, I describe the process of determining a suitable representation for the molecular geometry for a given purpose, elaborate on the training and evaluation procedure for the models and finally show that they can indeed replace DFTB as drivers of propagation simulations.

The discussion and analysis of the limitations encountered using the models during the simulations motivate the work I present in chapter 5. There, I train neural network models to predict not just the elements of the Hamiltonian for propagation, but their gradients with respect to atomic coordinates as well. These gradients are necessary for the model to be able to describe the relaxation of a molecule when its occupation changes, a process whose accurate description is crucial for obtaining physically sensible values for materials properties from simulations. I discuss the challenges I encountered in training the models and finding good hyperparameters for them, as well as discuss the process necessary to ensure that the forces predicted from trained models are sufficiently accurate to produce stable molecular dynamics simulations. Also, I discuss the implementation of the ML prediction in the program used to drive the simulations and draw performance comparisons between DFTB- and ML-driven simulations. I apply the training framework I developed for anthracene in this chapter to DATT as well, and show how the hyperparameters obtained for anthracene generalize for that system.

In chapter 6, I describe a flexible neural network architecture for learning various types of excitonic couplings, which I train and evaluate on data for the pentacene molecule. Finally, I use chapter 7 to summarize the results I have obtained and the challenges I encountered, and present a brief overview of the natural next steps for the application and further refinement of the models and techniques presented in this work.

2. Computational Chemistry Toolbox

The various mathematical descriptions of matter on the atomistic level that were developed during the last century all serve different purposes. One of two paradigms lies at the core of every such method, determining the degree of precision to which the mathematical model can reproduce experimentally observed behavior. The first is the assumption that atoms and molecules behave according to the laws of classical Newtonian mechanics. Modeling atoms as hard balls and the interactions between them as harmonic springs connecting them is an approach called 'molecular mechanics' (MM). Within the MM formalism, however, it is difficult to describe the breaking and formation of bonds, effects resulting from electronic delocalization, the molecular or atomic spectra obtained from the interaction of matter with light, and all other effects which require the concept of electronic states to be explained.

In contrast, quantum mechanics (QM) was specifically developed to properly describe the interactions between nuclei, electrons, and other subatomic particles. The QM formalism takes into account the wave-particle duality and the quantization of properties previously assumed to be continuous which occurs at small scales. While a QM description has the potential to fully describe the behavior of atoms and molecules, for any non-trivial system the resulting equations do not have readily available (analytical) solutions or are not solvable at all. Even when solutions can be derived, the computational costs associated with calculating them can become astronomical even considering the immense computational power available across supercomputing facilities today. This is additionally exacerbated by the fact that for most applications, calculating a description for the system for a given static geometry does not give much insight. Instead, the properties of interest (e.g. chemical reactions, their rate constants, molecular excitations and relaxations) only emerge from the dynamics of the system as it evolves over time. Calculating molecular dynamics requires an individual calculation for every time step and precisely controlling relevant parameters such as temperatures and pressures to obtain the proper thermodynamic conditions.

To alleviate the problem of exploding computational costs, numerous methods have been developed by introducing successive approximations to the core QM formalism. These methods each find their own compromise between speed and accuracy in order to be suitable for a specific application. Especially popular is the class of *semi-empirical* models, which reduce parts of the calculation to easy-to-evaluate parametrized functions, whose parameters are fitted to reproduce reference values from more accurate methods or experiments. The density-functional tight-binding (DFTB) method is a semi-empirical method based on the formalism of density functional theory (DFT). DFTB is used for many organo-chemical and biochemical applications and as a reference method for the machine learning models presented in this work.

In the following sections, I first introduce the basic terminology needed for understanding molecular dynamics simulations, independently of whether a QM or MM method is

used to drive the simulation. Subsequently, I briefly introduce the MM formalism and give a longer introduction to quantum mechanics and the DFT and DFTB methods. Finally, I discuss the scheme used for charge and exciton transfer simulations in chapter 4, chapter 5 and chapter 6.

Unless otherwise specified, the information presented in this brief overview can be found in more detail in standard computational chemistry textbooks such as [38, 39].

2.1. Molecular Dynamics (MD) Simulations

Molecular dynamics is the term for a variety of methods which are able to describe the behavior of a system as its state changes over time. In this section, I give a brief introduction to the core concepts these methods share regardless of whether they use quantum or classical mechanics.

A full description of the state of a system at a given time can be determined from the positions and momenta of all particles therein. The sequence of points in this state space passed through during a simulation is called a trajectory. With a given starting point for the trajectory, all subsequent points the system passes can be calculated. The total energy of the system is the essential quantity that must be provided by the mathematical model underlying the simulation, be that molecular mechanics, quantum mechanics or even a data-driven ML model. Given an expression for the energy, the forces acting on the particles in the system (usually the nuclei) can be calculated as the gradients of the energy w.r.t. atomic coordinates. From the forces, the new positions and momenta of the particles are then calculated by integrating the equations of motion.

If the dynamics is performed using an MM method, or a QM method within the Born-Oppenheimer approximation (section 2.3), Newton's equations of motion are used for the nuclei. In this manner, the energy can then be written as a function of the nuclear coordinates, and the system is said to be moving on a potential energy surface (PES) corresponding to a given electronic state. In non-adiabatic molecular dynamics (NAMD) methods, the Born-Oppenheimer approximation is not used, and the nuclear and electronic wave functions are not separable.

For most systems, the integration cannot be done analytically and one of several approximations (e.g. the Euler, Runge-Kutta, or Velocity-Verlet methods) must be used to integrate numerically. These methods all use a finite *time step* Δt at which forces are evaluated and positions updated. The length of the time step is crucial for both the accuracy of the simulation (the shorter the time step, the better the integration) and the computational cost, as each time step requires an evaluation of the energy and its gradients. As a rule of thumb, the time step should be at least one order of magnitude shorter than the fastest motion in the system, which are usually bond vibrations (occurring close to the femtosecond scale). Thus, for a 1 ms simulation of e.g. a protein folding, 10×10^{12} to 10×10^{13} simulation steps are required, and at a rate of 1 µs computational time needed per step, such a simulation would take over 16 weeks. This ballpark estimate again highlights the importance of keeping the computational cost for the evaluation of energy and forces for an individual structure as low as possible.

In order to obtain observables from the MD simulations which bear some resemblance to the processes occurring in reality, there are several thermodynamic parameters which must be considered in the simulation. Thermostats emulate the effect of a fixed temperature, while barostats allow the same for pressure. Both are essential to reproduce the ensembles from statistical mechanics and allow access to thermodynamic variables of the system such as free energies.

I will not discuss the intricacies of molecular dynamics schemes in detail here, as they are not crucial for the rest of this work, but refer the interested reader to references [38, 39]. As a final note, however, I want to emphasize the importance of the forces for the success of a simulation. In order to not break conservation of energy during the simulation, the forces must be consistent with the energy. Having an analytical expression for the gradients of the energy function is therefore preferable to a numerical approximation. If an approximate method is used to calculate the energy, any regions where the approximation is bad can lead to erroneous forces. An excessive overestimation of a force can lead to the same issues that occur when the time step is too large – atoms being pushed into each other or flying apart at great velocities, compromising the integrity of the entire simulation.

2.2. Molecular Mechanics

Molecular mechanics models chemical interactions using the laws of classical mechanics. Nuclei move according to Newton's equations of motion, and interactions between atoms are approximated by simple potentials with empirical parameters. The total energy of the system is written as a sum of various types of bonded and non-bonded interactions:

$$E(\vec{r}) = E_{\text{stretching}} + E_{\text{bending}} + E_{\text{dihedral}} + E_{\text{LJ}} + E_{\text{Coulomb}} \tag{2.1}$$

The bonded interactions are usually approximated by harmonic or Morse potentials and separated into terms for the stretching of individual bonds, as well as the bending of bond and dihedral angles. For the non-bonding terms, a Lennard-Jones potential is used for the exchange and dispersion interactions, and the Coulomb interaction between point charges on atoms. The force constants and other parameters for each of these terms are fitted to reproduce equilibrium distances, vibrational frequencies and other properties obtained from experimental or QM methods. Different bonding situations, e.g. carbon-carbon single, double and triple bonds, are described by creating parametrizations for different *atom types* of the same element. The pairwise nature of the non-bonded interactions leads to the overall quadratic scaling of the method's computational cost in the number of atoms. By introducing distance cutoffs beyond which these interactions are not evaluated, as well as other optimizations, the scaling can be improved.

However, the harmonic approximation and fixed definition of atom types limit the applicability of MM models to any systems where molecules are far from equilibrium and bonding situations change noticeably. Additionally, systems with strong polarization or electron delocalization effects are challenging for the method. While reactive and polarizable force fields have been developed to alleviate these issues, this improved description comes at an increased computational cost.

Overall, MM force fields are an invaluable tool for the simulations of large systems with limited chemical reactivity. They have become especially crucial in simulations of proteins, whose structure is determined by non-bonded interactions to a large extent.

MM force fields can be combined with QM methods in *hybrid QM/MM* schemes. Here, the bulk of a system (e.g. a large protein) is modeled using MM, while a small sub-region where an MM description would be insufficient (e.g. the substrate and reactive center of an enzyme) is instead described using a quantum-chemical method. Most of the protein serves the function of providing a specific steric and electrostatic environment, for which a classical description is sufficient. The changes in bonding from a reaction, molecular excitations from the interaction with light, or other effects where a QM description is necessary are frequently limited to not much more than a hundred atoms within the protein consisting of hundreds of thousands of atoms. The interaction between the QM and MM regions can be modeled in several ways, but most frequently, electrostatic embedding is used. Here, the MM region's point charges provide an additional external potential for the QM method, but the charges within the QM region do not affect the structure of the MM part (to avoid having to iterate to self-consistency). These hybrid QM/MM schemes have been used successfully for simulations of biological systems[40] and organic semiconductors[41].

2.3. Quantum Mechanical Methods

Several experimental results in the early 20^{th} century (such as Planck's law and the photoelectric effect) could only be explained by the realization that Newtonian mechanics is insufficient to describe behavior at microscopic scales. The discrete quantization of properties previously assumed continuous and the apparent wave-particle duality of matter required a new formalism for mechanics. At the core of quantum mechanics lie a few postulates: First, any quantum-mechanical system is described by a continuous wave function Ψ. The absolute square of the wave function is a probability density and can be used to obtain the probability of finding the system in a given state. Observable properties of the system can be calculated from the wave function using higher-order functions called operators. The Schrödinger equation must be solved to obtain the energy of the system using the Hamiltonian operator $\hat{H}$. For an isolated system, the Schrödinger equation is time-independent, corresponding to a standing wave:

$$\hat{H}\Psi = E\Psi \tag{2.2}$$

In general, many wave functions Ψ_i with their respective energies E_i satisfy this equation. The solution wave functions are eigenstates of the Hamiltonian, and can be stated in terms of basis functions referred to as *orbitals*. The Hamiltonian H contains all contributions to the total energy of the system. Usually, these are the kinetic energies T of all particles (electrons and nuclei) as well as potential energy terms V from the attraction of electrons to nuclei and the repulsion of identically-charged particles. The correlated interactions between particles greatly complicate solving the Schrödinger equation, and are usually abstracted away as much as possible. Due to the large mass difference between electrons and nuclei, the motions of the nuclei occur on a longer timescale than electronic motions.

In the Born-Oppenheimer approximation, this difference is used to decouple electronic and nuclear motions by formulating the Schrödinger equation for the electronic degrees of freedom only and solving it for a fixed nuclear geometry. The resulting wave functions are always eigenfunctions of the electronic Hamiltonian and called *adiabatic* states. The Born-Oppenheimer approximation is very powerful as it allows us to obtain potential energy surfaces from quantum mechanics, but is not compatible with spontaneous changes in electronic states as they occur when occupations change. It also does not help with the problem of electron-electron interactions, so the Schrödinger equation can only be solved for one-electron systems.

Regardless of whether the Born-Oppenheimer approximation is used, an initial guess for the shape of the wave function is necessary to solve the Schrödinger equation. Thanks to the variational principle, the quality of a candidate wave function can easily be quantified: The lower the energy corresponding to a wave function for a given system, the closer that wave function is to the true wave function of the system. In all but the smallest of systems, the wave function is expressed as a linear combination of several simple wave functions (e.g. plane waves or orbitals of the Hydrogen atom), and the weight coefficients are optimized to get the solution of minimal energy. The set of wave functions which are thus combined is referred to as the *basis set* for the calculation. One frequently used approach to construct molecular orbitals is to use orbitals centered on its constituent atoms, commonly called the *linear combination of atomic orbitals* (LCAO) approach.

There are several pathways to obtain solutions for many-electron wave functions which treat the electronic correlation terms differently, such as the Hartree-Fock Self-consistent field method, Density Functional Theory, Coupled Cluster or Configuration Interaction methods. Among these, only DFT is relevant for the present work and is described in the following section.

2.3.1. Density Functional Theory

When solving the Schrödinger equation as described above, the resulting wave function for an N-electron system depends on the three spatial coordinates of each electron. This high dimensionality leads to large computational costs in the evaluation of high-dimensional integrals for electronic interactions. Additionally, the wave function is not a physical observable itself, which is an obstacle to the overall interpretability of the results.

In 1964, Hohenberg and Kohn[42] showed that the Hamiltonian and the ground-state wave function are both fully determined by the electron density ρ and established that the variational principle holds when minimizing the ground-state density. The electron density which corresponds to a given wave function ϕ can be calculated as

$$\rho = \|\phi\|^2 \tag{2.3}$$

The energy of the system can then be stated as a functional of the electron density, resulting in the name Density Functional Theory (DFT) for the method. Throughout the following years, further development of the formalism cemented DFT as an essential tool in the quantum chemistry toolbox. Kohn-Sham-DFT[43] greatly expanded the applicability of the method by evaluating the energy for an auxiliary non-interacting system of N electrons

and adding correction terms to describe the interactions. The DFT energy functional then reads

$$E[\rho] = T_{ni}[\rho] + V_{ne}[\rho] + V_{ee}[\rho] + \underbrace{\Delta T[\rho] + \Delta V[\rho]}_{E_{xc}[\rho]}. \tag{2.4}$$

T_{ni} is the kinetic energy of the non-interacting system, V_{ne} the electron-nuclear interaction and V_{ee} the classical electron-electron repulsion. ΔT and ΔV are the corrections to the kinetic and potential energy terms resulting from the (non-classical) electronic interactions, and are usually combined in a single term called exchange-correlation energy E_{xc}. For all terms except E_{xc} exact analytical solutions without need for further approximations are known. The orbitals χ_i corresponding to the non-interacting system then can be obtained from the following set of eigenvalue equations:

$$\left(-\frac{1}{2}\nabla^2 + V_{\text{eff}}(\vec{r})\right)\chi_i(\vec{r}) = \epsilon_i \chi_i(\vec{r}) \tag{2.5}$$

The effective external potential V_{eff} is the sum of the potential created by the nuclei, the classical Coulomb interaction between every electron and the electrostatic field of all other electrons, and V_{xc}, the functional derivative of E_{xc} w.r.t. the density, for which no exact functional form is known:

$$V_{\text{eff}}(\vec{r}) = -V_{\text{nuc}} + \int \frac{\rho(\vec{r})}{\|\vec{r_i} - \vec{r'}\|}\mathrm{d}\vec{r'} + V_{xc} \tag{2.6}$$

The electron density $\rho(\vec{r})$ can then be calculated from these orbitals as

$$\rho(\vec{r}) = \sum_i^{N_e} \|\chi_i(\vec{r})\|^2. \tag{2.7}$$

As the density is also needed for the calculation of the effective potential, an initial guess for the density must be made, from which the orbitals are initially calculated as a linear combination of a given basis set. From the orbitals, a new value for the density can be calculated, and this process is iteratively repeated to achieve self-consistency. The quality of the initial guess for the density can impact the time needed until convergence. One method to construct the initial density is the Harris-Functional approach[44, 45], wherein the density is constructed from several fragment densities in a manner similar to the LCAO ansatz.

All calculations in DFT are exact and analytically solvable, except for the exchange-correlation functional for which the correct functional form is unknown. Therefore, the main factor determining the quality and computational costs of any DFT calculations is the approximation employed to calculate E_{xc}. All such approximations employ some empirical reasoning to obtain the functional form and/or parameters, and as such, most DFT variants in use today are semiempirical in nature.

The simplest approach to the exchange-correlation functional is the Local Density Approximation (LDA). Here, it is assumed that the value for V_{xc} at a given point in space depends only on the value of the density at that point. This approximation usually uses

results from the uniform electron gas to derive terms for the exchange contribution and approximate the correlation. Using the LDA functional, reasonable molecular geometries can be obtained, but the method fails to predict bond energies and reaction barriers due to a severe overestimation of bond strengths.

Changing the exchange-correlation functional to include the gradients of the density results in the class of Generalized Gradient Approach (GGA) functionals, which perform far better for molecular systems. However, GGA functionals are not able to fully describe London dispersion interactions, hydrogen-bonded systems, or intermolecular complexes held together by charge transfer interactions. Hybrid functionals such as the popular B3LYP functional introduce the Hartree-Fock exchange terms as contributions to the total exchange correlation functional in an attempt to improve this behavior. The relative weights of the HF and LDA or GGA contributions are empirically adjusted, and due to error cancellation effects, hybrid functionals can give very good results for many systems.

Most systems where GGA-based functionals fail include long-range effects, which are insufficiently described by the local properties of the density. One approach to mitigate this is to separate the electron repulsion term in two parts depending on the distance between the electrons. Then, the full HF exchange can be used only in the short range, and its contribution smoothly decreases with the distance in favor of the DFT exchange functional. These long-range corrected DFT functionals (LC-DFT, the term 'range-separated' has also been used) give very good results for polarizabilities, van der Waals interactions and are especially well suited for the description of molecular excitations.

Overall, DFT generally converges faster and scales better than wave-function based methods[38], and is very flexible in terms of basis sets. The quality and computational costs of DFT calculations greatly depend on the chosen exchange-correlation functionals.

2.3.2. Density Functional Tight Binding

Starting from the DFT formalism, it is possible to further reduce computational costs by adding approximations until a truly semi-empirical theory is obtained[1]. Density Functional Tight Binding (DFTB)[1, 46, 2, 3, 47, 48] was developed from that core idea, and has entered widespread use especially for large (biochemical) systems. Since the original method was published, several expansions and improvements to the formalism have been made, resulting in several distinct versions of DFTB. The `dftb+` program package[4, 5] has become akin to a reference implementation of the method and is freely available and actively developed.

DFTB Basics

Starting from Kohn-Sham DFT, several approximation steps are necessary to arrive at DFTB. The initial guess for the density is constructed from the fragment densities on the individual atoms as calculated from a minimal basis set of Slater-type orbitals. The orbitals and densities are radially compressed using a harmonic potential and an empirically determined compression radius, an approach similar to the tight-binding method used in solid state physics.

[1] Some DFT functionals have some semi-empirical properties due to empirically adjusted parameters.

The locality introduced by the radial compression can then be used for two crucial approximations: First, only the two-center terms of the Hamiltonian are evaluated. Second, the matrix elements for the Hamilton and overlap matrices for the orbitals of a given pair of atoms can be assumed to depend only on the atoms' chemical elements and the interatomic distance. Using a minimal basis of atomic orbitals to expand the Kohn-Sham orbitals $\Psi_i = \sum_\mu c_\mu^i \phi_\mu$ and a reference density ρ_0, the DFTB matrix elements $\hat{H}^0_{\mu\nu}$ can be calculated as

$$\hat{H}^0_{\mu\nu} = \langle \phi_\mu | \, \hat{H}(\rho_0) \, | \phi_\nu \rangle \tag{2.8}$$

These approximations allow to calculate all matrix elements for each pair of two elements across a wide range of interatomic distances once using DFT and tabulate the results, using them as parameters for later calculations, an approach introduced by Slater and Koster [49].

With this constructed density in hand, the Kohn-Sham equations are solved using a simple GGA functional. No iteration to self-consistency is performed, as the initial density is assumed to be sufficiently accurate by construction from the precalculated DFT matrix elements. This approximation, in addition to avoiding the computation of integrals during run-time, results in an immense reduction of the computational cost of DFTB compared to DFT.

Unfortunately, these approximations result in energy eigenvalues ϵ_i very far off from the reference or experimental values. One large factor contributing to the error is the compression of the densities and orbitals, which greatly reduces the electronic repulsion. This is alleviated by adding a repulsive energy E_{rep} to the electronic energy obtained from the KS equations. The repulsive energy is constructed from atom-pair repulsive potentials V_{rep} by shifting bond energies to match atomization energies obtained from experiment or higher-level reference methods. V_{rep} is a parametrized function of the interatomic distance of two atoms with given elements, whose parameters are optimized so that the DFTB total energy matches the reference as closely as possible. Due to this fit procedure, the repulsive potential must correct not just for the compression, but also for all other errors such as double-counting terms. This complicates the fit for the repulsive potentials, making it the most involved step of the DFTB parametrization. Original publications constructed V_{rep} as a sum of exponentials decaying with distance, but splines of polynomials have since become standard. The difficulty in fitting the repulsive potentials has resulted in several DFTB parameter sets being developed for different applications, of which the MIO parameter set is especially suitable for organic molecules and geometries.

These approximations give the non-self-consistent DFTB method (further referred to as DFTB1). DFTB1 is faster than DFT by two to three orders of magnitude, as the only costly step of the calculation is the matrix diagonalization necessary to calculate the Kohn-Sham orbitals. However, the harsh approximations taken in its construction result in issues for charged systems and systems with charge-transfer effects, as well as a systematic overestimation of bond strengths compared to steric effects due to the limitations placed on the repulsive potential.

Self-Consistent Charge (SCC) DFTB

For any atom pair with significant differences in the electronegativities of the involved atoms, charge is transferred from one atom to the other, and DFTB1 does not give good results due to the harshness of the underlying approximations. The DFTB formalism can be extended so that these effects are described more accurately. To this end, the energy functional is expanded into a Taylor series around a reference density ρ_0 and cut off after the second order.

$$E[\rho] = E^0[\rho_0] + E^1[\rho_0, \delta\rho] + E^2\left[\rho_0, (\delta\rho)^2\right] \tag{2.9}$$

The first two terms of the expansion are the terms included in the original non-self-consistent DFTB formalism, and the density fluctuations $\delta\rho$ are only used in the second-order term. The density fluctuations are usually constructed as a linear combination of spherically symmetric atomic contributions $\Delta\rho$, although DFTB variants using di- and multipolar contributions have also been developed. A functional form for the second-order energy can be derived by analyzing its behavior for very large and very short interatomic distances R_{ab} and interpolating between the two cases using a suitable function abbreviated here as γ_{ab}. For very short distances, γ_{ab} describes the electron-electron interaction at one atom, which is related to the atom's chemical hardness and parametrized in the Hubbard-parameter U_a obtained from DFT calculations. For $R_{ab} \to \infty$, γ_{ab} should approach $\frac{1}{R_{ab}}$. Therefore, the full second-order energy is calculated as:

$$E^2 = \frac{1}{2}\sum_{a,b}\Delta q_a \Delta q_b \gamma_{ab}. \tag{2.10}$$

The atomic charge fluctuations Δq determine the density fluctuations, but themselves depend on the molecular orbital coefficients, so the second-order equations must be solved by iteration to self-consistency.

The expansion to the second order improves the accuracy of geometries, reaction energies and vibrational frequencies. However, in cases where the charge fluctuations are large (e.g. in strongly polarized bonds and especially hydrogen bonds), this expansion is insufficient. To further improve the description of these systems, the method was extended by also including the third order of the energy expansion. The resulting method, known as DFTB3, gives very good results for a wide variety of systems, but is not used in this work and therefore not explained in detail.

Long-Range Corrected Time-Dependent DFTB

It is possible to formulate DFTB as a time-dependent theory[50, 3]. This allows using DFTB to calculate the non-adiabatic dynamics of molecular excitations at far lower computational costs than time-dependent variants of DFT would incur. In large systems and excited states, the local XC functionals (GGA) used to derive the DFTB formalism do not give very good results. Therefore, a long-range corrected version of the DFTB formalism was developed by Niehaus and Della Sala[47], and combined with the TD-DFTB formalism by Kranz et al.[51]. This LC-TD-DFTB method has since been shown to perform well for the description of photochemical properties in organochemical [51, 52] and biochemical [53] systems.

2.4. Simulating Charge and Exciton Transfer

Formulating a model for the description of moving charges or excitations is challenging. Quantum-mechanically, a charge transfer event changes the electronic state of a molecule. The PES corresponding to the different adiabatic states of the molecule are close in energy and approach a crossing. Here, the electronic wave function must be written as a linear combination of the adiabatic states as the states mix, and the Born-Oppenheimer approximation is no longer applicable. Solving this problem fully quantum-mechanically is complex and computationally costly as it would also require solution of the time-dependent Schrödinger equation. This is fine for examining small systems of only a few atoms, but for most applications this limited scale is insufficient.

Charge carrier mobilities in materials are an interesting property for semiconductor design and the design process can be greatly accelerated if the mobilities could be obtained from computer simulations. The formalism can be derived for both a hole and an electron as charge carrier. As I only use hole transfer in this work, all examples will be formulated from that perspective. The mobility μ is defined as the speed at which the charge carrier moves through the material, calculated from the Einstein-Smoluchowski [54] equation as

$$\mu = \frac{eD}{k_\mathrm{B}T} \tag{2.11}$$

where e is the elementary charge and D the diffusion coefficient of the charge carrier in the material calculated as

$$D = \frac{1}{2n} \lim_{t\to\infty} \frac{\mathrm{dMSD(t)}}{\mathrm{t}}. \tag{2.12}$$

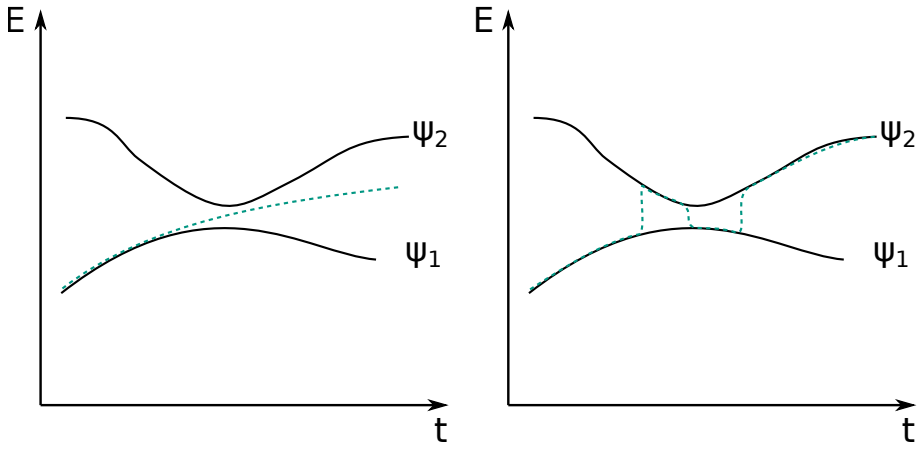
n here denotes the number of dimensions available for the movement – e.g. 1 for a chain of molecules, 3 for a crystal where the charge can propagate in any direction. The mean square displacement (MSD) of the charge carrier is the distance the charge carrier travels in a given time on average across N_traj independent simulations:

$$\mathrm{MSD}(t) = \frac{1}{N_\mathrm{traj}} \sum_{l}^{N_\mathrm{traj}} \sum_{A} (x_A(t)^l - x_0^l)^2 P_A^l(t) \ . \tag{2.13}$$

Here, $x_A(t)^l$ is the center of mass of a molecule A, $P_A(t)^l$ is the diabatic population of molecule A, and x_0^l is the center of charge at $t = 0$ along the trajectory l. In order to obtain a good estimate of the mobility, the charge carriers must be able to move through a large system for a very long time, and a single simulation is insufficient. Therefore, a very efficient method for determining the location of the charge carrier is desirable.

Instead of solving the complex non-adiabatic problem, the movement of the charge can be estimated using other models. In these semi-classical approaches, the nuclei are propagated classically, and only the adiabatic states of the electrons are considered. Using a given starting occupation and properties calculated for the adiabatic states, so-called *propagator* algorithms can output a new occupation. For most such algorithms, two quantities are crucial: the energies ϵ_i and ϵ_j of the individual states i, j and the couplings T_{ij} between them:

$$T_{ij} = \langle \phi_i | \, \hat{H} \, | \phi_j \rangle \tag{2.14}$$



(a) MFE (b) SH

Figure 2.1.: Evolution of a system with two adiabatic states (black) along a simulation. The actual state of the system is shown as a green dashed line. a) Using the mean-field Ehrenfest method for propagation, the system begins in a well-defined state and after the avoided crossing is in an averaged state. b) With surface hopping, the system changes between the two adiabatic states and is in a well-defined state after the avoided crossing.

One frequently used propagator is the mean-field Ehrenfest (MFE) method[55, 56, 57], where the system evolves in an effective potential calculated as the average of the adiabatic states weighted by their respective populations. Figure 2.1a shows a brief sketch of a two-state system which starts in one of the adiabatic states. After the avoided crossing, the energy is a mix of the energies of the two adiabatic states. This approach can give good results for electron mobilities, but has several important disadvantages: First, once the system has entered the region of strong non-adiabatic couplings, the system will remain in a mixed state, leading to overdelocalization. Second, the MFE method violates microscopic reversibility, i.e. the equations of motion are not symmetric with respect to the inversion of the time axis.

Another approach is the class of surface-hopping methods, e.g. the Landau-Zener [58, 59] or Fewest-Switches Surface Hopping (FSSH) [60] models. Here, the adiabatic states are not mixed, and the system instead 'hops' between the available states as the simulation progresses as can be seen in Figure 2.1b. The probability of the system to hop from one state to another scales with the non-adiabatic coupling between the states, but the details depend on the specific scheme. In all surface-hopping methods, conservation of energy must be explicitly ensured by adjusting the nuclear velocities to compensate for the change in potential energy whenever a hop takes place. This can be achieved by rescaling the velocities of the nuclei in the system to change the overall kinetic energy. If the kinetic energy of the system is insufficient to compensate for the change in potential energy, the hop cannot occur (it is 'frustrated'). The simplest method to achieve this is to distribute the change in kinetic energy isotropically across all velocities of all atoms[61]. However,

better results can be obtained[62] by rescaling velocities along the non-adiabatic coupling vectors (NACVs) $\mathbf{d}_{I,ij}$ for each atom I and every pair of states i, j:

$$\mathbf{d}_{I,ij} = \langle \phi_i | \nabla_I | \phi_j \rangle . \tag{2.15}$$

The NACVs can also be used in the case of frustrated hops, where it has been shown that the velocities along the NACVs should be reversed whenever a hop is rejected. If the NACVs are not available, the velocities can also be rescaled according to the Boltzmann distribution, resulting in the BC-FSSH method. A detailed description and evaluation of the different propagation algorithms is outside the scope of this work, but the interested reader is referred to [63, 64, 65].

2.4.1. Hybrid QM/MM Simulations for Charge Transfer

Considering the fact that we need long simulations, large systems and an entire swarm of trajectories, how can we obtain the energies of the states, the couplings and NACVs as efficiently as possible? Using a fast semi-empirical method for obtaining the electronic structure of the system is certainly a good idea, but not quite sufficient by itself, as the N^3 scaling with system size is problematic.

Two approximations make these calculations feasible: First, assume that in the case of loose coupling (e.g. in molecular organic semiconductors held together by van der Waals interactions), the relevant states are each localized on a single molecular fragment, and the orbital structure of each molecule is not influenced by its neighbors. This allows us to calculate the orbital structure of each fragment in an independent QM calculation and work with the diabatic states, which reduces the impact of the bad QM scaling to a manageable level and is trivially parallelizable. Second, assume that only the frontier orbitals are relevant for the charge transfer events, and that the orbital structures of the charged and uncharged fragments is essential except for that frontier orbital[36]. Thus, for the transfer of a single electron, the energy of the charged diabatic state (called site energy) is simply the energy of the neutral molecule E_0 plus the LUMO energy, while for the hole transfer, it is $E_0 - E_{\mathrm{HOMO}}$. The couplings are then calculated from the frontier orbitals of the different fragments as shown above in Equation 2.14. Using DFTB, the overlaps of the frontier MOs are quite cheap to calculate without the need to evaluate any integrals, and the resulting couplings excellently match those obtained from high-level reference calculations[66, 67] when corrected by a consistent, empirically determined scaling factor.

Computational cost can be further reduced by restricting the movement of the charge to a region of the bulk system in question, e.g. a one-dimensional chain of molecules within a larger crystal, and calculating all dynamics outside this region using an MM force-field. This reduces the number of fragments for which the expensive QM calculation must be performed. The effect of the 'environment' (in this case the bulk of the material) on the charge-carrier fragments and of the moving charge on the environment can be simulated by electrostatically embedding the QM region in the MM system. However, the former only has a notable effect in biological systems, so it can be omitted e.g. in organic semiconductors. This technique was originally developed for hybrid QM/MM simulations of large biochemical systems and later applied to charge transfer simulations in DNA[68, 35, 69] and later organic materials [70].

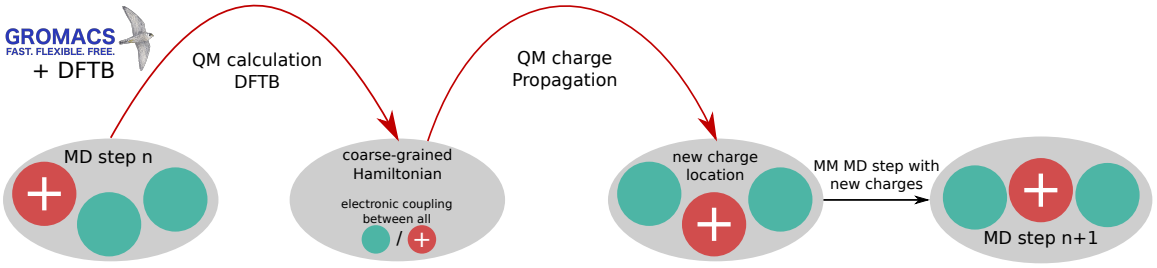


Figure 2.2.: Electron transfer simulation schematic. The system consists of the MM part (gray ellipses) and the individual sites treated quantum mechanically (cyan and red circles). In MD step n, the charge is located on the leftmost site (red circle). Using DFTB, the charge is propagated to the middle site, and finally the geometry is relaxed classically in response to the new charge distribution.

The hole wave function is then calculated as a linear combination of molecular orbitals ($|\phi_m\rangle$) of fragment molecules A,

$$\Psi = \sum_A \sum_{i \in A} a_i \, |\phi_i\rangle \ , \tag{2.16}$$

The elements of the Hamiltonian for propagation are then the orbital (site) energies (diagonal elements H_{ii}) and electronic couplings (off-diagonal elements H_{ij})[71, 68].

$$H^0_{ij} = \langle \phi_i | \, H[\rho_0] \, | \phi_j \rangle \tag{2.17}$$

From these calculations, the site energies and couplings are obtained and collated in the coarse-grained Hamiltonian required by the propagation methods to calculate changes in population along the trajectory.

A rough schematic of the steps necessary for a charge transfer simulation is shown in Figure 2.2: First, the geometry of the sites in the QM zone is passed to DFTB to calculate fragment orbitals on each site, as well as their pairwise couplings. This Hamiltonian is then passed to a propagator algorithm (e.g. the MFE method), which calculates transition probabilities based on the Hamiltonian and propagates the charge accordingly. The updated charge distribution is then passed back to the core MD code, where the system responds classically to the new charge distribution. If the forces in the occupied state are available, they are added to the MM forces acting on occupied fragments. This process is repeated for every step of the molecular dynamics simulation.

2.4.2. Implicit Relaxation in Charge Transfer Simulations

A crucial aspect of the simulation is how the change in occupation of a fragment affects the positions of its constituent atoms. Usually, the minimum of the PES in the charged state is not at the same atomic coordinates as that of the uncharged state. Thus, when the occupation changes, forces resulting from its electronic state should be applied to the fragment in addition to forces resulting from the MM description of the fragment and any electrostatic interaction with the environment. The fragment then relaxes, resulting in a

geometry distinct from that of the uncharged state and a lowered site energy. Within the approximations described above, these forces can be obtained from the gradients of the relevant frontier orbital only, as the electronic structure of the charged and uncharged species is assumed to be otherwise identical.

However, this gradient evaluation is costly, motivating another approximation: Instead of *explicitly* relaxing the fragment geometry using QM forces, the fragment can be *implicitly* relaxed. The occupation-dependent forces are omitted, and the site energy is adjusted by an empirically determined value λ called 'relaxation energy', weighted by the wave function coefficient c_i:

$$H_{ii}^{'} = H_{ii} - |c_i|^2 \lambda, \tag{2.18}$$

A value for λ can be obtained from more accurate quantum-chemical methods. This approach, which I will refer to as *implicit relaxation* saves some computational effort at the cost of completely eliminating any changes in the geometry of a fragment as its occupation changes except for simple electrostatic effects. Nevertheless, models using this approximation can still give good results[36] depending on the specifics of the system.

2.4.3. Adaptation of the Formalism to Exciton Transfer

The formalism described above can also be adapted to work for the transfer of excitons. An exciton is a quasiparticle, representing an excitation of molecular electronic states which can be passed on from molecule to molecule. Exciton transfer occurs in organic light-emitting diodes (OLEDs), solar cells, as well as proteins involved in e.g. photosynthesis or vision. The exciton is a combination of an electron and a hole resulting from an excitation of the electron from an orbital it occupies in the ground state to a higher-lying orbital. The electron and hole can be located at the same molecule, at molecules close to each other in proximity or be completely separated from each other. Which of these states is the case is determined by the Coulomb interaction between the positive charge of the hole and the negative charge of the exciton, as well as the dielectric constant of the material.

When electron and hole are tightly bound and the exciton is small, we speak of Frenkel excitons [72, 73], which frequently form in molecular organic semiconductors. In Wannier-Mott excitons, in contrast, electron and hole are loosely bound and can easily be separated, leading to a large effective radius for the exciton. They frequently occur in inorganic semiconductors like GaAs, but are of no further relevance to this work. An intermediate case are charge-transfer (CT) excitons, where the electron and hole are located on adjacent molecules. As these excitons propagate, the excess charges are passed from one molecule to its neighbor. CT excitons frequently form in molecular organic crystals in addition to Frenkel excitons.

Simulating exciton dynamics in quantum-chemical calculations requires non-adiabatic molecular dynamics and a theory based on the time-dependent Schrödinger equation, in order to capture the dynamics of the system transitioning from one electronic state to another. Therefore, a direct MD simulation of any but the smallest molecules is prohibitively expensive without further approximations.

In order to adjust the framework introduced above for charge transfer and apply it for exciton transfer, the Hamiltonian for propagation must be constructed. For this, the excited

states of the entire system must be decomposed to local excitation on individual fragments, and the notion of a site energy and couplings between the states must be introduced. Note, that I only describe the treatment of Frenkel excitons here, as these were used for the method described in chapter 4. The formalism can easily be expanded to include CT states, which are used in chapter 6.

The Frenkel Hamiltonian is thus constructed as follows[74]:

$$H = \sum_i \epsilon_i \left|i\right\rangle \left\langle i\right| + \sum_{i \neq j} T_{ij} \left|i\right\rangle \left\langle j\right|, \tag{2.19}$$

for the excited states i and j. The site energies ϵ_i are simply calculated as the excitation energies corresponding to the given state.

There are, however, two methods to calculate the pairwise couplings between the excitations. In a symmetrical dimer, two adiabatic states are obtained from the diagonalization of the diabatic Hamiltonian (which, in the context of this work, is always obtained using LC-TD-DFTB). The splitting of these states determines the *supermolecular* excitonic coupling, and is calculated from excited state calculations of the dimer and both monomers as

$$T_{ij} = \frac{1}{2}\sqrt{\left(E_2 - E_1\right)^2 - \left(E_i - E_j\right)^2}, \tag{2.20}$$

where E_1 and E_2 are the first two excitation energies of the dimer and E_i and E_j are the first excitation energies of the monomers i and j. With this method, the coupling can become imaginary if the monomer excitation energies are unfavorable. To avoid the occurrence of imaginary couplings as well as the monomer calculations, the supermolecular coupling can be approximated as

$$T_{ij} = \frac{1}{2}\left(E_2 - E_1\right) \tag{2.21}$$

This approximation is best if both monomers have the same internal geometry and are arranged in specific manners[75]. Supermolecular couplings include exchange effects and are essential when the distance between the fragments is small. As the exchange contribution vanishes compared to the Coulomb interaction at larger distances, the costly dimer calculation can be avoided by using *Coulomb* couplings. Using an excited state calculation for each monomer, the atomic transition charges Q can be obtained and used for the evaluation of the Coulomb coupling as[37]

$$T_{ij} = \sum_{a \in i, b \in j} Q_a Q_b \gamma_{ab} \tag{2.22}$$

where γ_{ab} is the function introduced for the Coulomb interactions in the second order of the DFTB formalism as described in Equation 2.10. The calculation of the excited states can be performed using LC-TD-DFTB2, which is far more computationally costly than the calculations required in the simulations of charge transfer, for which time-independent non-SCC DFTB can be used.

The exciton diffusion constant is the main metric for the speed of the propagation of the exciton. It is calculated from the mean square displacement as described in Equation 2.12[37].

3. Machine Learning Methods

3.1. Introduction to Machine Learning

The term *machine learning* (ML) subsumes a variety of statistical methods for systematically extracting patterns from data. This pattern extraction can take several forms: On the one hand, *supervised* learning methods are used where there is a clear input-target relationship in the data, such as when the goal is to predict the energy of a molecule from its geometry or to determine whether a compound is drug-like. On the other hand are the *unsupervised* ML methods, where the goal is to find structure and patterns in large data sets without a reference for the optimal solution. Unsupervised learning can be used to analyze molecular dynamics trajectories of large biosystems and extract geometries representing distinct intermediate states. Dimensionality reduction is a form of unsupervised learning used to extract the most relevant features from high-dimensional inputs which is also an established tool in the analysis of molecular dynamics simulation. In machine learning contexts, it is frequently used to preprocess data for other ML models or determine the relevant variables. In this work, I only use supervised learning, and thus will not further discuss unsupervised ML methods. The interested reader is encouraged to consult textbooks [76] and overview articles on dimensionality reduction techniques [77, 78].

To give a very general definition, a supervised ML *model* is a parametrized function mapping inputs x to targets y, whose parameters w have been adjusted using a set of reference data points to reproduce the input-target relationship as closely as possible.

$$\text{Model} : (\mathrm{x}, \mathrm{w}) \rightarrow \mathrm{y} \tag{3.1}$$

The shape of the function determines the complexity of relationships the model is able to handle. For example, a linear regression model will not be able to accurately capture nonlinear behavior, while a neural network with millions of parameters can describe exceedingly complex dependencies.

The adjustment of the function's parameters is usually performed by minimizing some form of *cost* or *loss* function which measures the deviation between the model's prediction and the reference value for a given input. The basic functional shape of the cost is determined by statistical considerations of the distributions of the errors, such as the Gauss-Markov theorem[79], and the type of problem the model faces.

ML models can be used to solve both *regression* and *classification* problems. Regression models take one or multiple input variables (called *features*) assumed to be independent and use them to quantitatively predict the target(s), i.e. one or multiple dependent variables. Fitting a line to a set of measurements using the least-squares method is the most commonly used form of regression. In contrast, classification problems arise when the target is not a continuous variable (i.e. a number), but some form of discrete property. Identifying

specific objects within images, for example, is a classification problem encountered in the development of autonomous vehicles. While both regression and classification models find applications in computational chemistry, in this work I will discuss and use mainly regression models. The fundamentals of all ML techniques presented in this chapter are covered extensively in textbooks such as [76, 80, 81], which I used as sources for the content of this chapter.

3.1.1. Linear Regression: A Toy Example

Linear regression is a technique which has been used for centuries in scientific applications to *learn* relationships between observables, e.g. in order to determine the constants within natural laws. While most natural scientists would not intuitively describe linear regression as a machine-learning method, it does in fact share the same conceptual foundation with modern machine learning methods, namely the fitting of parameters based on empirical data. The main difference between linear regression and e.g. a neural network lies in the number of fittable parameters, which for neural networks can be millions. Therefore, training neural networks typically requires machine support, where linear regression is simple enough to be performed with pen on paper. Due to fundamental similarities between linear regression and modern machine learning methods, I will introduce the core principle and terminology using a simple linear regression example before moving on to more complex methods.

Given a set of n values $X = \{x_1, x_2, \dots x_n\}$ at which a given property Y is measured and assuming a linear dependency of Y w.r.t. X, linear regression is the process of fitting a line through all available pairs (x_i, y_i) as closely as possible.

As an example, let's take a (synthetic) data set for the photoelectric effect, measuring the kinetic energy of the electrons knocked out of the metal w.r.t. the frequency of the incoming light. Figure 3.1 shows a set of sample measurements for the kinetic energies at various frequencies above the threshold frequency[1]. These quantities evidently have a roughly linear relationship to each other.

A linear model with only one input variable only has two parameters: the slope of the line and the intercept with the y-axis. Generalized to multiple variables, the linear model can be written as

$$\hat{y}_i = b + \sum_j x_{ij} w_j \tag{3.2}$$

where b is the intercept (also called bias) and w_i are the slopes of the line with respect to the different input variables x_i. From here on, all predicted values will be designated with a caret to distinguish them from the true values (i.e. y vs $\hat{y}$). To *train* the model here simply means to determine the optimal parameters w_i (and b[2]) for a given set of training data pairs. The most frequently used way to find the optimal parameters is to minimize the

[1] The code used to generate these plots can be found in section A.1.

[2] Frequently, b is rephrased as the slope for a fictitious, constant input variable $x_b = 1$, so that all parameters can be combined into one vector which is scalarly multiplied with the input vector to obtain a prediction. This notation is especially handy for more complex models like neural networks.

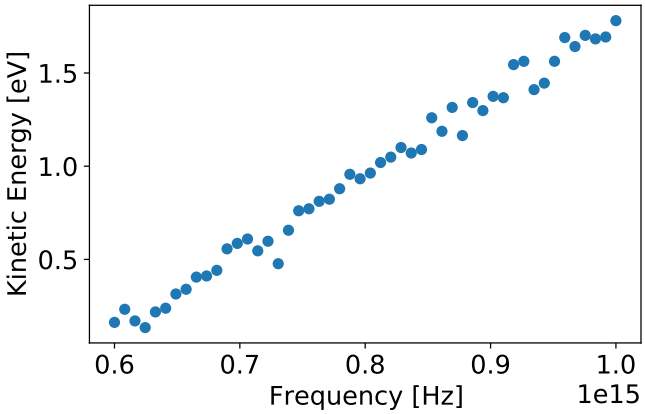


Figure 3.1.: A synthetically generated data set simulating measurements of the photoelectric effect in sodium. 50 data points were generated by adding randomized noise to the true function whose slope h[82] and intercept W(Na)[83] were obtained from reference data.

sum of the squares of the difference between each reference value y_i and each predicted value $\hat{y}_i$, for a model with only one input variable:

$$\mathrm{RSS}(w) = \sum_{i}^{n} (y_i - \hat{y}_i)^2 = \sum_{i}^{n} (y_i - (x_i w_i + b))^2 \tag{3.3}$$

The residual sum of squares of a linear regression model is a function with a well-defined minimum, for which a closed-form solution exists. In more complex models like neural networks, the minimum is no longer globally unique and must be determined numerically or using some form of gradient descent algorithm. Minimizing the sum of squares for this specific data set results in the fit shown in Figure 3.2.

The residual sum of squares for this fitted model is around 0.1 eV^2, but this metric is not very intuitive to understand. Instead, there are other quality metrics which are frequently used to quantify the quality of fitted models: Very often, the mean absolute error (MAE) is calculated, which describes the (euclidean) distance of each point from the fitted line:

$$\mathrm{MAE} = \frac{1}{n} \sum_{i}^{n} \mid y_i - \hat{y}_i \mid \tag{3.4}$$

However, the MAE is not a relative error measure, and comparing models across data sets using the MAE can be problematic. An MAE of 0.5 may be great for a model for which the reference values Y are on the order of magnitude of 100, but if they are on the order of 0.1, a model with that MAE would be useless. To allow comparison of model quality independently of the magnitude of the target values, the MAE can be normalized to give a mean relative error (MRE):

$$\mathrm{MRE} = \frac{1}{N} \sum_{i}^{N} \frac{|y_i - \hat{y}_i|}{\bar{y}} \tag{3.5}$$

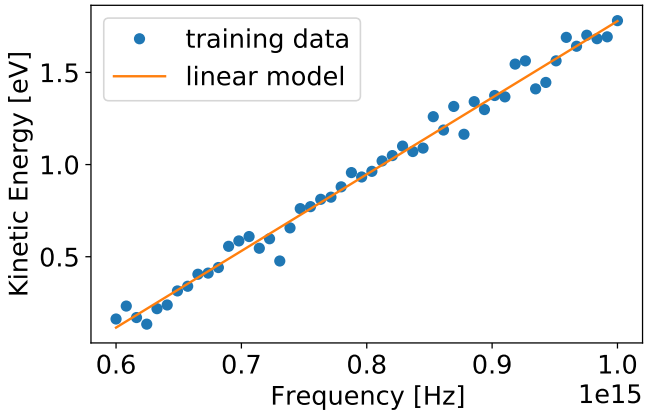


Figure 3.2.: A linear model (orange line) for the sodium photoelectric effect data set. The fitted parameters of the model correspond to the quantities in the equation $E_{kin} = hf - W$: the slope of the model is 4.13×10^{-15} eV s ($\approx h$), the intercept is -2.33 eV ($\approx W(\mathrm{Na})$)

The MRE, however is not frequently used, as most formulations encounter numerical instabilities, e.g. when the mean of the reference values $\bar{y}$ approaches 0. The coefficient of determination (also referred to as R^2) is the most frequently used magnitude-independent quality metric for empirically fitted models. It is calculated as

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}, \tag{3.6}$$

where $\bar{y}$ is again the mean of the target values. Intuitively, the coefficient of determination compares the errors made by the fitted model (numerator) with the error of the *null* model which only ever predicts the mean of the reference data distribution. A perfectly fitted model has $R^2 = 1$, the null model has $R^2 = 0$ and R^2 can be arbitrarily negative for an arbitrarily bad model.

The fitted model can now be used to *predict* the target values for new x_j that it has not seen during training by interpolating between the training data points using Equation 3.2. Note, however, that extrapolating beyond the bounds of the area sampled by the training data using the model can be problematic. In our photoelectric effect example, extrapolating the trained linear model to lower frequencies leads to an area where the kinetic energy of the expelled electrons is predicted to be negative, while when running the experiment, no electrons are expelled at all when the threshold energy is reached. As this is a very simple system, it is easy to see that the model's output in that region of input space is not reasonable at all, but as input spaces and data sets grow larger and both problems and models get more complex, determining when an ML model is extrapolating can become very challenging.

An essential step in evaluating a machine learning model is to evaluate how well it is able to deal with data it has not been trained on. If a randomly chosen subset of the data set is 'held out', i.e. not given to the model to train on, the model can then be evaluated

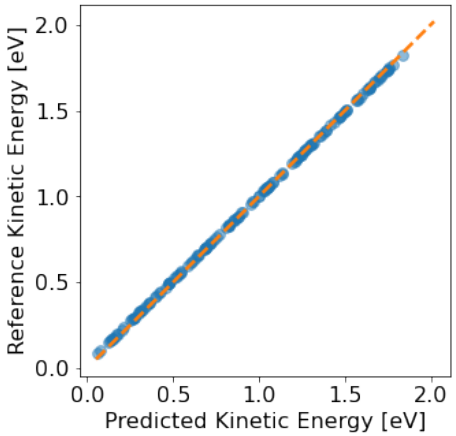


Figure 3.3.: Visualization of the evaluation of the linear regression example model by plotting predictions vs references. For a perfect model, all predicted points are on the orange line representing $\hat{y} = y$

by comparing the predictions it makes for this *test* data to the reference value. The same evaluation metrics can be used as discussed above to get a quantitative estimate of whether the model has learned the properties of the data set well.

Testing the fitted example model on 150 data points it has not previously seen, we see that it has achieved an R^2 score of over 0.99 and a mean absolute error of approximately 0.1 meV on the test set. The quality of the model's predictions can be visualized by plotting the predicted values vs the reference values like in Figure 3.3. In such a plot, a model which perfectly reproduces the reference has a slope of 1 and an intercept of 0, and is represented by the orange line. Each blue dot is one test data point, and they all closely follow the diagonal showing how well our model has learned. This type of plot can uncover interesting properties of the model, e.g. the frequency and intensity of outliers, or whether the targets in one interval are learned better than those in another. They are therefore a valuable tool for the evaluation of ML models and frequently used throughout this work.

Obviously, this kind of testing does not overcome the fundamental limitation resulting from the limited data set size – if the test data set does not contain any values below the threshold frequency at all, the model performance on these obviously cannot be tested. However, it is still an important step, as it allows checking whether the model is performing badly in its interpolation (e.g. because there is a part of the input space which is undersampled in the training data). This can be especially problematic in more flexible systems, such as neural network or kernel ridge regression models, whose behavior between data points can be chaotic (subsection 3.1.3).

The data set is therefore crucial to the success of the model and must cover all regions of input space in which the model will be asked to make predictions. Some techniques exist to detect when model predictions are not reliable and are discussed below (see section 3.7), but that does not mitigate the importance of a good data set and a robust evaluation routine for trained models.

One final thing to note for the evaluation of ML models is that in general, a model's prediction error on the test set should decrease as training set size grows. The reasoning for

this metric is that the larger the training data set, the easier it is for the model to interpolate between known data points, and the lower the prediction error. Of course, this trend is not linear – as the training set size grows, the model converges to a characteristic value for the error, which describes the maximum amount of complexity in the reference data that the model can capture. This convergence is visualized in *learning curves*: doubly logarithmic plots of prediction errors vs training set size, where models whose hyperparameters are well-optimized for every training set size usually form a line. If a model cannot be shown to improve its prediction error with added data, it is effectively already converged and the limiting factor to prediction accuracy is not the size of the training set, but the flexibility of the model.

3.1.2. Interpretability of ML Models

I want to note an important distinction here about the nature of regression models used in different contexts and the intent behind their application. In the example above, the slope parameter of the linear model corresponds to a fundamental and universal physical constant – Planck's constant, h, which reappears in many other contexts. The intercept is a material parameter, the work function of the metal determining the amount of work needed to remove an electron from the surface, which is again not specific to this experiment. Both model parameters used here are therefore assigned a direct and mostly intuitive physical meaning, and are fundamental enough to be transferable to other data sets, experiments and physical laws. The reason why the parameters have this connection, though, is that in the context of our mathematical description of the involved processes, there is indeed a linear relationship between the involved quantities[3].

Machine learning models, however, are usually employed as universal approximators when we either do not know the explicit functional form connecting the input and target variables (e.g. in image recognition, where it's very difficult to define what patterns fully determine and constitute e.g. a cat) or when we do know, but want to avoid using it for some reason, usually because it is too computationally costly, a frequent motivation for the application of ML in computational chemistry. In order to fulfill the role of efficient universal approximators, ML models are usually composed of simple, easy to evaluate functions with sufficient parameters to adapt to arbitrary dependencies between input and target variables. This becomes especially relevant when the relationship between inputs and targets is not linear. The number of parameters an ML model needs to be fitted to data corresponding to input-output pairs of a nonlinear function is frequently far larger than the number of parameters the function itself needs. Therefore, the parameters of an ML model may not have any intuitive connection to the parameters of the function it is fitted to reproduce.

This does not mean that no insights about the underlying problem can be gleaned from the structure and parameters of an ML model fitted to solve it. The structure of neural networks currently used for image recognition, for example, consists of many individual components, whose contribution to the final prediction can be separately investigated.

[3] When Einstein formulated his theoretical explanation, this was not the case – this model was therefore crucial for the development of quantum mechanics as it is today.

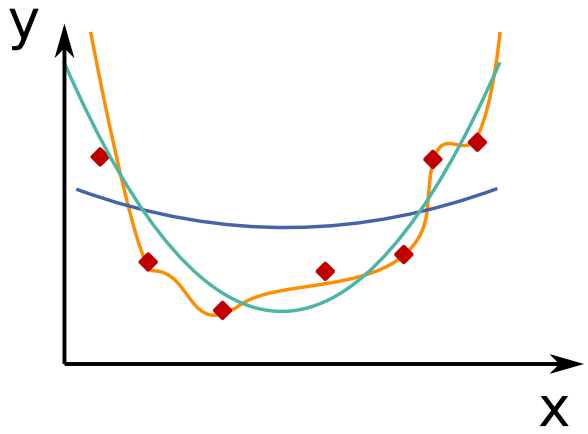


Figure 3.4.: Schematic representation of a generic data set (red dots) and three different models (blue, orange and green lines).

Thus, parts of the network can be identified as recognizing patterns which bear some relation to what we would use to describe the objects in the image (e.g. pointy ears on a cat, a dot pattern on a dalmatian)[84, 85]. While this form of decomposition is not possible with all ML models, further information about which patterns the model uses to solve the problem can almost always be extracted.

As a final remark, interpretability of ML models is less important when they are used to approximate a known, but computationally costly function. Here, the basis for the model's decisions is not important to better understand the problem at hand or to explain the decision-making process. In contrast to assigning some interpretation to the learned parameters, understanding the mathematical processes in the model can be crucial to ML models used in chemistry, as some applications require specific mathematical properties of the model (e.g. an ML model used to calculate energies and forces during an MD simulation must be continuous and smooth).

3.1.3. Overfitting and Underfitting

Figure 3.4 shows three models (represented by the blue, orange and green lines) fitted to a set of data points (red). Intuitively, the blue model is not a good model: It has a large RSS, has mostly learned the mean of the values for y and does not even come close to reproduce the input–target relationship. Models like this which do not reproduce the variance of the target values with changing inputs are called *underfit*. Generally underfitting can be a sign that the assumed correlation between input and target is not valid or not strong. For example, a linear model which attempts to learn a non-linear function would always severely underfit. Underfitting may also be caused due to inconsistencies in the training data (e.g. due to statistical noise), or may be a result of poorly-chosen hyperparameters. It also occurs if the training set size is too small to fully represent the nuances relevant for the problem at hand, so the model can only capture the general trend of the data. In models which learn iteratively, such as neural networks, an underfit model may also be a sign that the training was aborted too early.

Compared to the orange one, the green model also looks like it could be underfit – the orange model captures far more of the nuances of the training set. However, this is not clear-cut: If the training data has been determined through some empirical process or has a certain numerical uncertainty, the fluctuations which the orange model has learned can be just statistical noise, and may result in larger errors when the model is asked to predict the value for previously unseen data. A model which has learned too much of the idiosyncrasies of its training data set is called *overfit*. Symptoms of overfitting are excellent metrics on the training set, but notably worse performance on data outside it. Whether the orange model is overfit or not can therefore only be decided by not training a model on the complete data set, instead creating a 'hold-out' or test set and using this set to evaluate the model's performance after training. If multiple models (e.g. with different hyperparameters) are to be compared against each other, it is a good idea to first evaluate each model on its own subset of the data (*validation* set) and then evaluate the best model on a final test set[4]. This reduces the probability that models are overfit on the test set.

If number of overall available data points is limited, *cross-validation* can make sure that the performance metrics are stable and independent of the specific subset of data points the model saw during training. In k-fold cross validation, the data is partitioned in k subsets. k models are then trained on different combinations of $k-1$ subsets and evaluated on the remaining subset. Five or ten folds are usually sufficient to get robust results for the metrics.

Other techniques to mitigate overfitting are reducing the model complexity (i.e. the number of fittable parameters) or increasing the amount of training data, or finding a way to 'push' the model to more universal parameters. The latter technique is called *regularization*, and can take different forms depending on the specific model architecture. I will therefore discuss regularization as it is used in KRR and NN models in the respective sections.

3.2. Machine Learning Data Sets: Requirements, Preparation and Preprocessing

The available data is the foundation on which the rest of the model is built. There are some requirements which a data set must fulfill in order for good models to be trained on its contents which result from the fundamental statistical assumptions from which the ML formalism is derived[79]. First and foremost, it is crucial that the data set gives a good representation of the input-target relationship over as large part of the relevant input space as possible. This includes a good sampling of the input coordinates and target variables, the distributions of which should be as close to Gaussian as possible. If the data set contains incomplete, contradictory or redundant examples, these should be removed. If there are multiple input variables with strongly disparate magnitudes, this can make it difficult for the model to learn more than just general trends. In this case, it is frequently a good idea to independently scale the various inputs and targets to normal distributions

[4] The term 'validation set' is also used in the context of neural networks with a similar but not identical meaning. See section 3.5

with identical mean and variance (usually, a mean of 0 and a variance of 1 are preferred) before the data is used in the model. The scaled value for an input or target variable v is then obtained as

$$v' = \frac{v - \bar{v}}{\sigma_v}, \tag{3.7}$$

using the mean $\bar{v}$ and standard deviation σ_v of the variable in the training data set. Other important metrics to keep track of in the data set are the means and variances of the output distributions. Knowing the these values is crucial for the interpretation of quality metrics calculated for trained models.

As per the mathematical basis for most ML methods, input variables should not be correlated to each other. If there are such correlations, either removing some of them may help, as can combining them into a new variable. In many applications, almost all input variables must be pre-processed in this manner. In ML for chemical applications, the inputs are frequently the Cartesian coordinates of the involved particles, whereas the properties to be predicted are invariant to rotations or translations of the entire system. Just as a neural network should recognize a cat independently of whether it is facing left or right, the energy of a molecule in vacuum predicted by an ML model should not change. Input variables which are not well-suited for direct use must therefore be transformed into a *representation* or descriptor to introduce such invariances. Because this step is so crucial for the models in this work, I will discuss different representations for molecular geometries in detail in the following section.

3.3. Representations of Molecular Geometry for Machine Learning

For use in an ML model, it is advantageous to transform the 3D-coordinates of the molecule or system into a more suitable representation that obeys common invariances, e.g. invariance to rotations and translations as well as permutation of the atomic index ordering, as this often leads to faster-learning models.[86] The simplest way to introduce invariance to rotations and translations to the $3N$ Cartesian coordinates of an N-atomic system is to use internal coordinates, e.g. the interatomic distances, or combinations of distances and angles. As chemical interactions between atoms decay with increasing interatomic distance, the matrix of inverse interatomic distances is a frequently-used, easy to calculate representation. If the chemical element of every atom is encoded in the diagonal term of the distance matrix, the resulting representation is called the *Coulomb matrix* (CM) representation[87]:

$$x_{ij} = \begin{cases} 0.5 Z_i^{2.4} & , i = j \\ \frac{Z_i Z_j}{\|\mathbf{r}_j - \mathbf{r}_i\|} & , i \neq j \end{cases} \tag{3.8}$$

As the numbering of the atoms in a system (and therefore a given atom's position in the CM) is arbitrary, swapping the indices of two atoms of the same element should not have any effect on the representation. In order to introduce this permutational invariance, the CM can be sorted by row norms.

Both the CM and the inverse distance matrix contain quite a bit of redundant information, and this is compounded the larger the number of atoms is: An N-atomic molecule has $3N - 6$ internal degrees of freedom, but the corresponding distance matrix has N^2 entries ($(N(N+1))/2$, if only the upper triangular is used). Both representations are also *global*, i.e. they encompass the entire system. There are many other global representations which add bonds or many-body terms, e.g. the bag-of-bonds or SLATM representations[88].

All global representations have a fixed size dependent on the number of atoms in the system, so using them with models which should be able to give predictions for molecules of varying size can become problematic. This can be circumvented to some extent by padding the representation for a small molecule with zeros until it has the same size as the largest representation in the training set. However, this padding introduces yet more redundancies, and the trained models still have an upper bound to the number of atoms which can be described at once.

In general when training models to learn chemical quantities, the more of the underlying physics is encoded in the representation, the faster and easier it will be for the model to learn. The representation can expose the relevant patterns in the data by handling some of the more fundamental functional dependencies, e.g. that interatomic interactions are weaker the further two atoms are from each other. One time-honored approach in theoretical chemistry is to describe molecular properties as a combination of atomic contributions (c.f. the LCAO ansatz), and this method is also used in ML for chemistry. *Local* representations describe the environment of an atom by probing the space around the atom with a set of two- and multi-body basis functions to quantify the presence and location of atoms. The resulting representation takes the form of a (frequently long) vector for each atom, which can then be combined to predict properties for the entire system. The individual basis functions are usually of the form

$$G(x) = \xi(x) f_{\text{cut}}(x) \exp\left(-cx^2\right) \tag{3.9}$$

where x is the distance of a pair of atoms, resulting in a two-body contribution, or a pair of distances for three-body terms. f_{cut} is a smooth cutoff function, c is an empirical constant and ξ is a relative weight factor for the two- and multi-body contributions.

Examples of representations using this approach include the symmetry functions introduced by Behler and Parrinello [89], atomic environment vectors [9], the SOAP kernel [86] or the FCHL representation [90, 91]. These representations can result in excellent models[86, 92] because they make so much of the information about the environments of atoms available and transparent. However, they are also comparatively expensive to calculate and are not very well suited when the property cannot easily be decomposed into atomic contributions. They therefore have difficulties with large molecules, intermolecular interactions and long-distance effects such as Coulomb interactions.

For large, rigid molecules, frequently reduced geometric representations are used such as a few characteristic distances between atoms or angles[93]. This type of representation has a lot in common with the choice of collective variables for metadynamics simulations or the analysis of complex MD trajectories. The resulting representations are very system-specific and finding them requires significant domain knowledge. Additionally, they frequently reduce the system too much, limiting the upper bound of any ML model's prediction quality.

3.4. Kernel Ridge Regression (KRR)

The kernel trick is a clever way to solve a nonlinear regression problem using linear regression[94]. At its core lies the idea of transforming the input-target pairs from the training set into a higher-dimensional space, where the relationships between each input and the target are linear, performing the linear regression in that space, and transforming its coefficients back to the original feature space. In most cases, however, finding this perfect higher-dimensional space would be impossible. Using the kernel trick allows us to avoid explicitly formulating the mapping function to the higher dimensional space by instead describing the relationship of the data points in that space via a kernel function.

More formally, a kernel ridge regression model uses a basis of kernel functions placed on a number of training samples to expand the target value:

$$\hat{y} = \sum_{i}^{N_{\text{train}}} k\left(x^q, x_i\right) w_i \tag{3.10}$$

where x^q is the representation of the query sample, the x_i are the representations of the training data points, and the coefficients w_i are the trainable parameters. k is the kernel function depicting the similarity of two points in the higher-dimensional kernel space. The most frequently used non-linear kernel is the Gaussian kernel, defined as

$$k(x, x') = \exp\left(-\frac{\|x - x'\|_2^2}{2\sigma^2}\right), \tag{3.11}$$

where σ is kernel width or length scale of the problem, a hyperparameter of the model, for which the optimal value is strongly coupled to distances between the training data points. The Laplacian kernel is also used especially in chemical contexts where the training data is almost noiseless [94]:

$$k(x, x') = \exp\left(-\frac{|\, x - x' \,|}{2\sigma^2}\right), \tag{3.12}$$

Given a set of input–target pairs $(\mathbf{x}, \mathbf{y})$, there is a closed-form solution to obtain the regression coefficients:

$$\boldsymbol{w} = (\mathbf{K} - \lambda \mathbf{I})^{-1} \mathbf{y}, \tag{3.13}$$

Here, the kernel matrix $\mathbf{K}$ contains the pairwise kernels between all data points in the training set. λ is called the regularization coefficient, a hyperparameter of the model used to guarantee that the kernel matrix is invertable. The value for λ also reduces the terms on the diagonal of the kernel matrix, thus lowering the influence of each individual data point for the prediction and helping to avoid overfitting.

The prediction for a new data point is then obtained using Equation 3.10 by evaluating the kernel function between the query data point x^q and each of the training data points x_i and multiplying the resulting vector with the vector of fitted weights $\mathbf{w}$.

Compared to neural network or other iteratively learning models, KRR has the advantage of a closed-form solution which can be easily calculated by inverting the matrix in Equation 3.13. A KRR model has only very few hyperparameters – the regularization strength λ, the type of kernel used and kernel-specific parameters such as the Gaussian

width σ. The various kernel types can capture different types of nonlinearity, and the value of the kernel of a query data point with the training examples can be used as an estimate of the uncertainty associated with the prediction.

However, obtaining a prediction for a single data point requires that the kernel between that point and all training samples is evaluated. The computational cost for prediction therefore scales with increased training set size. When training sizes are large, the construction and inversion of the kernel matrix can also become prohibitively costly. Due to the closed-form solution, KRR models do not provide an easy way to adapt the loss function to a specific application, nor are they amenable to active learning approaches where data is successively added to the training set. Additionally, obtaining the gradients of a KRR model w.r.t. its input coordinates is quite complex and costly. KRR models therefore shine especially in contexts with small data sets or when used in conjunction with a technique to decide a priori which data points are included in training.

3.5. Neural Networks

Neural networks are non-linear statistical models which are widely used to solve complex regression or classification problems. At its core, a neural network consists of 'layers', where each layer makes multiple linear combinations of its inputs, transforms them via a non-linear activation function, and passes the results to the next layer. An intuitive representation of this structure is shown in Figure 3.5. The simplest kind of neural networks takes a vector of inputs $(x_1, x_2, \dots x_i)$ and makes a number of linear combinations $(n_1, n_2, \dots n_j)$ referred to as neurons of these inputs, each with a different set of weights w_{ij} for each of the inputs. The result of each linear combination n (i.e. the value of each neuron) is then passed through a non-linear *activation* function f, and the results taken as the inputs for the next layer of neurons. The output of the ith neuron in the lth layer is then calculated as:

$$n_i^l = f\left(\sum_j^{N^{l-1}} w_{ij}^{l-1} n_j^{l-1}\right), \tag{3.14}$$

where N^{l-1} is the number of neurons in the previous layer and w_{ij}^{l-1} is the weight parameter connecting the jth neuron in layer $l-1$ to the ith neuron in the current layer. In the very first layer, instead of the previous layer's output n_j^{l-1}, the elements of the input representation x are used instead. The final layer of the network then carries the result, i.e. the prediction the network makes for a given input.

The non-linear activation function is what allows the neural network to approximate arbitrarily complex functions. Originally, activation functions like the *tangens hyperbolicus* (tanh) or sigmoid functions were overwhelmingly used, inspired by the biology of firing synapses. The sigmoid function takes the following form:

$$f_{\text{sigmoid}}(v) = \frac{1}{1 + \exp^{-v}} \tag{3.15}$$

Both the sigmoid and tanh functions saturate the larger the absolute value v of the linear combination of the neuron's inputs. This behavior can cause problems if networks with

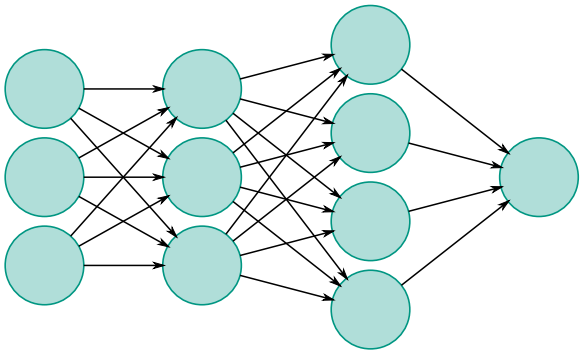


Figure 3.5.: A neural network represented as a directed graph. Green circles represent the individual neurons, which are arranged in clearly defined layers. The leftmost layer receives the inputs to the network, and the sole neuron in the rightmost layer carries the network's output. The arrows between the neurons represent the linear combinations which determine the input of every neuron beyond the first layer. The weights associated with the edges in the graph are the trainable parameters in the model.

many layers are trained, as the saturated areas can lead to the gradient needed for training the model vanishing. To alleviate this, activations like the rectified linear unit (ReLU) became more popular:

$$f_{\text{ReLU}}(v) = \max(0, v) \tag{3.16}$$

For positive v, the gradient of the ReLU certainly does not vanish, but for negative inputs, the gradient is 0 and can cause the neurons to get stuck during training. The leaky variant of the ReLU function has a small, but nonzero gradient for negative values v, circumventing this problem:

$$f_{\text{leakyReLU}}(v) = \begin{cases} v & \text{if } v > 0 \\ \alpha v & \text{otherwise} \end{cases} \tag{3.17}$$

The ReLU function is not smooth, so smooth approximations like the softplus function

$$f_{\text{softplus}}(v) = \log\left(\exp(v) + 1\right) \tag{3.18}$$

are used to obtain fully differentiable models and can be combined with a leaky slope for negative values to avoid the vanishing gradients.

The weights in the NN are initialized randomly and adjusted such that the prediction for inputs for which the result value is known matches the reference value as closely as possible as measured by a metric referred to as *cost* function. Cost functions can be as simple as the mean squared error (MSE) between prediction or reference,

$$\text{MSE} = \frac{1}{N} \sum_{i=0}^{N} (y_i - \hat{y}_i)^2 \tag{3.19}$$

but can also include other terms to impose additional constraints on the structure of the network.

Training is then the process of minimizing the cost function by giving the network inputs from the training data set and evaluating the cost. Minimization uses the backpropagation algorithm, repeatedly making small steps towards the nearest local minimum of the function[5]. Starting from a network with randomly generated initial weights, the derivative of the cost function C w.r.t. each weight w_{ij} is calculated by the chain rule with some optimizations to avoid redundant calculations. Then, for every input–target pair in the training set, the gradients $\frac{\partial C}{\partial w_{ij}}$ are evaluated, and an *optimizer* algorithm follows these gradients to generate a new set of weights by making a small step in the direction of the minimum:

$$w_{ij}^{m+1} = w_{ij}^{m} - r\frac{\partial C}{\partial w_{ij}} \tag{3.20}$$

The parameter r determines the size of the steps and is called *learning rate*. If the learning rate is too high, the model will not be able to fully converge into local minima, if it is too low, the learning process becomes exceedingly slow, raising computational costs during training. This process takes many passes over the training data set to converge to a minimum, and it is not uncommon to train NN models for hundreds or thousands of such *epochs*.

The gradient descent algorithm used during the training and its parameters are a crucial factor for the quality of the results. Due to the high-dimensional nature of the loss surface traversed during training, there are potentially very many local minima, some of them very shallow. While finding the global minimum is not computationally feasible, several techniques have been developed to avoid getting stuck in narrow or shallow minima. One simple yet effective technique is changing the learning rate (the length of each 'step' in the descent) as training progresses, initially making large steps to find wide, low basins and only then reducing the learning rate to refine the results.

Due to the large number of trainable parameters, neural networks can be quite prone to overfitting. An essential tool to avoid overfitting on an NN on the training set is the technique referred to as *early stopping*: By periodically evaluating the model's performance on a set of data points not used to adjust the weights (the so-called validation set), the point at which the model starts learning idiosyncrasies of the training set can be determined, as the loss function calculated for the validation set will begin increasing as the loss on the training set continues to decrease. Model training can then be stopped at the point where the network gives the best performance on the validation set, which should correspond to the most generalizable parameters.

Another method to avoid overfitting in neural networks is regularization. The smaller the weights in the network are, the smoother the resulting function is w.r.t. the input parameters. Therefore, by adding the sum of the weight norms as a penalty to the cost function, the model can be pushed to converge to smoother functions. L_1 regularization adds the following term to the loss function:

$$L_1 = -\lambda \sum_{ij} \mid w_{ij} \mid, \tag{3.21}$$

where the sum of the absolute values of weights is weighted by the regularization strength λ, which corresponds to the strength of the push to more general parameters. An over-

[5] For a detailed and intuitive look at the backpropagation algorithm, consult [81]

regularized model (i.e. a model where λ is too large) will underfit. A stronger regularization penalty to large weights can be applied with the L_2 function:

$$L_2 = -\lambda \sum_{ij} \mid w_{ij} \mid^2, \tag{3.22}$$

Of course, both functions may also be applied in combination if necessary. *Dropout* regularization is another method to prevent an NN model from overfitting. By switching off random subsets of neurons in each epoch, it can ensure that no individual connections in the network are overpowering or irrelevant. As a result, the distribution for the weights in the network becomes narrower and the entire model again becomes smoother.

Neural networks are powerful universal approximators, and architectural variants not discussed here can make them applicable to problems as diverse as protein folding and complex team-based problem-solving in strategic multiplayer games. However, this flexibility also comes at a cost – neural networks have far more hyperparameters to choose from than e.g. KRR models, and the extremely large amount of parameters makes training opaque and cost-intensive. These challenges notwithstanding, neural networks have become an essential tool in many applications due to their versatility and prediction power.

3.6. Hyperparameters and Optimization Strategies

Apart from the parameters fitted to the training data, any machine learning model has *hyperparameters*. In the case of a neural network model, common hyperparameters can be the number of neurons and layers, activation functions or the type of cost function. The conceptually simpler KRR models have three main hyperparameters: the kernel function, kernel width and regularization strength. These parameters represent design choices, and are not optimized in the process of fitting the model to training data. Nevertheless, the choice of hyperparameters can strongly influence the performance of the resulting models, and bad hyperparameters can prevent models from learning at all. Finding the proper values for the hyperparameters of a model is far from trivial, however.

Hyperparameter optimization usually requires repeated training of models on the same data set while the values for the hyperparameters are varied. This can be quite computationally demanding, as the search space can grow rather large very quickly. While educated guesses about the influence of specific hyperparameters on model performance can reduce the search space for hyperparameter optimization, this space can be too large to fully sample, requiring clever techniques for parameter exploration. A complex method for systematic hyperparameter optimization may have tunable parameters itself, only shifting the optimization problem one level upwards. Therefore, a crucial rule to keep in mind is that the goal of hyperparameter optimization is *not* to find the best possible parameters, but to find a set of parameters which is sufficiently good for solving the problem at hand.

3.6.1. Grid Search and Random Search

The naive approach to hyperparameter optimization is to formulate the search space as a set of values to be tested for every optimizable hyperparameter. Then, a candidate model can

be trained for each such combination, the models evaluated on out-of-sample data, and the model with the lowest error or best metrics is the designated winner. This method, known as *grid search* is simple and systematic. The sampling resolution for every hyperparameter is tunable by the density of the predetermined grid, but determining reasonable bounds and intervals to test for each hyperparameter may be difficult without some preliminary exploration or multiple iterations of the grid search as the method does not converge on promising parts of the search space itself. The number of candidate models to be trained is known a priori, so it is easy to adjust the search space to the amount of computational power one is willing to invest. However, the number of candidate models quickly grows large, and every candidate model is trained to completion, even if it's clear that that specific set of hyperparameters is far off the optimum. To reduce the computational cost, instead of training all candidate models in the search space, a certain amount of parameter combinations can be chosen at random from the grid. The resulting *random search* allows balancing the time invested with the thoroughness of the search and can alleviate some disadvantages of the grid search.

3.6.2. Hyperband Algorithm

The Hyperband algorithm[95] for hyperparameter optimization in iterative learning methods uses random search to sample the hyperparameter space. However, instead of training every model to full convergence, the algorithm trains a large amount of models for very few epochs and only continues training the models which show most promise. The algorithm has two main loops, *brackets* and *rounds*. Each bracket of the algorithm is assigned a computational budget, which can be stated in terms of e.g. a maximum allowable number of epochs to train across all models. At the beginning of each bracket, the algorithm generates a population of candidates with random hyperparameters. Then, a certain fraction f of the budget available for the bracket is distributed among these candidates to begin their training. The candidates are then compared, and all but the top-performing f-th part of the population are discarded. The best models then go to the next round of training, where one-f-th of the remaining budget is used to continue their training. This process continues until the budget has been fully used and only one model remains, which is the winner of this bracket. Then, a new bracket can be started to sample more of the hyperparameter space. In each bracket, the number of models in the population is varied. This allows the method to switch between training a lot of models for a short time each and training few models for longer before deciding which to discard.

This approach has one great advantage to many other hyperparameter optimization methods: its only user-facing parameter is the factor f, with which the speed at which the population is culled can be adjusted. It leverages the strength of random sampling to cover large search areas and avoids the pitfall of investing resources in obviously unusable models. However, the Hyperband algorithm can run into difficulties when the learning rate of the models is a tunable hyperparameter, or when other hyperparameters affect the convergence rate too strongly (as the number of layers or neurons in an NN model can).

3.7. Uncertainty Estimation and Outlier Detection

Generally, it is not easy to give error bars for predictions obtained from ML models, as the error can vary wildly depending on the specific inputs. Additionally, the behavior of a model for inputs for which it is forced to extrapolate depends heavily on the type of model, the training data, and often even random factors such as the initialization of weights. Having an estimate of the uncertainty of a given prediction can be very important during the use of the model: If a computer vision model used in a self-driving car is only 60 % sure that a given object is a shadow and not a person, it should be very careful in driving over that spot. In ML models for computational chemistry, having an uncertainty estimate is especially helpful if the model predicts forces used in an MD simulation, because on bad prediction, the simulation framework can fall back to a reference method. Uncertainty estimates can also be used to augment or tailor the training set, only adding those data points for which the model does not give good predictions already. This technique known as *active learning* is especially valuable for models where the number of data points which can be included in the model are limited (e.g. KRR), or when generating training data is expensive (e.g. high-level QM calculations).

Some metrics evaluated on the test set can be used to give rough estimates for the error distribution in the predictions of the model. The mean and variance of the absolute error can give estimates for overall reliability, while the maximum errors on the test sets can give an indication of the worst-case scenarios. However, these metrics must be calculated on the test set, and cannot be calculated for a given query point without calculating the ground truth value of the target.

Some types of ML models have built-in uncertainty estimates (Gaussian process regression and Bayesian models in general), but these models have other disadvantages such as large computational costs during training and limited flexibility. In KRR models, the kernel function calculated during prediction shows the similarity between a query point and each point in the training set. If the elements of the kernel are very small, the query point is very dissimilar to the training set and the prediction will tend to the baseline of 0. This can be used as a rough estimate of model uncertainty.

In neural networks, there are a few ways to obtain estimates for the uncertainty of the prediction. A simple and commonly used method is to train not one but multiple networks for a given task with different hyperparameters or on different subsets of the data. For each query, the predictions of all networks are collected and compared. If the predictions are similar, it is likely that the queried data point was indeed well-represented in the training set and the prediction error is low. In contrast, if the networks disagree, chances are high that what's determining their predictions is the random factors during their training and the point was undersampled in the training set.

Part II.

Contributions

4. Charge and Exciton Transfer Simulations Using Kernel Ridge Regression Models

Charge and exciton transfer simulations using the methods described in section 2.4 are a useful tool for materials discovery, as they allow screening of materials for desirable properties without synthesizing them. Additionally, they allow insights into the mechanism and pathways of the transfer, which are essential for understanding the function of biological systems such as light-harvesting complexes involved in photosynthesis. However, the computational cost of the simulations limits their applicability. The approximations introduced in subsection 2.4.2 allow for simulations of charge transfer in organic semiconductor candidates at the relevant time scales, but only by limiting the propagation to a one-dimensional chain of molecules within the system. The calculations necessary to run exciton transfer simulations within the propagation formalism are much more expensive, as they require the orbital structure of both the ground and excited states of every fragment.

Therefore, a faster way to obtain the site energies and pairwise couplings within such systems is desirable. Any machine learning model for this application must be able to capture the changes of site energies and couplings during an MD simulation, as well as be simple to train and give predictions exceedingly quickly in order to compete with DFTB for charge transfer.

Machine learning methods for charge and exciton transfer properties have been investigated previously: Lederer *et al.* [93] used a kernel ridge regression model with a very small geometric representation to predict charge transfer couplings in pentacene crystals. Musil *et al.* [96] trained Gaussian process regression models for charge transfer couplings in both pentacene and azapentacene using a simplified SOAP-Kernel, reaching errors of a few meV with only a few thousand high-quality training examples. Çaylak *et al.* [97] presented a neural network model, which predicted hole mobilities and charge transfer couplings for an amorphous aluminum complex. For excitonic properties, Häse *et al.* [98, 99] used a modified Coulomb matrix together with a neural network to predict excitation energies of Bacteriochlorophyll in the Fenna-Matthews-Olson complex and study the exciton dynamics. Finally, Wang *et al.* [100] published a systematic investigation into the effects of representation and target value on model training and prediction performance of ML models for charge-transfer properties.

However, most previously published methods require large amounts of *ab initio* level training data, system-specific representation design or extensive hyperparameter optimization. These factors hinder the easy applicability of the model training procedure developed

for one molecular system to other contexts. None of the aforementioned procedures attempt to generalize across chemical space and give predictions for systems with different chemical compositions, so going to a new system requires completely re-training the models. Also, it is unclear how the raw accuracy of the ML model on low-level electron transfer parameters such as couplings affects the observable properties of the system obtained from simulations. Evaluating the models in that respect requires their explicit use in direct simulations, which has previously not been done.

In this chapter, I describe a method to directly predict charge and exciton transfer site energies and couplings from a given geometry using kernel ridge regression models. I develop a simple and automatable training procedure, which can be easily integrated into multi-scale workflows and avoids the need to generalize across chemical space. Finally, I apply the trained models in direct charge and exciton dynamics and compare the obtained observables to reference data from DFTB and experiment.

The results presented in this chapter have been published[101], parts of this chapter follow that publication closely and are used with permission. Training data for the models was generated by Daniel Holub and Philipp Dohmen, with the latter also performing all propagation simulations. Model design, training and evaluation was done by me, with Anders Christensen and Weiwei Xie contributing in advisor roles. Klara Eckhard contributed to automating the model training process as a student assistant. All model design, experiments and evaluation were performed by me.

4.1. Basic Requirements and Design Decisions

In order to replace DFTB in CT and ET propagation simulations, a machine learning model needs to fulfill several requirements.

First, it must be able to capture the geometry-dependent changes of site energies and coupling data adequately well to reproduce observables as closely to the DFTB reference as possible. This is challenging to evaluate, as there is no clear quantitative cutoff in form of an error metric for prediction (e.g. ‘absolute errors for couplings no bigger than x meV’). The only way to properly evaluate whether a model is good enough for running the propagation is to use it in the simulations, which requires implementing the model into the simulation code. Therefore, the second requirement is that implementation of the prediction should be as fast and easy as possible.

Third, model training can be decoupled from the simulation code, but must be as automatable as possible in order to allow easy training of models for new molecular systems. This approach of training models for individual molecular systems (e.g. one model for anthracene, another for pentacene, etc) avoids two of the most complex problems that machine learning models for chemical application face: constant size molecular representations and generalization across chemical space. The difficulty of designing efficient yet universally applicable molecular representations has been discussed in section 3.3. Creating models which are able to learn a given property for all known molecules is orders of magnitude more difficult than creating models which only capture the limited part of chemical space which represents a given molecule and its thermally accessible distortions. Therefore, I limited the models’ applicability to a single molecular system of identical

monomers of constant size, and automated the process of training models from scratch. While this requires generating new training data for every system for which a model is to be trained, calculating the DFTB references for several thousand structures is very cheap. The structures themselves can be obtained from the equilibration step of the molecular dynamics workflow. In this manner, it is possible to seamlessly insert training an ML model between equilibration and propagation.

The final critical requirement is that the ML models should be able to outperform DFTB for both charge and exciton transfer applications. The approximations taken in the CT propagation formalism using DFTB make it extremely efficient, as calculation of the orbital structure of the charged state is avoided. Additionally, the fragmentation allows calculating one fragment at a time, and the values for site energies and couplings obtained from non-self-consistent DFTB are of sufficient quality to perform the simulation. Thus, the entire calculation is reduced to one diagonalization per fragment and step, with the calculations of couplings being comparatively inexpensive matrix operations. Therefore, beating DFTB for charge transfer in terms of speed is a high bar to clear. For exciton transfer, the calculation of the excited state cannot be avoided, and therefore the DFTB reference calculation is significantly slower.

With these requirements in mind, a reasonable first candidate model was the kernel ridge regression (KRR) method. As introduced in section 3.4, KRR uses the kernel trick to allow for a linear regression in a non-linear problem by implicitly transforming the problem into a higher-dimensional kernel space. Training of and prediction by KRR models can be summarized neatly in a few equations, and the prediction specifically can be very concisely implemented. Additionally, KRR models have very few hyperparameters, which is advantageous for optimizing the training and hyperparameter search process.

As the site energy is a monomeric and the coupling a dimeric property, the question arises whether to attempt training a single model for both quantities, or whether separating the problems into two models is better. Training a single KRR model for both properties would require a representation which has the same dimensionality for a monomer as for a dimer structure. Most representations developed for chemical coordinate data (section 3.3) do not have constant size. Therefore, if a model is to be trained for both site energies and couplings, the dimensionalities of the monomer and dimer representations must be made compatible. One simple method to do this is padding the monomer representation with zeros until it reaches the dimensionality of the dimer representation. Another option is to calculate the representation for monomers as though they were dimers of two identical geometries. Both options introduce redundancy to the representation, which may unfavorably affect model performance.

Additionally, the respective targets (i.e. site energies and couplings) lie orders of magnitude apart. For anthracene, site energies are about 5.8 eV on average, while magnitude of couplings ranges between 0 meV–150 meV. The formalism for training most ML models (including KRR) is derived under the assumption that the target variable follows an uni-modal normal distribution (see section 3.1 for details). Therefore, a single KRR model asked to reproduce this bimodal distribution will not be able to do so with good quality[1].

[1] Neural network models would struggle as well.

Both the input dimensionality and target distributions therefore point towards training two separate models – one model gets monomer coordinates as input and predicts site energies, the other uses dimer structures to predict couplings.

4.1.1. Generation of the Reference Data Set

Crystalline anthracene is a very well-studied organic semiconductor, making it a suitable test system for the development of ML methods. Regarding number of atoms, anthracene ranges at the lower end of the spectrum of interesting semiconductor materials, but on the high end of what ML representations and models for chemical structures are tested and evaluated on. Training and test data was sampled from an MM MD simulation of an anthracene crystal. Force field parameters for the MM dynamics were obtained from the general AMBER force field (GAFF)[102, 103]. Atomic charges were obtained using the restrained electrostatic potential fitting procedure (RESP) [104, 105] calculated at the HF/6–31G* [106, 107] level of theory using Gaussian09 [108]. Setup of the system and classical MD simulations used version 5.0.4 of the GROMACS package [109, 110]. The initial structure was constructed from the data published in Mason[111], which was expanded to form a crystal containing $10\times40\times5$ molecules along the *a*, *b*, and *c* crystal axes. The energy of the initial structure was minimized and the temperature then equilibrated at 300 K for 1 ns using the Nose-Hoover thermostat [112]. Structures for the data set were then obtained from every 1000th step of a 10 ns simulation originally performed with a time step of 2 fs to minimize the temporal correlation of the structures. Within the crystal, a subset of 75 molecules was chosen (a continuous $5 \times 5 \times 3$ block) for the reference data generation.

Keeping training set sizes low and without redundancy is advantageous for KRR model performance (both in terms of prediction time and quality). Within the block of molecules there are many pairs which are far away from each other and have no meaningful coupling interaction. Therefore, three different data sets for couplings were generated: First, the `full` data set, which contains all pairwise couplings in the $5 \times 5 \times 3$ block. Two other data sets were generated by including only dimers whose center of mass distance was below 0.75 nm (`short` data set) and 1.25 nm (`long` data set). The cutoff of the `short` data set was chosen so that it includes the first neighbors of every fragment along the *a*- and *b*-crystal axes, while `long` cutoff also includes the first neighbors along the *c*-axis, as well as the second neighbors along the *b*-axis.

The reference charge transfer site energies and couplings for the geometries in each data set were then calculated using the non-self-consistent variant of DFTB [1, 46] using the approximations and implementation described in Heck et al.[36]. Only the HOMOs were considered for these calculations, and electronic couplings were scaled by the factor of 1.54 which has previously been determined to improve the accuracy of the couplings obtained with DFTB to that of second-order coupled cluster (CC2) calculations.[66] For the exciton transfer data sets, excited states calculations were performed using long-range corrected time-dependent density functional tight binding (LC-TD-DFTB) [2, 3, 47, 51] as implemented in DFTB+ [4, 5].

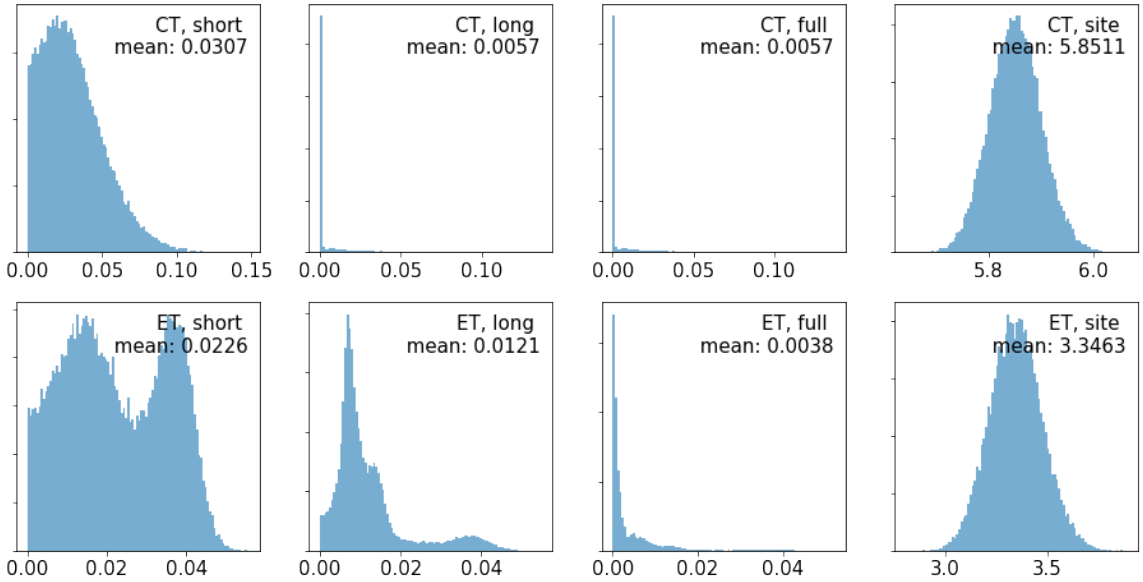


Figure 4.1.: Distributions of the target values for site energies and the absolute value of couplings in the three data sets for both charge and exciton transfer.

4.1.2. Choice of Prediction Targets

Figure 4.1 shows the distributions of site energies and couplings for both charge and exciton transfer across all three data sets. For the site energies (rightmost column), the distributions are almost normal, and the values can be used without further preprocessing as targets for KRR models[2].

For the couplings, the situation is a bit more complex. As previous studies of similar learning problems[93, 96] used the logarithm of the coupling as prediction target, I initially also trained models to predict $\log|V|$. While this worked well for data sets sampled from one-dimensional chains of molecules along one crystal axis, the models performed worse for the data sets described above. Wang et al.[100] investigated different methods for KRR models and reported that taking the logarithm was unnecessary if the models were well-optimized. We therefore adjusted our methods and removed the logarithm from our models and I only repeated the experiments whose results directly depend on the scaling of the targets.

Without the logarithmic scaling of the couplings, it became possible to also consider the sign of the coupling integral as part of the calculation. The sign of an individual coupling is a consequence of the usually arbitrary signs of the wave functions on the individual fragments. Along a dynamics simulation and across many pairs, the relative relationships between the signs must be preserved in order to make the simulation as a whole consistent. This is usually done by fixing the sign of the wave function coefficients to some geometric parameter (e.g. the direction of a given bond relative to the rest of the molecule) in the

[2] Contrary to neural networks, for KRR models it is not necessary to scale features or targets to a normal distribution with a mean of 0 and variance of 1. In the application presented here, scaling the features can actually be detrimental to the model performance (see orange line in Figure 4.4c)

initial calculations of the fragments. However, the connection between the geometry and the sign of the coupling can be tenuous, and first experiments showed that KRR models were not able to reproduce the sign correctly unless given very large amounts of data ($\approx$ 60 000 structures). KRR models become increasingly unwieldy in both training and prediction with larger training sets, and this problem is exacerbated by the size of the representation necessary to capture the degrees of freedom. In order to circumvent these issues, we decided to train models only on the absolute value of the couplings, and use a chain of anthracene molecules as a proof-of-concept system for propagation. If the charge is limited to movement along a one-dimensional chain, the sign of the coupling is irrelevant for the propagation mechanics.[113]

Using the absolute value of the coupling distorts the distributions (which are slightly multi-modal due to the clear presence of first and second neighbors) to the forms seen in Figure 4.1. In first experiments, the KRR models were nevertheless able to reproduce the target distributions, indicating that the partial undersampling of parts of the target space and the reduction to the absolute value did not invalidate the core assumptions for the fit.

For both the charge and exciton transfer case, it is evident that increasing the cutoff from the `short` to the `full` data sets mostly adds structures whose couplings are close to or equal to 0. In the charge transfer case, the `long` and `full` data sets are identical. This is due to the fact that the integrals tabulated in the Slater-Koster files for DFTB are limited to interatomic distances of 20 a_0 (1.05 nm). While the cutoff used for construction of the data set was framed in terms of center-of-mass distance, it is apparent that in the anthracene crystal, there are no pairs whose centers of mass have a distance larger than 1.25 nm for which two atoms are close enough to each other to result in a nonzero coupling interaction. In the following, I nevertheless treat the `long` and `full` data sets as separate, in order to keep the presentation consistent with the exciton transfer data sets where these data sets are distinct.

4.1.3. Comparison of Representation Variants

As noted in section 3.3, choice of representation can make or break an ML model. The decision to make models system-specific with a fixed atomic composition allowed the use of both global and local representations. However, the size of the representation (e.g. the number of entries in a vector) and the computational cost to calculate it can strongly impact the speed of individual predictions. Therefore, I decided to investigate two representations – coulomb matrices (Equation 3.8) and FCHL19 (section 3.3). Implementations of both representations are readily available from the QML python package [114].

The coulomb matrix (CM) is a conceptually simple representation, based on the inverse of the interatomic distance matrix. It is cheap, easy to calculate and compatible with standard kernels such as the Gaussian or Laplacian. However, an N-atomic system has $3N - 6$ degrees of freedom, but its CM representation has $\frac{N\cdot(N+1)}{2}$ entries[3]. The CM representation is therefore somewhat redundant and its size grows with the square of the number of atoms per fragment.

[3] As the matrix is symmetric, its upper triangular contains the same information as the entire matrix.

FCHL19[91] is a local density-based representation similar to the symmetry functions introduced by Behler *et al.* [89] or the SOAP kernel [115]. Using FCHL19 for KRR models requires using a kernel specific to that representation in order to differentiate chemical elements. Both the representation itself and the kernel are more costly to calculate than the coulomb matrix representation. However, the FCHL19 representation has been shown to reduce the number of training examples required to reach a given accuracy threshold[116], which might compensate for the increased computational cost.

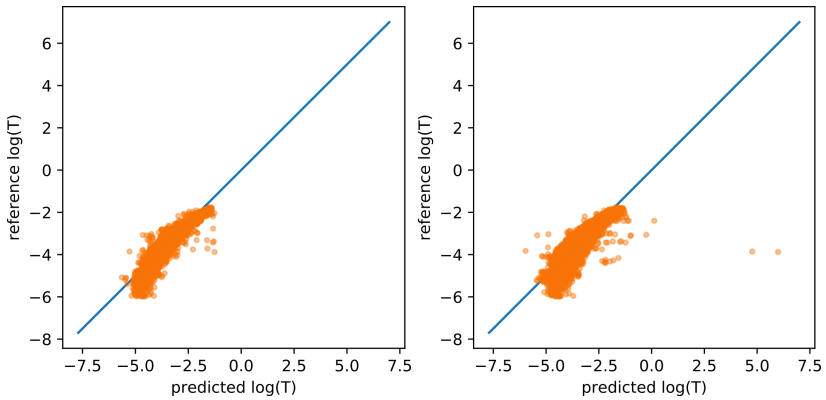
In the following, I present and briefly evaluate several variants of the coulomb matrix representation and the FCHL19 representation for use in this project. All models presented in this section use $\log|V|$as learning target and the `full` data set. Due to the exploratory nature of these models, I do not give a detailed evaluation of each model and instead focus on the relative changes between the different representation variants. This exploration was only necessary for the coupling models, as first tests with the site energies showed that they trained very well regardless of the model configuration. As the CT and ET data is fundamentally similar, the initial parameter exploration was only performed for the CT models.

4.1.3.1. Heavy-Atom Coulomb Matrix

In order to reduce the redundancy of the coulomb matrix, I investigated whether including only the non-hydrogen ('heavy') atoms can capture enough information for learning to be successful. As the Carbon-Hydrogen bond vibrations are not activated at 300 K, the relative motions of the hydrogen atoms to the anthracene core should be relatively small and may not contribute much to the variance of the couplings. Slight modification of the default QML version of the coulomb matrix representation allowed omitting the hydrogen atoms. With the number of atoms reduced to 14 per fragment, the coulomb matrix representation for the site energy models was reduced from 300 elements to 105 elements and the pairwise representation used for coupling predictions shrank from 1176 to 741 elements.

To compare the two representations, I trained one model for charge transfer couplings using the default coulomb matrix and another using the heavy-atom version. Both models were trained and evaluated on the same subset of 25 000 structures from the `short` data set using the Gaussian kernel with $\log|V|$as fit target.

As can be seen in Figure 4.2, both models were able to learn similarly well. However, the model with hydrogen atoms showed a prediction MAE of 5.5 meV, while the model without them reached only a MAE of 46 meV. Additionally, evaluation of the model using the heavy-atom coulomb matrix showed more outliers, i.e. that model had a far greater tendency to be very wrong in its predictions. Due to the log scaling of the prediction target, the couplings for these outliers are off by several orders of magnitude. These extreme cases may introduce instabilities to the transfer simulations, leading to erratic behavior of the charge or exciton. While only extensive test simulations can determine whether this occurs frequently enough to be a problem, the small benefit of a reduced representation did not warrant the investment of time and computational resources. I did not repeat this experiment for the other data sets or targets, as it was evident that the positions of the hydrogen atoms in the molecules encode information vital for the value of the couplings, and the reduction of computational cost was not very notable.



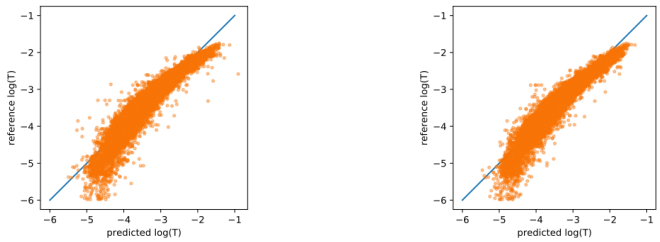
(a) (b)

Figure 4.2.: Two models for CT couplings in anthracene, (a) trained on the full coulomb matrix, the other (b) on the heavy-atom CM. Outliers in the heavy-atom model indicate that essential information is lost in this dimensionality reduction process.

4.1.3.2. Normalization of Coulomb Matrix Terms

For neural network models, it has been frequently established that scaling the individual input features to resemble normal distributions with a mean of 0 and variance of 1 facilitates the learning process[76, 81]. For KRR models, feature scaling may not always be necessary or helpful.

Therefore, I tested whether scaling the Coulomb matrix entries to normal distributions according to Equation 3.7 using the `StandardScaler` included in the `scikit-learn`[117] package. Results are shown in Figure 4.3, with the model using scaled features showing a slightly lower mean absolute error (4.8 meV vs 5.3 meV). The model using the scaled CM terms has a slightly narrower error distribution, and especially fewer outliers. This effect is small, however, with the variance of the absolute error distribution on the test set decreasing from 8.5×10^{-2} meV to 6.0×10^{-2} meV. Scaling the elements of the Coulomb matrix therefore appears to have small but tangible advantages for the quality of the model.



(a) (b)

Figure 4.3.: Prediction performance of a model using the default Coulomb matrix (a) and one for which the entries of the CM have been scaled (b) as described in section 3.2.

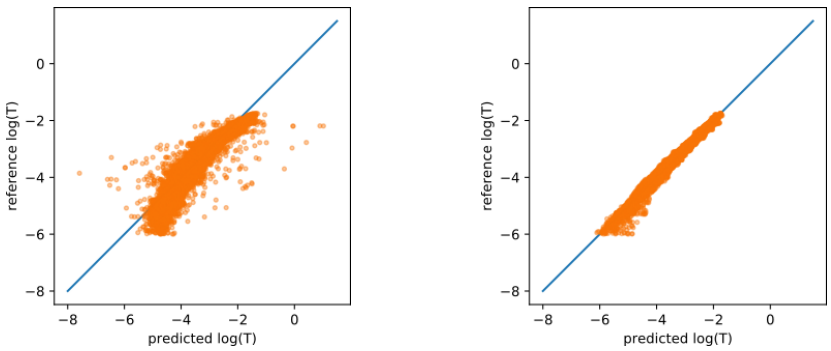
4.1.3.3. Forgoing Permutational Invariance of Coulomb Matrices

Sorting the Coulomb matrix by row-norm is the established method to make the representation invariant to swapping the indices of two atoms of the same element. This invariance is a desirable property for machine learning of molecular properties, as it uncouples the model's prediction from the order in which the atoms are listed. In this application, however, the atom indices do not change as the goal is to train the model from data generated during system equilibration where the atomic indices are already fixed. This allows us to forgo the sorting of the CM by row-norm, as this process introduces discontinuities in the function mapping atomic positions to the representation. These discontinuities can occur because near equality of the row norms does not necessarily imply chemical equivalence. As an example, an atom which is fairly close to most other atoms may have a very similar row-norm to one that is very close to a few atoms, but far away from the rest. The thermally accessible molecular motions may then cause these two atoms to swap places in the CM representation, depending on which of the two is currently closer to the center of mass of the molecule. These two atoms are chemically far from identical, however, and this may create difficulties for an ML model operating on the sorted CM representation.

Avoiding these discontinuities can make it easier for ML models to learn the true (smooth) geometry-dependence of the target value without spurious interference. While the continuity and smoothness of the predicted site energies and couplings is not a strict requirement for the propagation method to work, a smooth model is a closer approximation to the underlying physics and may give better results during propagation.

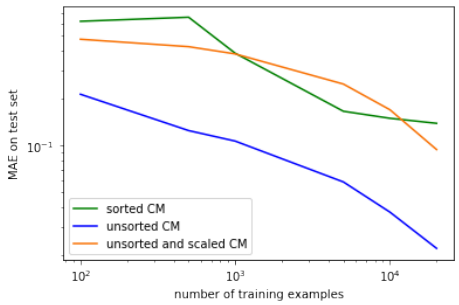
To examine the effects of using the unsorted CM, I trained three sets of models for charge transfer couplings on identical data sets of varying sizes. One set used the sorted CM, another used the unsorted CM, and a third set used the unsorted CM but applied feature scaling as described in subsubsection 4.1.3.2. All models used the Gaussian kernel and the logarithm of the absolute value of the coupling as target, learning curves were calculated on 100, 500, 1000, 5000, 10 000 and 25 000 training examples each to evaluate the prediction quality of the model as a function of training set size. Hyperparameters were optimized using the largest training set size for each representation variant and applied to models trained on fewer data points to calculate a rough learning curve without elaborate hyperparameter optimization at every step.

The results of model evaluation are shown in Figure 4.4. Leaving the Coulomb matrix unsorted helps the model to properly learn the couplings without having to filter out the noise from atoms switching places, as can be seen when comparing Figure 4.4a with Figure 4.4b. The model using the unsorted CM reaches an accuracy of ≈ 5 meV with far fewer training examples, and there are also fewer outliers. With an unsorted Coulomb matrix, however, scaling the feature and target distributions is actually detrimental to the prediction performance of the model. The model using the unsorted, unscaled CM reaches a prediction MAE of 1.4 meV with 25 000 training examples.



(a) sorted CM

(b) unsorted CM



(c)

Figure 4.4.: A model trained on the sorted Coulomb matrix (a) performs far worse than the model trained on the unsorted representation (b). The learning curve (c) reflects this as well. Scaling the elements of the unsorted CM to normal distributions results in a worse model than leaving them as-is.

4.1.3.4. Comparison to the FCHL19 Representation

Many of the recently developed representations are far more elaborate than the CM, and have been shown to give more accurate models, reducing the number of training examples to learn properties drastically. However, they come at a higher cost for generating the representation, and in some cases more costly kernel function evaluations.[89, 9, 86, 90, 91] A more sophisticated representation could improve the prediction performance of the coupling models. Among the many density-based representations developed in the recent years, the FCHL19 representation was developed for and extensively tested in KRR models. The reference implementation of FCHL19 and its corresponding kernel are freely available in the `qmlcode` program package[4] and therefore provided an interesting alternative to the coulomb matrix.

A first test with the FCHL19 representation showed errors around 2.8 meV for after training on just 1161 data points. This far exceeds the best CM model (unsorted, unscaled CM) until now, which needed 10 000 examples to reach similar errors lower than when using either the sorted or unsorted Coulomb matrices. Unfortunately, the cost for prediction using KRR models based on this representation far exceeded the computational cost of both the simpler CM-based model and the reference method, and was found to be impractical in comparison. One of the core ideas of the density/distribution based representations like FCHL is that the inclusion of some physically motivated structure in the representation can improve the performance of ML models by 'front-loading' part of the physics. The increased computational cost for the calculation of the representation is not a problem for most of the published applications of ML models using density or distribution based representations, as the prediction times of the models are usually compared to *ab initio* methods (usually DFT variants). However, in this application the ML model is trained to replace DFTB, which due to its semi-empirical nature is orders of magnitude faster than DFT. Here, the reduction in training set size that is achieved by using FCHL is insufficient to offset the comparatively exorbitant costs of the representation and kernel.

4.1.4. Comparison between Gaussian and Laplacian Kernels

I first tested the difference between using the Gaussian and Laplacian Kernels in combination with the sorted CM representation. Laplacian kernels had been shown to perform better on quantum-chemical reference data[94], but the implementation available in `scikit-learn` as of this writing is quite slow. As the learning curves in Figure 4.5 show, the errors with the Laplace kernel were indeed slightly lower, albeit not by a large margin. In the absence of large advantages of the Laplacian kernel and in the interest of efficiency during model training and hyperparameter optimization, I decided to use the Gaussian kernel for the other experiments.

[4] At the time of writing, this was only available in the `develop` branch of the project available on GitHub.

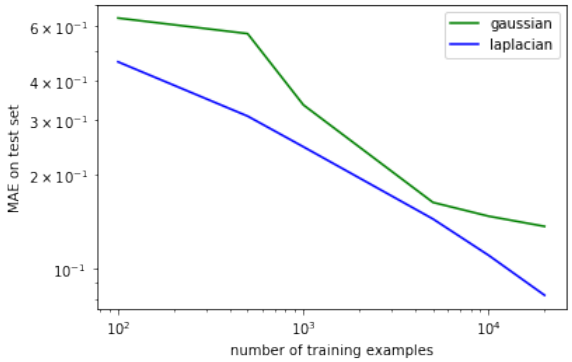


Figure 4.5.: Learning curves using the Gaussian (green) and the Laplacian (blue) kernels. The slope of the model with the Gaussian kernel changes as the kernel width was not optimized for every point in the learning curve, while the Laplace kernel seems to be less sensitive to this parameter.

λ_{site}	1×10^{-3}, 1×10^{-4}, 1×10^{-5}, 1×10^{-6}, 1×10^{-8} and 1×10^{-9}
σ_{site}	5, 10, 25 and 50
$\lambda_{\text{coupling}}$	1×10^{-2}, 1×10^{-3}, 1×10^{-4}, 1×10^{-5} and 1×10^{-6}
σ_{coupling}	5, 10, 15 and 20

Table 4.1.: Grid points sampled in hyperparameter optimization for site energy and coupling models for charge transfer.

4.2. Details on Training and Evaluation Procedures

4.2.1. Model Training and Hyperparameter Optimization

I trained models on charge and exciton data for each of the data sets `short`, `long` and `full` as well as for site energies using the unsorted Coulomb matrix representation as implemented in the `QML`[114] code package. For each data set, a model was trained on 100, 1000, 5000, 10 000 and 25 000 training examples with optimized hyperparameters obtained from a five-fold cross-validated grid search as implemented in the `scikit-learn`[117] python package. As the training set distribution is strongly skewed towards 0 by the frequency of pairs with negligible couplings, all data points which were equal to 0 were discarded from the training set. Because KRR model predictions go to 0 when the model is forced to extrapolate, these data points were nevertheless included in the test set to ensure that the models do indeed give the correct behavior. In a first evaluation step, I then tested each model on 60 000 data points from its respective data set, which were not used during training or hyperparameter optimization.

The values used for the hyperparameter optimization grids for charge transfer and exciton transfer are given in Table 4.1 and Table 4.2, respectively. The full results of the grid search for all data sets and training set sizes can be found in section A.2.

λ_{site}	1×10^{-3}, 1×10^{-4}, 1×10^{-5}, 1×10^{-6}, 1×10^{-8} and 1×10^{-9}
σ_{site}	5, 10, 15 and 20
λ_{Coulomb}	1×10^{-2}, 1×10^{-3}, ..., 1×10^{-8}, 1×10^{-9} and 5×10^{-10}
σ_{Coulomb}	2.5, 5, 7.5, 10, 15, 20, 25, 30, 40, 50, 60 and 70

Table 4.2.: Grid points sampled in hyperparameter optimization for excitation energy and coupling models for exciton transfer.

As evaluation metrics for each model, I used the mean absolute error (MAE), the coefficient of determination R^2, the maximum error, as well as the mean relative error (MRE) with respect to the average target value (Equation 3.5) to evaluate model performance.

Contrary to many other applications of ML methods to quantum-chemical problems, it is not possible to *a priori* formulate a quantitative threshold for when a model is good enough. Proper evaluation of the models thus requires using them in propagation simulations and comparing results for both time series of Hamiltonian elements and observables.

4.2.2. Evaluation in Charge and Exciton Transfer Simulations

A second anthracene crystal containing $40 \times 30 \times 14$ molecules along the crystal axes was constructed using the same process as described for training data generation in subsection 4.1.1. Simulations of charge and exciton transfer using both DFTB and the trained ML models were performed in this system. The calculation of the CM representation and KRR prediction were implemented directly in the code used for CT simulations using DFTB.

As QM zones, one-dimensional chains located in the middle of the crystal along the *a*- and *b*-direction of the crystal were chosen. The chain along the *a*-axis contained 36 fragments, while the one along the *b*-direction contained 28. No simulations were performed along the *c*-direction, as the couplings in this direction are very small, resulting in very low mobilities which would have necessitated overly long simulations to be reliably calculated. Additionally, the crystallographic and experimental *c*-direction differ by about 35° [118], and thus direct comparison between observations from simulations and experiments is difficult. Snapshots of the equilibrated system were taken in equidistant time intervals and used as starting structures for the ensemble of simulations of charge and exciton transfer.

The mean-field Ehrenfest [55, 56](MFE) and Boltzmann-corrected fewest switches surface hopping [60] (BC-FSSH) methods were used for non-adiabatic dynamics as described in section 2.4. Time steps of 1.0 fs were used for the MFE simulations, while for the BC-FSSH simulations the time step was 0.1 fs. The hole/exciton wave function was initially localized on the first molecule, $\Psi(0) = \phi_1(0)$ and the TDSE was integrated numerically with the fourth-order Runge-Kutta algorithm with an integration time step of 0.01 fs.

Observables were obtained from averages over swarms of 100 (MFE) and 500 (BC-FSSH) trajectories, simulated for 1 and 5 ps, respectively. The propagation used a local version of GROMACS 4.6 [119], wherein both the propagation methods and DFTB had already been implemented as a part of previously published work[69, 36, 37]. Coulomb matrix and

Gaussian kernel calculation as well as the final prediction step for the ML model were added by W. Xie and me.

The KRR models do not support easy calculation of gradients of the Hamiltonian elements w.r.t. atomic coordinates, which are necessary to properly describe the relaxation of fragments in response to a change in occupation. Therefore, the implicit relaxation scheme introduced in subsection 2.4.2 was used, regardless of whether DFTB or the KRR model was used to calculate the Hamiltonian elements. The reorganization energy used for charge transfer simulations was calculated using DFTB as 0.084 eV [36]. For exciton transfer, the reorganization energy was obtained from LC-TD-DFTB as 0.563 eV, which is in good agreement with previously published CC2 calculations [120].

Hole mobility was calculated using the Einstein-Smoluchowski equation as described in section 2.4.

4.3. Results of Model Evaluation

Overall, I trained $2 \times 4 \times 5 = 40$ models (site energy+three coupling data sets for CT and ET each, with five training set sizes). In the interest of brevity, I only present the results for the training set sizes of 100, 1000 and 25 000 here. I also only discuss in detail the coupling models trained on the `short` data set, as these were the only models used in the propagation simulations later on. The full results for all models can be found in section A.2.

4.3.1. Evaluation of Trained Models on Held-Out Data

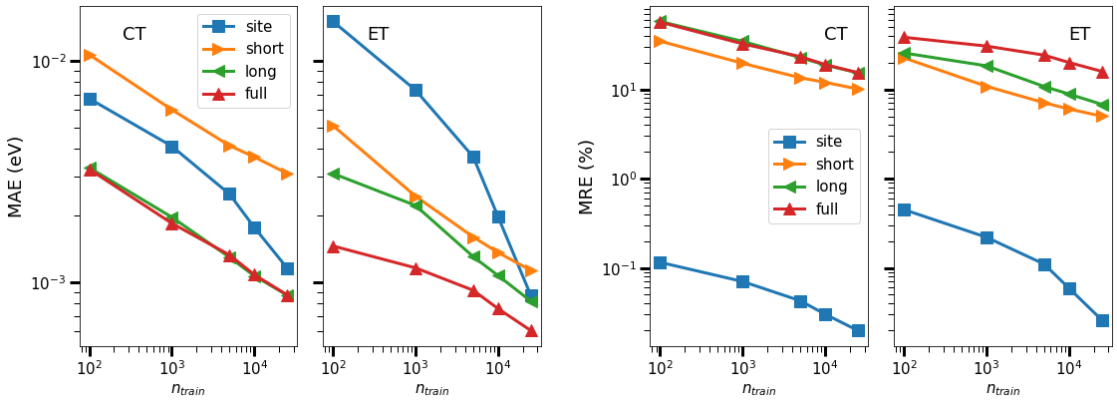
All models were able to learn their data sets and systematically improve with increasing training set size, as demonstrated in Figure 4.6. For the site energies, it is evident that the relative errors are far lower than for the couplings, indicating that the models have achieved better fits.

4.3.1.1. Evaluation of Charge Transfer Models

The mean and relative errors on the test sets are summarized in Table 4.3. Figure 4.7a–c shows the agreement between prediction and reference along the learning curve for the site energies and the `short` data set.

The site energies gave very good results, with errors reaching 1.2 meV at a training set size of 25 000, and a mean relative error below 1 % at all training sizes. For all training set sizes, there are few outliers, with maximum errors decreasing from 45 meV to 14 meV as the training set size grows. The standard deviation of the distribution of unsigned prediction errors decreases from 8.6 meV to 1.5 meV. That very small training set sizes are already sufficient to obtain low relative prediction errors is beneficial for the propagation, as the computational cost of the prediction increases linearly with the size of the training set.

At a training set size of 1000, the model for the couplings trained on the `short` data set reached a MAE of 6.0 meV, saturating at 3.1 meV for 25 000 training examples. Comparison



(a) (b)

Figure 4.6.: Learning curves showing the decrease of test set error of models trained on site energies and the three coupling data sets for both charge transfer (CT) and exciton transfer (ET). a) shows the absolute errors in meV, while b) uses the relative error defined in Equation 3.5.

	MAE in meV			max. error in meV			MRE (Equation 3.5)		
	100	1000	25000	100	1000	25000	100	1000	25000
site energy	6.75	4.12	1.16	45.8	36.3	14.1	0.1 %	0.07 %	0.02 %
`short`	10.7	6.02	3.10	124	74.4	66.7	35 %	20 %	10 %
`long`	3.29	1.97	0.87	120	108	62.9	58 %	34 %	15 %
`full`	3.34	1.85	0.88	129	94.3	71.0	57 %	32 %	15 %

Table 4.3.: Mean absolute and maximum errors in meV and mean relative errors for charge transfer models at 100, 1000 and 25000 training examples.

of the errors on the three data sets confirms that the `long` and `full` data sets are equivalent, as the results are almost identical. While the absolute errors for the `long` and `full` models are lower than those of the `short` model, this does not indicate that these models perform better. The average coupling in the `short` data set is 30.7 meV, much higher than the 5.7 meV average in the data sets with longer cutoffs. Thus, direct comparison of the MAE across the data sets is misleading, and mean relative errors must be compared instead. The MREs of the `short` model are lower at all training set sizes, indicating that this model has indeed learned its data set better.

4.3.1.2. Evaluation of Exciton Transfer Models

Results for excitation energies were similar to charge transfer site energies, with prediction errors of 15 meV to 0.9 meV along the learning curve and similar error distributions Figure 4.8 and Table 4.4). Again, the models showed few outliers even with small training set sizes, and relative errors are below 1 % for all training set sizes.

For the ET couplings, errors decreased from 5.1 meV to 1.1 meV along the learning curve for the `short` model. Again, the relative errors for the `long` and `full` models were slightly

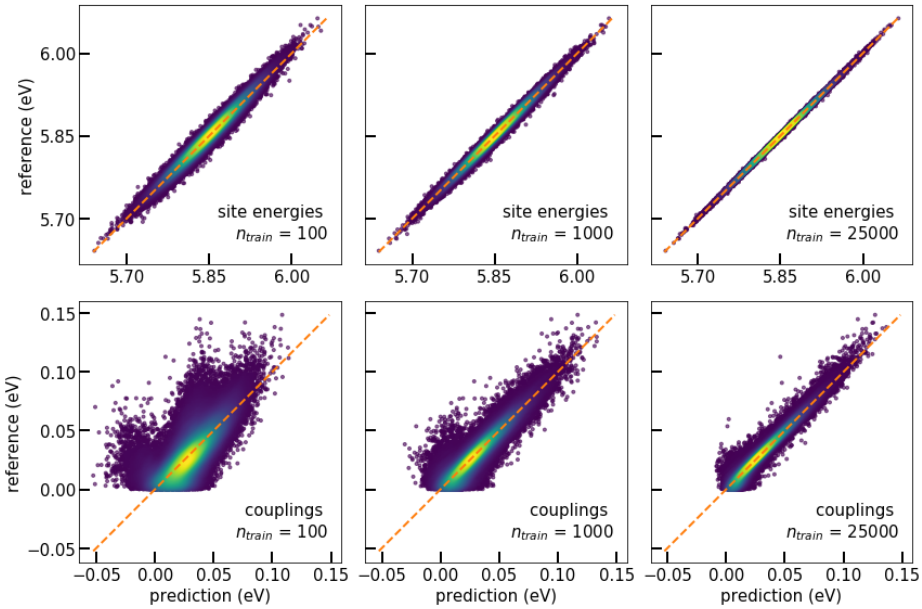


Figure 4.7.: Prediction on test set vs reference for CT site energies (top) and couplings (bottom) along the learning curve for the `short` data set.

	MAE in meV			max. error in meV			MRE (Equation 3.5)		
	100	1000	25000	100	1000	25000	100	1000	25000
site energy	15.1	7.37	0.87	110	69.3	11.7	0.45 %	0.22 %	0.03 %
`short`	5.09	2.45	1.12	35.4	35.6	32.8	23 %	11 %	5 %
`long`	3.10	2.22	0.82	34.0	26.3	23.6	26 %	18 %	7 %
`full`	1.46	1.16	0.60	38.5	31.2	25.3	38 %	30 %	16 %

Table 4.4.: Mean absolute and maximum errors in meV and mean relative errors for exciton transfer models at 100, 1000 and 25000 training examples.

higher. Maximum prediction errors were lower overall compared to those for the CT couplings, reaching 32.8 meV for 25 000 training examples on the `short` data set.

4.3.2. Time Evolution of Predicted Couplings

The next step in the evaluation was to apply the models to temporally consecutive structures and compare the predictions obtained from the models with the reference. The geometries of one arbitrarily chosen pair of first neighbors in a and b direction each were extracted from the equilibration trajectory of the crystal which was to be used for propagation. For each pair, I calculated predictions for CT and ET couplings on the geometries along the 0.6 ps time series. The results for the `short` models for both CT and ET are shown in Figure 4.9. As expected, the use of a smooth kernel and the unsorted CM results in generally smooth predictions. Compared to the DFTB curves, there is some noise in the predictions, but no extreme outliers are visible in these trajectories. While this small sample does not exhaustively analyze the behavior of the models along long

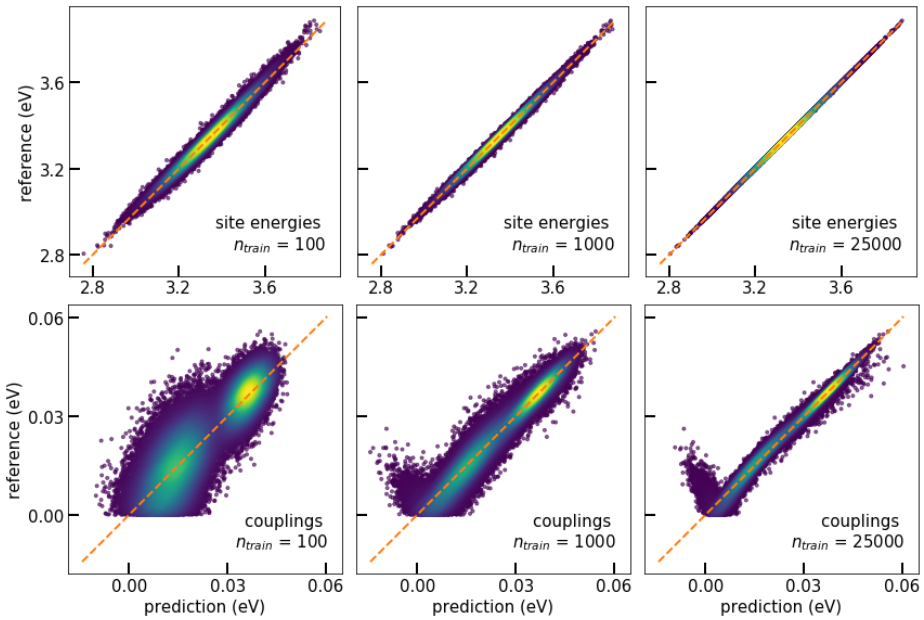


Figure 4.8.: Prediction on test set vs reference for ET site energies (top) and couplings (bottom) for the `short` model along the learning curve.

MD trajectories, it is a good indicator that the models could perform well enough to give reasonable results in propagation.

Evidently, the models start learning the couplings along the *a* axis properly only at a training set size of 1000. With only 100 training examples, there is not enough information to learn much more than the mean of the coupling in this direction. The predictions for the *b* direction are better at the lowest training set sizes, indicating that these couplings are predominantly learned at lower training set sizes. However, already the models trained on 1000 examples show close agreement with the reference curve for both crystal axes.

4.3.3. Application in Propagation Simulations

In the following, I present the application of the trained models for propagation simulations. First, I compare the computational costs of the simulations driven by DFTB and by the ML models. Then I show the results of the propagation simulations and compare observables obtained from DFTB and the ML models to reference data.

4.3.3.1. Performance Comparison to the DFTB Reference

To evaluate performance, I timed the duration of the Hamiltonian calculation in our test system and broke down this duration to obtain the time for one pairwise coupling calculation for both charge and exciton transfer. All times were recorded using a single core of an Intel Xeon CPU E5-2630 v4 @ 2.20GHz processor. Times for the CT Hamiltonian calculations using DFTB were obtained within the GROMACS code used for propagation, while times for the ET Hamiltonian were recorded using the method used to generate the training data using standalone LC-TD-DFTB as described in subsection 4.1.1. I estimated

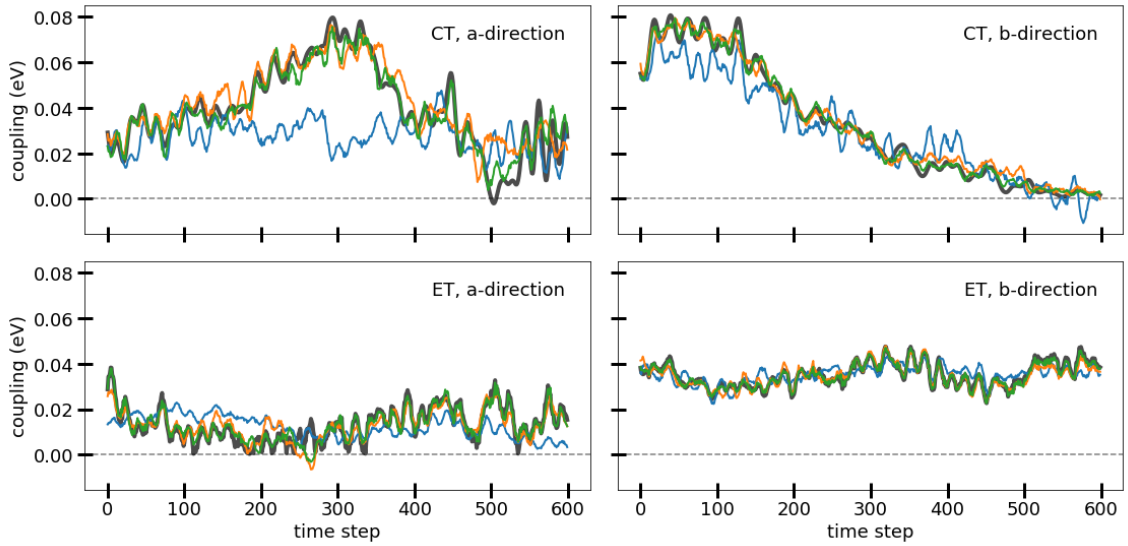


Figure 4.9.: Time evolution of the electronic and excitonic couplings between a pair of first neighbors in *a* and *b* directions. DFTB reference in black, colored lines show predictions of models trained on the `short` data set with training set sizes of 100 (blue), 1000 (orange) and 25000 (green).

times for one ML prediction within the python framework which was used for training. The python libraries used here use efficient implementations in Fortran (for the CM calculation) and C (Kernel calculation and prediction) for the individual steps necessary to obtain the prediction. I then summed times recorded for the individual steps to obtain the average time for one coupling prediction. While this is not directly comparable to the final computational costs in the simulation code, the sum of costs of the efficiently implemented individual steps gives a lower bound to the overall time per prediction which can be compared to the time necessary to obtain the same value using DFTB.

Calculation of the CT Hamiltonian using DFTB took on average 0.57 ms per anthracene pair, while for the ET Hamiltonian 55 ms per pair were necessary on the same processor. The timings for the different ML models varied strongly with the number of training examples, as the kernel calculation is the rate-limiting step. For site energies, a model trained on 1000 examples was always used as it gave excellent accuracy, while the number of training examples for the coupling model was varied.

As shown in Table 4.5, only machine learning models trained on 1000 or fewer examples could outperform DFTB for charge transfer couplings. This is a consequence of the simplicity of DFTB and the coarse-grained formalism – the costly calculations are done individually for the fragments, and assembling the Hamiltonian is then an almost trivial operation. In contrast, the ML model must calculate the comparatively costly kernel function with every training representation, as well as the interatomic distance matrix needed for the CM representation, and then needs one prediction from the site model per fragment and an additional (more costly) prediction per fragment pair to obtain couplings.

Overall, cost for both DFTB and the ML predictions scale equally, roughly with the square of the number of fragments if all pairwise couplings in a system are calculated. If a

	ML	DFTB	
n_{train}		CT	ET
100	3.1×10^{-4} s		
1000	3.8×10^{-4} s	5.7×10^{-4} s	5.5×10^{-2} s
25000	3.1×10^{-3} s		

Table 4.5.: Comparison of timings (in seconds) for the calculation of couplings per pair.

cutoff is introduced beyond which the couplings are not evaluated, this scaling converges towards linear behavior. However, for the ML model, the computationally costly part are the pairwise evaluations (as the models are larger both in training set and representation size), while for DFTB, the limiting step are the fragment-wise diagonalizations to obtain orbital coefficients. This diagonalization step in DFTB scales with the cube of the number of atoms per fragment. In the ML model, the calculation of the CM representation introduces a quadratic scaling with fragment size. Therefore, the larger the individual fragments are in number of atoms, the more the ML models' will win out compared to the cubic scaling of DFTB. This improved scaling can make simulations of systems with large fragments (such as Rubrene or Phthalocyanine) feasible, which are currently too costly to perform in DFTB. In contrast, the DFTB calculations necessary for exciton transfer are far more costly than those for charge transfer, so the ML model for anthracene is at least one order of magnitude faster even with the largest training set size tested here.

The ML models I present can also provide a significant gain in efficiency for simulations using the kinetic Monte Carlo model for hopping-like transfer. These simulations require an efficient sampling of average couplings in a system to calculate accurate transfer rates. For highly ordered structures such as crystalline anthracene, this is simple, as the average couplings for only one fragment pair along each crystal axis must be computed on a set of sampled structures (usually hundreds or thousands). In more disordered materials, this sampling is far more complex and costly, as the similarity of structures along the axes cannot be exploited, leading to a combinatorial explosion of computational effort. In such simulations, using an ML model instead of QM methods for the calculation of the couplings can thus result in a substantial decrease in computational cost.

4.3.3.2. Charge Transfer Simulations

Here, I present the application of the ML models to perform non-adiabatic dynamics of hole transfer along the *a*- and *b*-crystallographic axes of anthracene. Simulations were performed and analyzed by Philipp Dohmen, and I briefly summarize them here. Again, the results from the ML models trained on 100, 1000 and 25000 examples are compared to simulations using the DFTB reference, and the full results can be found in section A.2. All propagation simulations used the `short` model, as only nearest-neighbor couplings were calculated using both DFTB and the ML models.

The averaged time-dependent MSD obtained using the MFE method can be found in Figure 4.10a, and mobilities calculated from these simulations are given in Table 4.6. As expected, the model with 100 training data points gives a large error in the mobility along the *a*-direction, overestimating it by a factor of two compared to the DFTB reference. For the

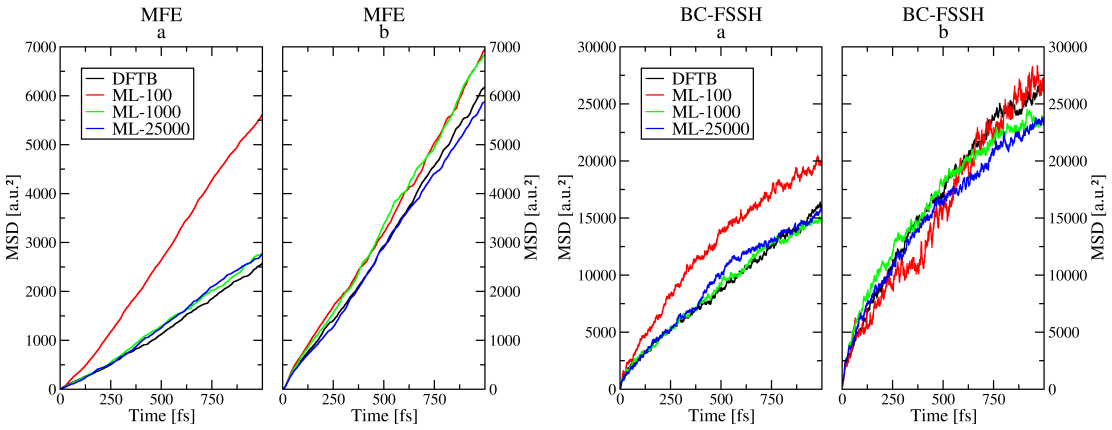


Figure 4.10.: Time evolution of the averaged MSD in *a*- and *b*-direction using the MFE (a) and BC-FSSH (b) methods for hole propagation with DFTB and ML models with various training-sizes.

b-axis, all models give reasonable results. The model with 1000 training examples already comes within 0.12 $cm^2\,V^{-1}\,s^{-1}$ (8.5 %) of the DFTB value and within 0.79 $cm^2\,V^{-1}\,s^{-1}$ (34 %) of the experiment. Further increasing the training set size does not improve this margin significantly, showing that spurious and small errors only have a low impact on the mobilities obtained after averaging over the swarm of simulations.

		Exp.[118]	DFTB	ML-100	ML-1000	ML-25000
MFE	*a*	1.14	1.41	3.20	1.53	1.58
	b	2.93	3.36	3.67	3.72	3.22
BC-FSSH	*a*	1.14	8.00	10.05	7.73	8.14
	b	2.93	13.32	14.30	11.08	11.02

Table 4.6.: Hole mobility in $cm^2\,V^{-1}\,s^{-1}$ as calculated from the averaged MSD in *a*- and *b*-direction using the MFE and BC-FSSH methods for hole propagation with DFTB and ML models with various training-sizes.

For simulations using the BC-FSSH method, the time-dependent MSD is shown in Figure 4.10b. The performance of the machine learned models compared to the DFTB reference is overall similar to the MFE simulations, including the overestimation of the mobility along the *a*-axis by the smallest model.

However, the absolute hole mobilities obtained from all BC-FSSH simulations differ strongly from the experimental values – even when DFTB was used to calculate the Hamiltonian. The overestimation of mobilities observed here is rooted in two approximations which were necessary because of the machine learning model:

First, in any SH method, atomic velocities must be adjusted after each hop to guarantee energy conservation [33], usually by scaling them in the direction of non-adiabatic coupling vectors (see section 2.4) obtained from the gradients of the Hamiltonian elements w.r.t. the atomic positions. As these cannot be obtained from the KRR model, we use the

Boltzmann-corrected version of the FSSH method, where the hopping probabilities are rescaled with a Boltzmann factor, as an approximation of the true correction. This leads to an overestimation of hole mobilities as demonstrated by Xie et al.[65].

Additionally, the relaxation of the fragments in response to the change in occupation is not fully taken into account (see Equation 2.18 and subsection 2.4.2). Again, to fully take this into account, the gradients of the Hamiltonian elements w.r.t. the atomic positions would be needed but cannot be obtained from the KRR model. To examine whether the explicit relaxation alone is sufficient for an accurate description, additional simulations were performed using DFTB with full relaxation of every fragment. This method decreases the MSD and slows down the corresponding mobility to 1.83 and 4.03 $cm^2\ V^{-1}\ s^{-1}$ for the *a*- and *b*-directions. These values are in much better agreement with the experimental values of 1.14 $cm^2\ V^{-1}\ s^{-1}$ and 2.93 $cm^2\ V^{-1}\ s^{-1}$, indicating that the relaxation scheme accounts for the bulk of the error. In the MFE simulation, this is not a problem as the charge delocalizes quickly and thus the forces on every fragment are smaller than in the BC-FSSH simulation where the charge remains localized.

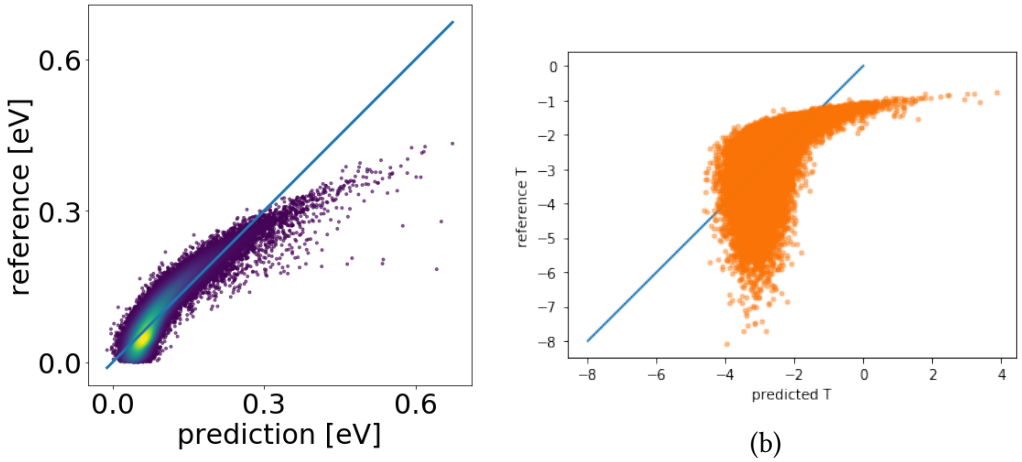
4.3.3.3. Exciton Transfer Simulations

For the exciton transfer case, only the BC-FSSH simulations were performed, as the diffusion is in the hopping regime and the MFE method would overestimate diffusion constants due to overly strong delocalization.[37] Additionally, no simulation code for explicit exciton transfer propagation using LC-TD-DFTB was available, and thus no DFTB reference could be obtained. Instead, the diffusion constants calculated using the ML models are compared to those obtained by solving the master equation (ME) with Marcus theory [121, 122] using kinetic Monte Carlo simulations. The time average over 5000 structures of one pair along each axis was used to compute the average couplings for the Marcus rate formulae. The diffusion constants obtained from this reference and the ML models are shown in Table 4.7.

The BC-FSSH method again shows an overestimation of the diffusion constants by a factor of 40 compared to ME results. Using the same arguments as in subsubsection 4.3.3.2, the implicit relaxation scheme and lack of non-adiabatic coupling vectors are likely the causes of the overestimation here as well, although lacking a DFTB implementation of propagation no detailed investigation can be made.

	ME	ML, BC-FSSH
a	7.4×10^{-9}	2.8×10^{-7}
b	4.2×10^{-8}	1.9×10^{-6}

Table 4.7.: Diffusion constants in $m^2\ s^{-1}$ as calculated from the averaged MSD in *a*- and *b*-direction using the ME and BC-FSSH with an ML model for Coulomb couplings with different methods for propagation.



(a) (b)

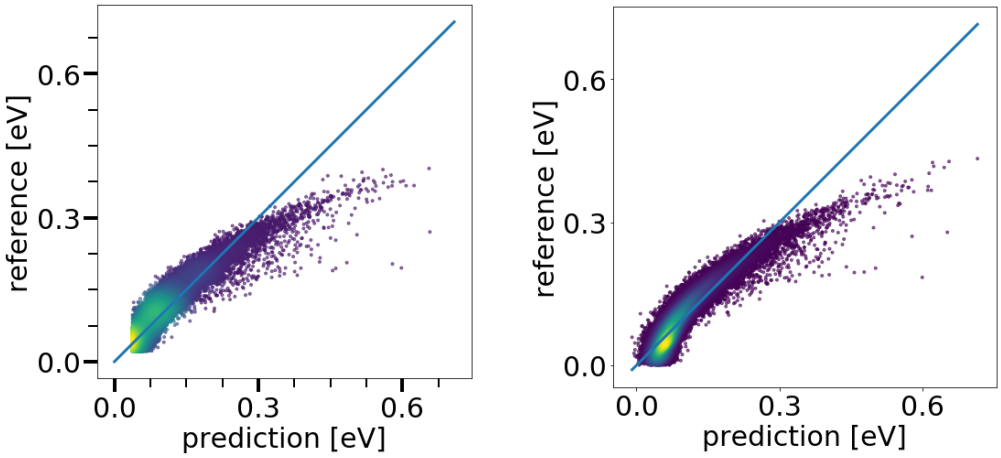
Figure 4.11.: Predictions on the test set for models trained on supermolecular ET couplings using a) the absolute value and b) the logarithm of the coupling as targets.

4.4. KRR Models for Supermolecular Exciton Transfer Couplings

Originally, I intended to also train KRR models for supermolecular ET couplings using same procedure as described for the Coulomb couplings in subsection 4.2.1 above. Supermolecular couplings were calculated for the pairwise geometries present in the `short` data set using LC-TD-DFTB according to Equation 2.21. In the following, I only show the results for the models trained on 25 000 training data points.

I attempted to train models with both the logarithm and the absolute value of the coupling as targets. The results are summarized in Figure 4.11, where it is evident that neither of the targets leads to satisfactory models. The model using the absolute value appears to have issues with larger couplings, so I also trained a model on the logarithm of the coupling in an attempt to make it easier for the model to bridge the different orders of magnitude. However, this made the model predictions even worse, so I used the absolute value as target for further experiments. Considering that these difficulties might result from the fast decay of the strength of supermolecular couplings with intermolecular distances, I also trained models using modified coulomb matrix representations, where the default $\frac{1}{r}$ dependency is changed to $\frac{1}{r^6}$ or even $\exp(-r)$. This shorter range of the CM representation terms could perhaps aid the model in learning the true distance dependency of the couplings. However, as Figure 4.12 shows, this did not significantly affect the results of the training.

My next attempt to determine what made the supermolecular data set so difficult to learn was to go back to the way they are calculated. Philipp Dohmen investigated this in more detail, and concluded that the approximation in Equation 2.21 breaks down when the difference between the monomer geometries becomes too large and the symmetry of



(a) (b)

Figure 4.12.: Predictions on the test set for models trained on supermolecular ET couplings using modified CM representations, with a) $\frac{1}{r^6}$ and b) $\exp(-r)$ scaling.

the system is reduced[**schieschke_Geometry**]. Therefore, the supermolecular couplings were re-calculated using the exact formula (Equation 2.20), and all structures which gave imaginary couplings were discarded from the data set. Additionally, to test whether the problem with imaginary couplings could be avoided, a linear approximation to the exact diagonalization result was also tested:

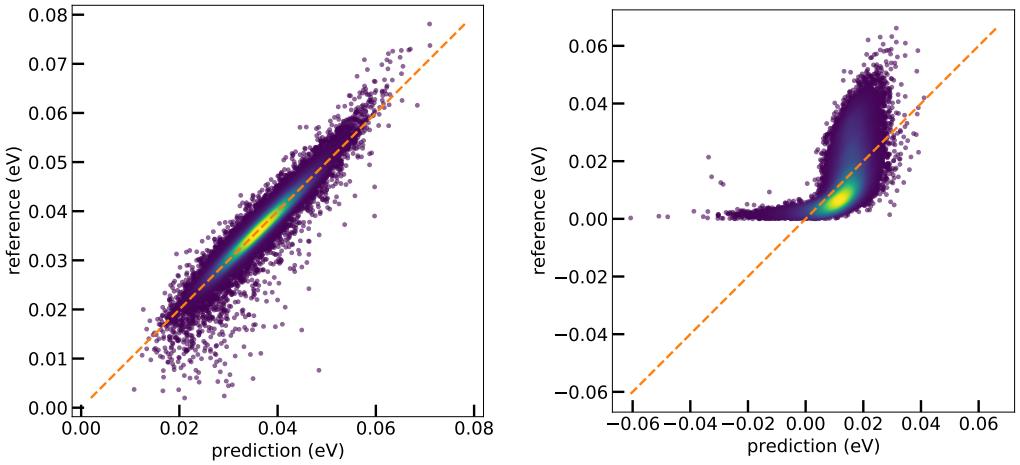
$$T_{ij} = \frac{1}{2}\left((E_2 - E_1) - \left(E_i - E_j\right)\right) \tag{4.1}$$

Figure 4.13 shows the results for two models trained on the exact and approximated supermolecular couplings. The model using the exact formula gave very good results, with a MAE of 1.05 meV and an R^2 score of 0.92. In contrast, the linear approximation did not seem to give a good model. This is an indication that Equation 4.1 is not a good approximation for the exact diagonalization result.

4.5. Application of KRR Models in Light-Harvesting Systems

As a final note, the models presented in this work were also applied by Sopheak Seng during his Bachelor's thesis for learning exciton transfer site energies and couplings in Bacteriochlorophylls (BChl) from the biological system LH2, and performed quite well in this context[123]. LH2 from *Rhodoblastus Acidophilus* is part of a larger complex involved in light-harvesting and contains 24 Bacteriochlorophyll molecules arranged in two rings referred to as B800 and B850 for their spectral properties (see Figure 4.14). It is these BChl molecules which are crucial to the absorption of light and the transport of the excitation to the reaction center in a different protein nearby.

Using the KRR models and the unsorted CM representation, the site energies of all BChl sub-units could be learned. For the couplings, the models were able to learn both



(a) (b)

Figure 4.13.: Predictions on the test set for models trained on supermolecular ET couplings using different formulae for the supermolecular coupling: a) exact solution (Equation 2.20) and b) linear approximation (Equation 4.1).

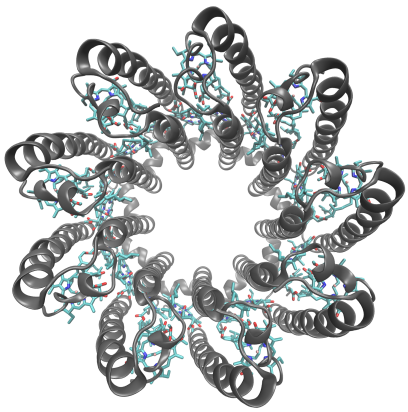


Figure 4.14.: Visual representation of the LH2 system. The Bacteriochlorophyll molecules are rendered in cyan. Image reproduced with permission from [123]

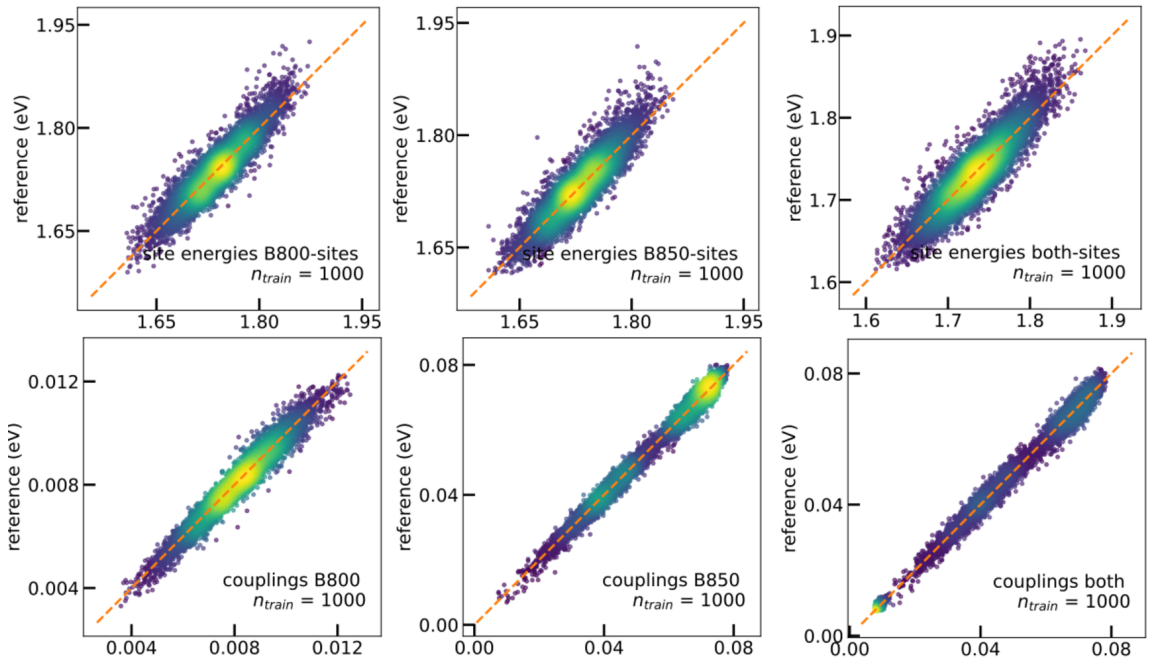


Figure 4.15.: Performance of ML models trained on 1000 data points for the site energies and excitonic couplings of the Bacteriochlorophyll rings in the light-harvesting system LH2. Image reproduced from work by S. Seng[123] with permission (modified)

the intra-ring couplings (subplots of Figure 4.15 labeled "B800" and "B850", respectively) and the couplings between all BChl units regardless of ring membership (subplots labeled "both").

This system is not only interesting for its biochemical properties, but also because each BChl molecule is about four times the size as an anthracene fragment. If ML-enabled simulations of exciton transfer in biological systems is to be feasible, the ML models should be able to work with representations of molecules of this size. Due to the N^2_{atoms} scaling of the Coulomb Matrix representation, this is not a foregone conclusion, as the larger a molecule gets, the more redundant the representation is compared to the actual molecular degrees of freedom and the longer the prediction using KRR takes. However, for molecules the size of BChl, the cubic scaling of DFTB raises the computational costs even higher, so direct simulations are infeasible. While the KRR models trained for the chlorophyll systems were far larger and slower than those for anthracene, they were indeed able to give good results. Indeed, if the representation is further reduced e.g. by only taking the intermolecular terms in the CM for the dimer as suggested by Wang et al.[100], both prediction time and accuracy can be even further improved.

4.6. Chapter Summary

In this chapter, I showed that it is possible to create compact and simple ML models which are able to provide Hamiltonians and closely match the observables given by the reference

method when used in direct charge and exciton transfer simulations. I investigated the effects of different design decisions for the model architecture and experimented with different representations to arrive at a suitable compromise between accuracy and speed. Additionally, I designed the models to be easily applicable to new systems by making the data generation, training and optimization procedures automatable, so that they can be integrated in existing simulation workflows.

Even a low amount of training data is sufficient to reproduce hole mobilities within 8.5 % of the DFTB reference and within 30 % of the experimental values in anthracene. This accuracy comes at a computational cost several orders of magnitude lower than *ab initio* methods, but only a little faster than the semiempirical DFTB method in this specific application. However, the models outperform DFTB significantly for exciton transfer, and will pull ahead even for charge transfer with increasing size of the individual fragments due to their more favorable scaling with fragment size.

The experiments on the supermolecular coupling data set showed that ML models can be used to sanity-check existing approximations: A bad approximation can confuse the patterns in the structure-property relationship in a manner that prevents ML models from learning.

The large overestimation of charge transfer mobilities when using the BC-FSSH propagation algorithm could be traced back to the problem of obtaining gradients from the ML model. These gradients enable the calculation of additional properties such as nonadiabatic coupling vectors and molecular relaxation effects to give reliable estimates of mobilities or diffusion constants in systems where the MFE propagator is unsuitable. In the following chapter, I will present my work on making these gradients obtainable from the machine learning method.

For an application to biological systems like light-harvesting complexes [53], the influence of an electrostatic environment on electronic structure properties is essential. To this end, the representation can be modified and additional interaction terms such as those in Ref. [98] can be added.

While the ML models presented here do not generalize across chemical space, the use of a fast but accurate semi-empirical model as reference, the low requirements on the number of training data points and the automatable training procedure enable the quick and easy training of an ML model for any specific system. This training step could be included in the setup of a multi-scale simulation approach, where the machine learning models trained using structures obtained during equilibration can then directly be used for propagation of charges or excitons.

5. Neural Network Models for Nonadiabatic Coupling Vectors and Relaxation

In the previous chapter, I presented simple Kernel Ridge Regression (KRR) based models which could drive simulations for charge and exciton transfer. With a very small training set size the models could already reproduce hole mobilities obtained with DFTB when used together with the mean-field Ehrenfest method for propagation. However, the KRR models imposed some harsh limitations on the applicability of the method, as it became evident that for a proper description of hopping-like transfer, occupation-dependent forces and nonadiabatic coupling vectors are necessary. Both of these properties can be calculated from the gradients of the diagonal and off-diagonal Hamiltonian elements (i.e. site energies and couplings) with respect to the atomic positions. In the following, I will explain why these properties are important and present neural network models which can predict the Hamiltonian elements as well as their gradients.

First, the gradients of the diagonal elements of the Hamiltonian provide the forces used for the relaxation of a site's geometry in response to a change in occupation. As described in subsection 2.4.2, we approximated this process implicitly by artificially reducing the site energy of the charge-carrying fragment by a specific amount. This 'reorganization energy' was obtained *a priori* from quantum chemical calculations[37]. However, this artificial change in energy does not affect the fragment's geometry in any way, so cannot truly capture the process of relaxation. As Figure 5.1 shows, using this approximation in FSSH simulations leads to large errors, even when DFTB and not the ML model is used to calculate site energies and couplings. This problem did not occur in MFE simulations, as the charge delocalizes strongly across the fragments, spreading this effect out over multiple sites. Second, as the charge hops from one potential energy surface to another, the potential energy of the system changes instantly. As total energy must be conserved, the kinetic energy of the system must be adjusted to match. Usually, this is done by re-scaling the velocities of the atoms in the system[124]. Scaling factors can be obtained from the Boltzmann-distribution[63], but a more precise method is to use nonadiabatic coupling vectors (NACVs)[125, 124], which can in turn be calculated from the derivatives of the off-diagonal Hamiltonian elements. Further details can be found in section 2.4.

Unfortunately, obtaining the gradients of a KRR model w.r.t. its input features is cumbersome. While the model is smooth and continuous when the CM representation is not sorted, the derivatives of the model must be calculated analytically (or approximated numerically), which is computationally costly and can limit which representations, kernel functions or training set sizes can to be used. To my best knowledge, at the time of writing

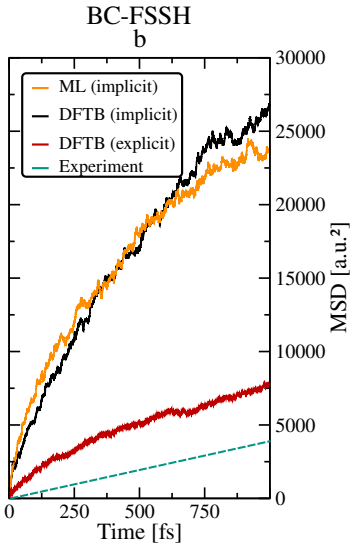


Figure 5.1.: Propagation along the b crystal axis in an anthracene crystal using the Boltzmann-corrected Fewest Switches Surface Hopping (BC-FSSH) propagator. Both the ML result (the model trained on the 1000 data points from the `short` data set in subsection 4.2.1) and the result from DFTB using implicit relaxation are far off from the DFTB result with full relaxation, which is closer to the experimental value[118].

the only publicly available implementation of KRR gradients is included in the QML code package [114] but requires using the slow FCHL19 representation.

Here, moving from a KRR model to a neural network architecture can help, as the commonly used NN frameworks (e.g. Tensorflow[126] or Theano[127]) have efficient automatic differentiation routines built-in. Furthermore, the computational effort needed to get a prediction from an NN only depends on the network architecture. For KRR models, prediction time scales linearly with the number of training data points that the model was trained on, limiting the training set size for all practical applications to a few thousand data points. Because DFTB is so computationally efficient (see subsubsection 4.3.3.1), this scaling severely restricts the usual approach of systematically improving the ML model by increasing training set size. A good sampling of conformational space is especially important when ML forces are used to change the system geometry, as bad force predictions can lead to catastrophic failures.

Previous work on learning non-adiabatic coupling vectors has focused on their application in excited-state dynamics [128, 129, 130, 131] and have used reference data from costly ab initio calculations. The resulting models work by learning the energy and its gradients for the ground and relevant excited states. As the reference values for every state are obtained from an individual calculation, a phase correction scheme is necessary to track and fix the absolute phase of the wave functions and obtain consistent signs for all values involved. In the context of the simulation formalism used in this work, this phase correction can be avoided, and only the properties of one state must be calculated.

In the following sections, I describe the design, training, optimization and application of a neural network model for the Hamiltonian elements for charge transfer propagation

and their derivatives w.r.t. atomic positions. As a test system, I again use the anthracene crystal to allow direct comparison of the new models with the KRR models presented in the previous chapter. For the NN models, I only use charge transfer Hamiltonians, as the formalism and implementation to calculate gradients and NACVs for exciton transfer using DFTB is not ready yet.

All training data calculations for the models in this chapter were performed by Weiwei Xie, while the propagation simulations were done by Philipp Dohmen.

5.1. Requirements and Fundamental Design Decisions

Several of the points discussed in section 4.1 apply here as well: Models should be as computationally efficient as possible while capturing the geometry-dependence of Hamiltonian elements, this time also including the gradients. In order to obtain gradients usable in MD simulations, the models must be continuous and smooth, and predicted energies and gradients must be consistent. Model training should use geometries sampled from MD simulations and be as automatable as possible in order to slot neatly into the simulation workflow for obtaining mobilities. For the same reason, no generalization across chemical space is necessary.

The decision to train separate models for the site energy and coupling elements motivated by the different representation sizes and scales of the targets needs to be reevaluated. On the one hand, an NN model has a lot more flexibility to handle these two factors compared to the rigid KRR models and could be set up to take as input a dimer geometry and return both site energies and the coupling (as well as the respective gradients). By scaling the targets separately to normal distributions with a mean of 0 and a standard deviation of 1, the scale separation is made irrelevant. Obtaining both site energies and the coupling for a pair of fragments with one network call may sound efficient, but as every fragment is contained in multiple pairs, multiple predictions of every site energy are made. This redundant calculation motivates the decision to make two separate models again – one takes the monomer structure and gives the site energies and their gradients, while a second model takes dimers, predicting couplings and their gradients.

In order to facilitate integration into GROMACS, the models must be as self-contained as possible. A clear definition of the interface between GROMACS and the models allows experiments with the architecture, representation, and other model specifics without modifications of the simulation code itself. Optimally, the model should require only the coordinates of the fragments/pairs and return the predicted couplings and gradients. Scaling or preprocessing steps should be part of the model if possible. For the representation, this is especially important, as not only the function for the calculation of the representation but also the calculation of its derivatives must be implemented. Implementing the representation as a layer of the NN model is therefore crucial for the flexibility of the approach. For scaling transformations of coordinates and model outputs, this is less crucial, as the simple multiplication with a scalar value is easy, efficient, and is independent of the functional model architecture.

As the gradients obtained from the model will be converted to forces applied to the system, it is especially important for force predictions to be as reliable as possible. Due to

the feedback of the forces to the geometries, a bad force prediction can push the atoms into conformations which are out of the area of the PES sampled during training. There, the model must extrapolate and can give erratic predictions for the forces, which can lead to even more distorted geometries. These errors can add up to not only single physically nonsensical geometries, but lead to catastrophic failures of the simulation by pushing atoms into each other.

One way to avoid this issue would be to quantify the uncertainty in the prediction, which allows falling back to DFTB whenever necessary or using active learning to craft the training set to be minimally redundant yet maximally representative. However, most techniques for uncertainty estimation in neural network models are formulated for classification problems or come at a large added computational cost (e.g. by training an ensemble of models). Because DFTB is so fast, this approach runs the risk of slowing down the propagation for charge transfer instead of speeding it up. Therefore, I decided to forgo uncertainty estimation for the time being in favor of training as fast a model as possible and reconsider that decision once a speed comparison between DFTB and the NN model is available.

Instead, to make the gradients as stable as possible I attempted to minimize the probability that extrapolation is needed during the course of an MD simulation by extensively sampling the conformational space, ensuring the inclusion of 'exotic' geometries in the training set. I describe this approach in detail in section 5.2, and evaluate whether the models obtained this way are sufficiently stable in subsection 5.5.2.

As far as the specific structure of the neural network is concerned, simple fully-connected networks with a nonlinear activation have been previously used to predict site/excitation energies or couplings[96, 93]. Recently, Li et al.[132] have presented an NN architecture specifically for photodynamics simulations of organic molecules. Published in the `pyNNsMD` package, it allows predicting energies and gradients of multiple electronic states, uses a simple representation whose calculation is integrated into the model, and is implemented using Tensorflow. It is very well suited to the problem as posed in this work, and forms the basis for the models in this chapter.

5.2. Generation of Data Sets for Training and Validation

All training data calculations in this chapter were performed by Weiwei Xie. Training data points for both diagonal and off-diagonal elements and their respective gradients were obtained from an MD simulation of the anthracene similarly to how the data for the KRR models was generated (subsection 4.1.1): An anthracene crystal supercell was constructed, containing $30 \times 15 \times 6$ molecules along the crystal axes (a, b and c), respectively. Force field parameters were derived from the general AMBER force field (GAFF) [102, 103]. Atomic charges were generated from restrained fitting on the electrostatic potential (RESP) [104, 105], calculated at HF/6-31G* [106, 107] using Gaussian09 [108]. All MD simulations were performed with the GROMACS 5.0.4 software package [109, 110]. After an initial energy minimization the temperature was equilibrated for 1 ns with the Nose-Hoover thermostat [112]. Productive MD simulations were run for 10 ns with a time step of 2 fs, in which structures were saved every 5000 steps. A subset of 10 anthracene molecules from the

supercell was used for obtaining training data. Hamiltonian elements and their derivatives with respect to atomic coordinates were calculated for every data point with the highest occupied molecular orbital (HOMO) as frontier orbital using non-self-consistent DFTB [1, 66, 67].

An initial data set referred to as `orig` was obtained from simulations equilibrated at 300 K. This data set contains 30 980 data points for the diagonal and off-diagonal elements each and was used for first tests of the model architecture and rough tests on hyperparameter effects, as well as tests of the implementation of NN prediction in GROMACS.

However, during the course of this work, the importance of thorough sampling of geometries became evident. When forces predicted by the ML method are used to change atomic positions in simulations, the adverse consequences of individual bad predictions can be catastrophic: If a geometry is encountered for which the model gives a drastically bad prediction for the energies, the predicted forces can become quite large and displace the geometry of the molecule to a point in the PES even farther outside the training set, where the prediction in the next step will then give even worse predictions. Therefore, strong enough outliers can cause the ML model to become stuck in a vicious cycle of deteriorating predictions until the simulation crashes. Avoiding outliers requires careful sampling of the conformational space of the fragments and pairs in the crystal to make sure that no thermally accessible geometries are far outside the training set. As this project progressed, the data sets I used for training evolved as well to improve this sampling, reduce outliers and minimize the probabilities of simulations crashing.

One approach to improve MD-based sampling for ML purposes is to sample the training geometries from a simulation performed at a temperature far higher than the temperatures for which the model would be asked to provide forces (e.g. sample training data at 1000 K and use the model for simulating at 300 K). This approach has previously been shown to work well for ML-driven molecular dynamics simulations of silicon crystals [133]. By obtaining the structures from a simulation with very high temperature, even the parts of the PES which are rarely seen in the lower-temperature MD are sampled for the training data set. In the anthracene crystal, however, the feasibility of this approach appears limited by the phase transition points of anthracene: Outside of MD simulations, anthracene melts at 489 K and boils at 614 K[83]. However, these phase transitions do not occur noticeably at the times scales along which the geometries are sampled. The geometries seen in high temperature MD simulations beyond the melting point (and arguably even a bit beyond the boiling point) are therefore obtained from sampling a similar but expanded conformational space compared to that seen at lower temperatures.

Using this approach, a second data set (henceforth called `mixed_temperatures`) was created by obtaining structures from simulations of the crystal equilibrated at 300 K and 500 K. For each temperature, 10 000 data points for diagonal elements and 45 000 data points for the off-diagonal elements were included in the `mixed_temperatures` data set. This data set was used during the tests of the different hyperparameter optimization methods.

There is another effect which must be taken into account for the minimization of outliers: The structures for the training data were previously sampled from an MD simulation of the uncharged sites. I will illustrate why this is a problem using the diagonal elements of the Hamiltonian (i.e. site energies), but the same argument applies for the off-diagonal

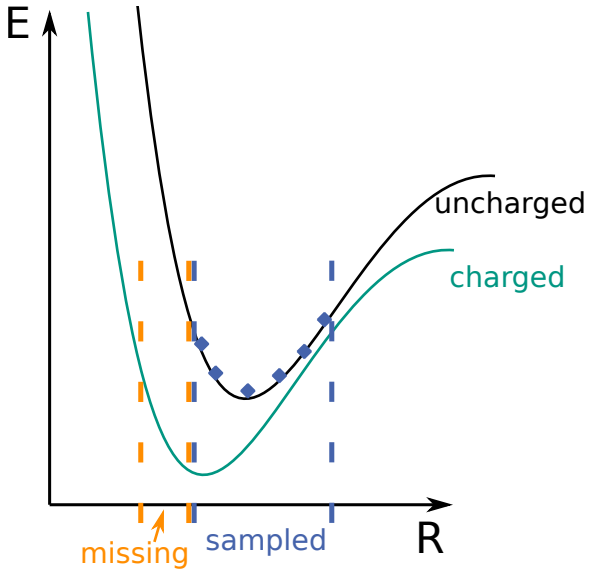


Figure 5.2.: Schematic PES of uncharged molecule and molecule occupied by a hole. Sampling of training data happens on the uncharged PES, but following the learned gradients of the charged PES towards its minimum can lead to geometries unknown to the model yet thermally accessible (orange interval).

terms. In the procedure described above, the site energy and gradient information is sampled around the minimum of the PES of the uncharged molecule, as no charge is present on any of the fragments used in the training sets `orig` or `mixed_temperatures`. However, in charge transfer simulations following the forces calculated from the gradients on an occupied fragment leads into the minimum of the PES of the charged molecule[1], which may be notably different from the minimum of the uncharged state. The bigger the difference between the PESs of charged and uncharged states, the more structures which are thermally accessible for the charged fragment lie outside the originally sampled geometry space. This effect is schematically illustrated in Figure 5.2 for a diatomic molecule. The conformational space sampled for training data generation using the uncharged state at a given temperature is shown in blue. As the teal PES of the charged fragment is shifted to shorter distances, there is a part of the charged PES which is fairly close to the minimum but outside the range sampled in the training set. For geometries with a bond length from the orange interval, any model trained on data from the blue interval must extrapolate, and bad force predictions here can move the structure even further outside the model's area of competence. These effects can compound, leading to crashes in the simulation as atoms are pushed together or restraints placed on the fragment geometries are exceeded.

In order to also sample this area of the conformational space of the anthracene system, the training set must be augmented by including geometries of charged fragments sampled from MD simulations where the fragment occupations were kept constant and no movement of the charge was possible. This data set I will refer to as `charged_uncharged`. These simulations were performed exclusively at 500 K to increase the structural variance

[1] This a rephrasing of the goal to reproduce the relaxation of the fragments on occupation changes

in the training data set. To obtain diverse structures of charged fragments, ten 0.63 ns simulations were performed with a hole fixed on each of the ten fragments from the same subset used above. This gave 42 000 data points for the diagonal elements and 63 000 data points for the off-diagonals sampled from the PES of charged fragments. These data points were combined with data for the uncharged state obtained in the same manner as in the `mixed_temperatures` data set, adding 47 800 data points for the diagonal and 45 000 for off-diagonal elements. The total amount of data available in the `charged_uncharged` set was therefore 89 800 and 108 000 for diagonal and off-diagonal elements, respectively.

5.3. Architecture of Neural Network Model

Here, I describe the model architecture presented by Li et al.[132] and the modifications made to the network by me or Patrick Reiser for application to DFTB Hamiltonians for anthracene. In the aforementioned work, two different kinds of networks are used: One model is trained to predict the energies and forces of both ground and excited states of the molecule. For this model, the loss function takes into account the errors on both energies and gradients. The second network predicts NACVs as the derivatives of a physically meaningless potential by using the same network architecture as for energies and forces, but only including the gradient term in the loss function.

While in this work, the goal is to obtain NACVs from the NN prediction, the approximations made in the DFTB reference method for propagation allow using only the energy-gradient network: The diagonal elements of the DFTB Hamiltonian for charge transfer are the energies of the charged fragment's frontier orbital, so the energy in the uncharged state is not needed. The gradients of the Hamiltonian elements are not the NACVs themselves, but the forces that would apply were the involved fragments occupied by the charge carrier which are then used to calculate NACVs with little added cost. Therefore, using one energy+gradient network for the diagonal elements and another for the off-diagonals is sufficient. Each of these networks functions like described by Li et al.[132], but is trained to predict the Hamiltonian elements and their gradients instead of the energies and forces of several electronic states.

The networks use the spatial coordinates of either a single fragment or a fragment pair as inputs and transform them into a matrix of inverse distances between atoms. This representation fulfills all necessary invariances, is inexpensive to calculate and can easily be integrated in the network architecture for automatic gradient calculation. As the representation using inverse interatomic distances is already redundant compared to the degrees of freedom of the system and is also quite large for molecules as big as anthracene, I did not include angles or dihedrals. Instead, for the coupling network I experimented with further reducing the representation to only include the intermolecular distances between atoms. This reduction roughly halves the size of the representation without loss of information and has previously been shown to improve learning of electronic couplings [100]. For the monomeric representation, such a trivial reduction is not possible, but also not quite as necessary: Due to the N_{atoms}^2 scaling of the distance matrix, the monomer representation is one fourth the size of the dimer representation.

The individual entries of the representation are then scaled so that every entry resembles a normal distribution with mean 0 and variance 1 across all points in the training data set. This scaling is essential for the successful training of the neural network. The energy value (diagonal or off-diagonal Hamiltonian element) is also scaled in the same way, while for the gradients, care must be taken to keep consistency with the energy. Therefore, the gradient values are only scaled with the variance obtained from the energy distribution in the training set and not shifted at all.

The trainable part of the network consists of a stack of fully-connected layers using a leaky soft plus activation function

$$\texttt{leaky\_softplus}(v) = (1-\alpha)\cdot\log\left(\exp(v)+1\right)+\alpha\cdot v, \tag{5.1}$$

where α quantifies the leakiness of the slope (see Equation 3.17) and is fixed at its default value of 0.3. The final output layer uses a linear activation to give the Hamiltonian element. The number of hidden fully-connected layers as well as the number of neurons per layer are optimized during the hyperparameter search (see subsection 5.3.2 below).

The loss function includes the mean squared errors on both the energy and force predictions as proposed by Schütt et al.[134] shown in Equation 5.2.

$$L = \alpha\mathrm{MSE}\left(E^{\mathrm{ref}}, E^{\mathrm{pred}}\right) + \beta\mathrm{MSE}\left(F^{\mathrm{ref}}, F^{\mathrm{pred}}\right) \tag{5.2}$$

α and β are hyperparameters used to balance the impact of errors on energies with errors on force components.

A notable technical aspect of these networks is that they are implemented in Tensorflow, which provides bindings for working with neural network models in the C programming language. This is especially useful, as it means that the models can be loaded and used in GROMACS with little effort, and even extensions to active learning schemes would be feasible for the future.

5.3.1. Configuration of Model Training

In order to speed up model training, the features of the training data and their gradients were calculated once and reused in every epoch.

I trained all models using the Adam optimizer [135]. While the learning rate was included in the hyperparameter search, other optimizer parameters were kept constant at their default values.

All models received training and validation data in a 9:1 ratio, and the loss on the validation set was monitored in order to prevent the model from overfitting on the training set. The training process was stopped if the loss on the validation set did not improve for a certain number of epochs, and the model parameters restored to the point where that loss was minimal.

I included the type of regularization (L1 or L2) in the hyperparameter search space, with regularization strengths at their default values of 0.01.

5.3.2. Hyperparameter Optimization

In first tests of the NN architecture described above, it became evident that several parameters are crucial to the performance of the trained models. These included the number of hidden layers and neurons per layer, as well as the relative weights of the losses on the energy and the forces. Additional hyperparameters whose effects were not as transparent were the type of regularization and the starting learning rate. In order to sample the large search space, I experimented with several techniques for hyperparameter optimization. An initial rough exploration of hyperparameter combinations showed that especially the loss weights heavily impact the final score of the model, and best results were obtained with the energy:gradient losses at ratios as disparate as 5:1 – 1000:1. I was therefore able to reduce the search space a bit by fixing the gradient loss to a value of one, and only including the energy loss weight in the hyperparameter optimization.

In order to automatically optimize the hyperparameters, an optimization criterion is needed. Usually, the loss on the validation set or the R^2 coefficient can be used, but this is not as straightforward when the model has two outputs. The losses are not directly usable as a target, as they are scaled with the loss weight hyperparameters, which are part of the optimization process. While R^2 coefficients are not directly dependent on the hyperparameters in the same manner, the R^2 values for the energy and force outputs must still be combined into a single metric in order to be compatible with standard hyperparameter optimization algorithms. It is desirable to keep the behavior of the combined R^2 coefficient consistent with that of the individual R^2 values, so any combination should have an optimal value of 1, the value of 0 should correspond to the null model, and there should be no lower bound. Additionally, I want to favor models which perform equally well on site energies and forces so that the optimization does not converge to models which are excellent for one but terrible for the other. In a first attempt, I experimented with criteria which take the sum or arithmetic mean of the two R^2 coefficients and apply a penalty which scales with the difference of the coefficients, e.g.

$$R_{\text{merged}} = R^2_{\text{energy}} + R^2_{\text{force}} - \lambda |R^2_{\text{energy}} - R^2_{\text{force}}|. \tag{5.3}$$

However, this type of criterion introduces an additional hyper-hyperparameter λ to adjust the relative importance of the difference term to the original R^2 coefficients. To avoid this, I decided on another approach: The individual R^2 coefficients would be scaled using a nonlinear function which punished low R^2 values and final combined score is their arithmetic mean:

$$R_{\text{merged}} = \frac{R^{\text{scaled}}_{\text{energy}} + R^{\text{scaled}}_{\text{force}}}{2} \tag{5.4}$$

I experimented with a quadratic and an exponential scaling for the individual R^2 coefficients, as it is *a priori* unclear which scaling is better.

$$R^{\text{scaled}}_{\text{quadratic}} = -(R^2 - 1)^2 + 1 \tag{5.5}$$

$$R^{\text{scaled}}_{\text{exponential}} = -\exp(-(R^2 - 1)) + 1 \tag{5.6}$$

As for the optimization algorithm, I had to find a balance between the computational cost of the algorithm and the coverage of the search space. A grid search approach has

the disadvantage that it must naively cover all points in the search space and does not converge on the most promising areas. Considering that an individual fit of the NNs on a training set size of 9000 took around two hours on a GTX 1080Ti GPU, a grid search would take an unreasonable amount of time.

Among the optimization algorithms which converge on promising parts of the search space, I first experimented with a genetic algorithm. While this algorithm was able to converge to satisfying results, the need to fully train every candidate model drove the computational costs of the search to an unacceptable level. Therefore, I moved on to use the Hyperband search (subsection 3.6.2) as implemented in the `keras-tuner` package. This algorithm has several parameters which can be adjusted to reduce the search time to around 3 h to 6 h despite the large search space.

5.4. First Generation of Models: Proof of Principle

Using the original training data set `orig` I performed a few initial experiments with the NN model. I did not systematically optimize hyperparameters at this point, but some exploration of the relative loss weight parameter was necessary to get the model to learn both the Hamiltonian elements and their gradients.

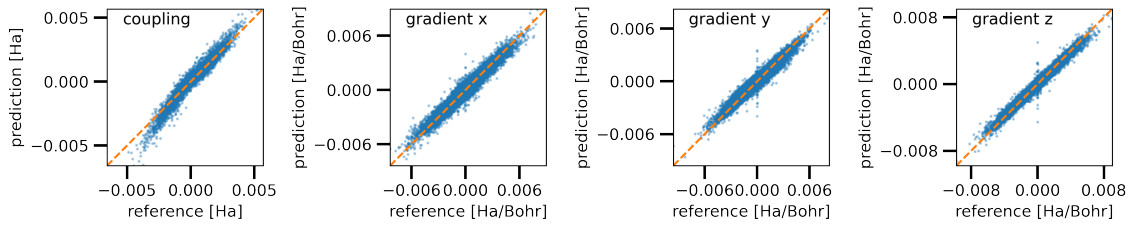
5.4.1. Models for Off-Diagonal Elements

For the off-diagonal elements, I used a training set size of 45 000 structures, with a validation set size of 5000. The hidden part of the network consisted of four layers á 100 neurons, with the loss of the energy weighing 5 times as much as the force loss. I used early stopping combined with a learning rate scheduler to periodically decrease the learning rate when there was no improvement for ten epochs and finally stop training before the model could overfit, resulting in learning rates of 10^{-3}, 10^{-4}, 10^{-5} and 10^{-6}. I trained two models in order to compare two variants for the representation – one model used the full upper diagonal of the distance matrix, while a second model used only its intermolecular block (see above).

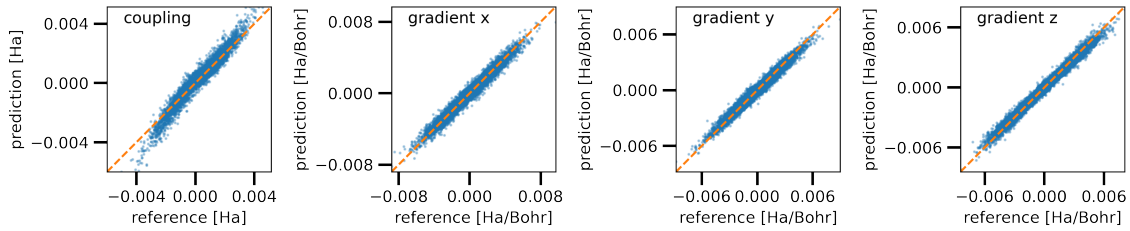
The model using all interatomic distances (`coupling-full`) converged to an MAE of $1.7 \times 10^{-4}\,E_\mathrm{h}$ (4.5 meV) on the couplings and $4.9 \times 10^{-5}\,E_\mathrm{h}\,{a_0}^{-1}$ (2.5×10^{-3} eV Å^{-1}) for the gradients. It reached R^2 scores of 0.92 for the couplings and 0.94 for the gradients. The performance of the trained model on 10 000 structures from the held-out test set is shown in Figure 5.3b.

When the reduced representation was used, the model reached a MAE of $1.6 \times 10^{-4}\,E_\mathrm{h}$ (4.4 meV) for the coupling and $4.3 \times 10^{-5}\,E_\mathrm{h}\,{a_0}^{-1}$ (2.2×10^{-3} eV Å^{-1}) for the gradients. The R^2 score for the coupling is 0.92 and for the gradients R^2 =0.96, so the model has learned both properties about equally well. The results for this model, which I will refer to as `coupling-inter`, are summarized in Figure 5.3b.

While these two models reached very similar levels of accuracy, there was a significant difference in the computational cost for the training of these two models although the convergence criteria for the early stopping method were identical. Using the full representation, the model needed 744 epochs to converge, while with the reduced representation,



(a) off-diagonal elements with full representation (`coupling-full`)



(b) off-diagonal elements with reduced representation (`coupling-inter`)

Figure 5.3.: Performance of two NN models trained to off-diagonal elements of the Hamiltonian and their gradients. Model a) uses the full upper triangular of the interatomic distance matrix as representation, while b) only uses its intermolecular block.

the model already gave excellent results after 210 epochs. The change of the training set MAEs on the couplings and gradients during the course of the training is shown in Figure 5.4 for both models.

As the reduced representation speeds up both the training and the prediction and has a slightly positive effect on the prediction quality, I only used this representation in further experiments.

5.4.2. Models for Diagonal Elements

I trained diagonal element models on 45 000 structures as well, with a validation set containing 5000 structures. Again, I used early stopping combined with a learning rate reduction (5×10^{-3}, 5×10^{-4}, 5×10^{-5}, 5×10^{-6} and 5×10^{-7}). For the diagonal elements, it was far more difficult to find good values for the hyperparameters to give good results for both energies and gradients.

At a loss-weight ratio of 1000:1 (Figure 5.5a) with a trainable network of four layers á 200 neurons, the energies were learned well, but the gradients showed errors where especially large gradients were overestimated. This model reached a site energy MAE of $3.7 \times 10^{-4}\ E_\mathrm{h}$ (10.1 meV) and a MAE of $7.4 \times 10^{-4}\ E_\mathrm{h}\ {a_0}^{-1}$ (3.8×10^{-3} eV $\mathrm{Å}^{-1}$) for the gradients on the test set. R^2 scores were quite good as well, 0.93 for the site energy and 0.97 for the gradients.

In contrast, a ratio of 500:1 (Figure 5.5b, referred to as `site`) and below and smaller network sizes (four layers á 30 neurons) resulted in almost perfect forces, but the site

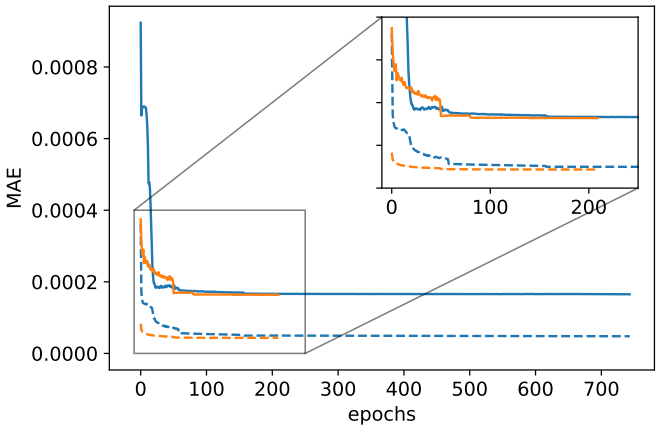
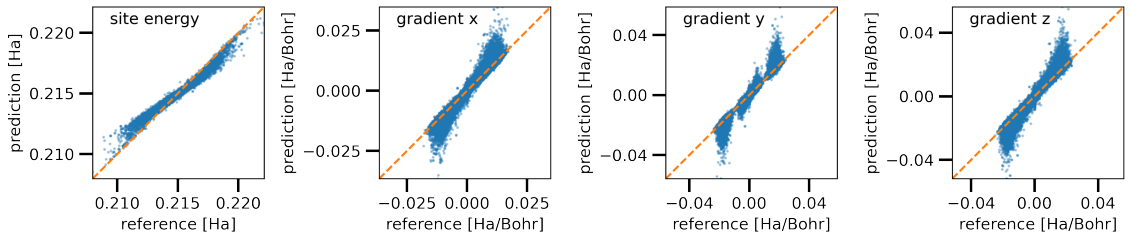
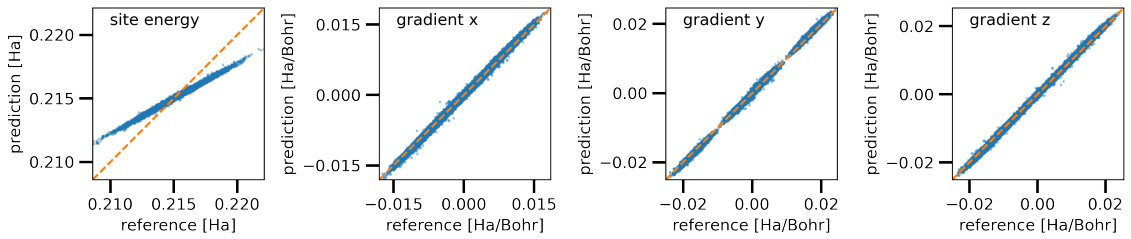


Figure 5.4.: Training set MAEs of couplings (solid line) and gradients (dashed line) for the `coupling-full` (blue) and `coupling-inter` (orange) models.



(a) diagonal elements with loss weights 1000:1



(b) diagonal elements (loss weights 500:1, `site`)

Figure 5.5.: Performance of two NN models trained to diagonal elements of the Hamiltonian and their gradients. The model in a) uses a 1000:1 ratio of the loss weights of energy and gradients and a large NN, while the model in b) uses a smaller network and a 500:1 ratio.

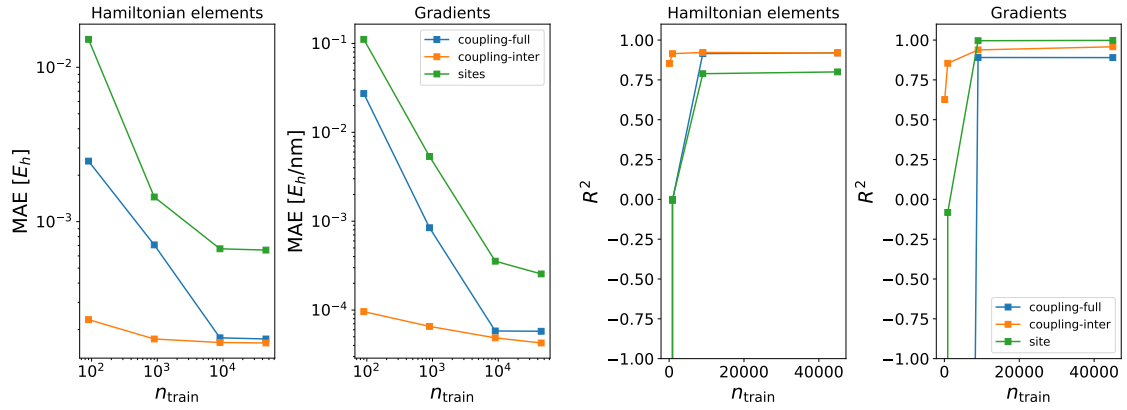


Figure 5.6.: MAE (a) and R^2 (b) as a function of training set size for the `coupling-full`, `coupling-inter`, and `site` models.

energies showed a systematic shift, so that large site energies were underestimated by the model. For the site energies, the MAE was $6.5 \times 10^{-4}\ E_\mathrm{h}$ (17.6 meV), with an R^2 score of 0.8, while the gradient MAE was $2.6 \times 10^{-4}\ E_\mathrm{h}\ {a_0}^{-1}$ (13.6×10^{-3} eV Å^{-1}), giving an R^2 score of 0.99.

As accurate gradients are crucial for obtaining a stable simulation, I preferentially used the diagonal element models with better forces for first tests of the GROMACS implementation. For more productive use, a model which gives good results for both site energies and gradients is required, which should be discoverable with a more extensive and systematic hyperparameter search.

5.4.3. Effects of Training Set Size on Model Accuracy

In order to evaluate the behavior of the `coupling-full`, `coupling-inter`, and `site` models as training set size increases, using the hyperparameters given above I trained five versions of each model at training set sizes of 90, 900, 9000 and 45 000. I used identical hyperparameters for each training set size because I wanted to avoid repetitively running the costly hyperparameter optimization scheme. In contrast to the kernel width σ in KRR methods, this NN model does not have any hyperparameters with such a tight coupling to the training set size. Figure 5.6a shows the mean absolute error and Figure 5.6b the R^2 score of every model evaluated on a 10 000 structure test set. Evidently, the `coupling-inter` model performs far better even at low training set sizes, while the model using the full representation needs more data to reach satisfactory results. This observation is in agreement to the analysis performed by Wang et al. [100]. All models improved as training size increased, a good indicator that they are systematically learning patterns in the training data set. However, models with a training set size below 10 000 data points were frequently completely unusable as the low R^2 scores show. This information is crucial for the hyperparameter search, as this gives a minimum amount of data necessary to reliably obtain working models.

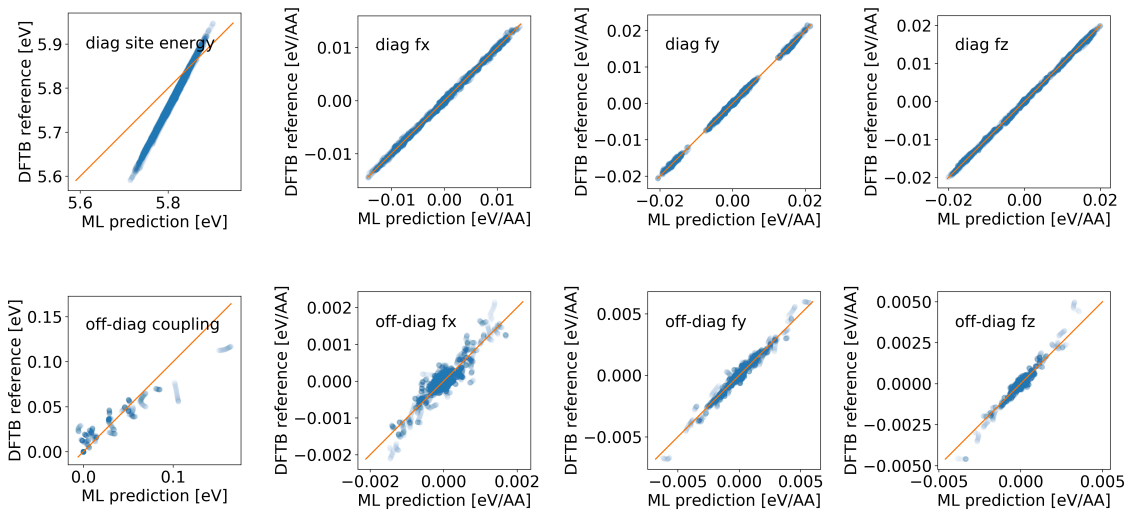
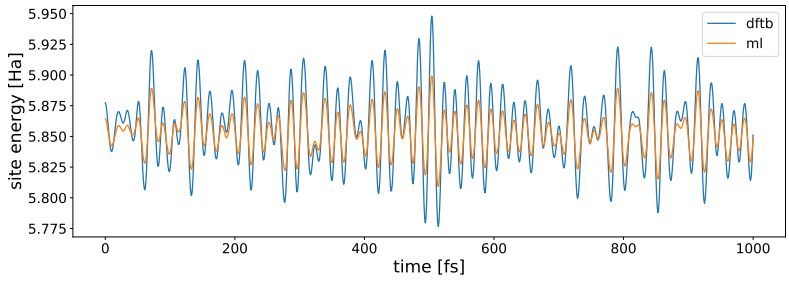


Figure 5.7.: Hamiltonian elements and their gradients as calculated by DFTB and the NN models. Upper row diagonal elements, lower row off-diagonal elements.

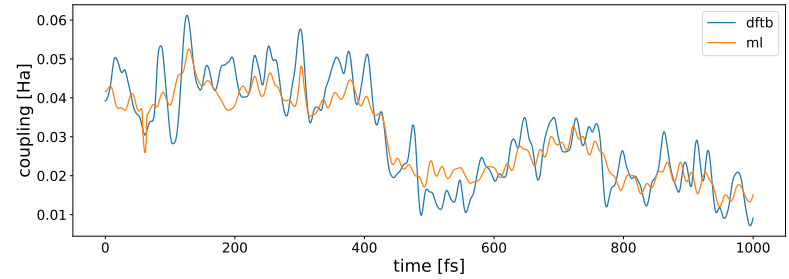
5.5. Implementation in GROMACS and Performance Considerations

I implemented the Hamiltonian prediction using the NN models in GROMACS using the Tensorflow C API and was able to load pre-trained models, pass the fragment and pair coordinates to the model and the predictions for Hamiltonian elements and gradients back to the propagation code. When the Hamiltonian is calculated using DFTB, the coupling is only evaluated if the there is an intermolecular pair of atoms which are closer together than the 20 a_0 cutoff of the DFTB SK files. I ensured that the ML method uses the same criteria to decide which pairs to calculate the coupling and gradients for to keep computational costs comparable.

I tested the implementation by performing a 0.1 ps (1000 MD steps) simulation without propagation in a system whose QM zone consists of a chain of 25 anthracene molecules along the *b* crystal axis using DFTB and then re-running the simulation to obtain the ML results on the same trajectory. As ML models, I used networks very similar to the off-diagonal model and 500:1 diagonal model presented above. The results are summarized in Figure 5.7. The systematic error in the site energy can be observed here as well, otherwise the results appear quite similar to the results outside GROMACS. A look at the time series for the couplings and site energies can highlight the effects of the prediction errors during the simulation. For the site energy, Figure 5.8a shows that the ML model favors the mean of the distribution and does not capture the variance of the site energies. In Figure 5.8b, it is evident that the coupling model reproduces the general trends of the coupling as the geometry changes over time, but cannot capture its small fluctuations. In chapter 4 it became evident that good values for the hole mobilities could indeed be obtained with models which did not capture these fluctuations, so the couplings predicted by this model



(a)



(b)

Figure 5.8.: Comparison between values calculated using DFTB (blue) and the NN (orange) along the trajectory of a simulation of an anthracene crystal. a) Site energies of a randomly chosen fragment, (b) coupling between a randomly chosen pair of neighboring fragments.

should be sufficient. However, the systematic issue with the site energy model must be resolved and the quality of the gradients tested before propagation simulations can be performed.

5.5.1. Effects of Outlier Predictions on Stability of MD Simulations

In order to evaluate if the ML method is stable enough to drive MD simulations, I also started some test simulations where the ML forces were used to relax the fragment geometries. The R^2 coefficients for both the site energy and the coupling models shown above looked quite promising. However, every one of the five long simulations I started crashed after 20 000 to 40 000 steps. Closer examination revealed that in every simulation, outlier predictions occurred, whose unusually large gradients pushed the geometry into a configuration where the predictions became even worse. Figure 5.9 shows one such example, where the site energy predicted at one point is $-500\,E_\mathrm{h}$ and the simulation crashes briefly after. In Figure 5.9b it is evident that not every outlier results in a fatal crash – small outliers do occur previously in the simulation, but after 3.27 ps the predictions get successively worse until the breaking of positional restraints of the MM simulation framework brings the simulation to a stop. I could not determine more precisely what causes these crashes, aside from the potential 'runaway' outliers. Detailed analysis of the NN prediction routine in GROMACS ruled out common programming errors which might cause bogus data to be

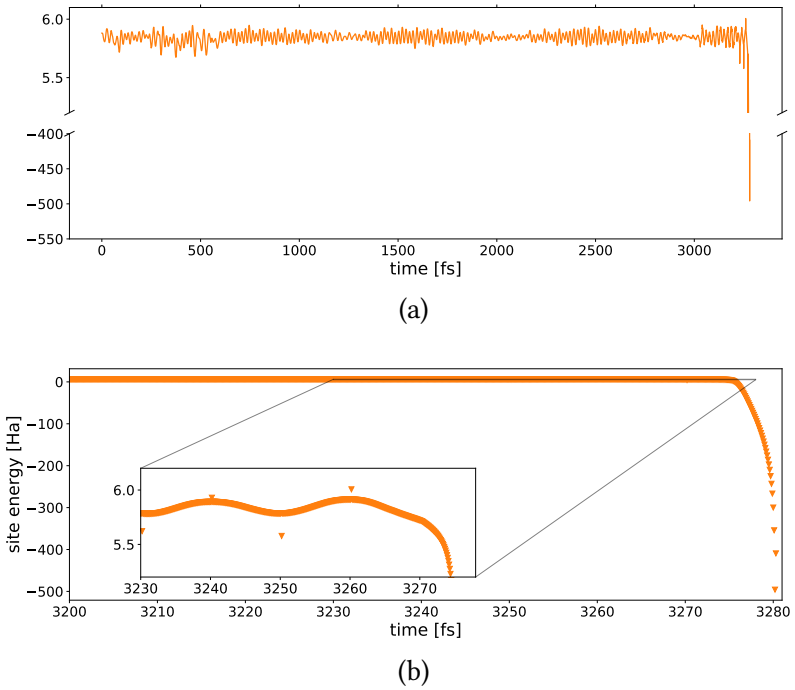


Figure 5.9.: Example for a site energy prediction so bad that the simulation crashes after 32 804 steps (3.28 ps). a) Overview over the entire simulation, b) a detailed look at the last few steps without the interpolated line.

read from memory instead of the predictions. As a next experiment, I decided on reducing outlier frequency by including structures sampled from high-temperature simulations in the data set. If this did improve the stability of the simulations, this would be evidence that it is indeed catastrophic outliers causing the crashes. The results of this experiment are discussed in section 5.7 below.

5.5.2. Performance Comparison with DFTB

To compare running times with DFTB and the NN, I re-ran and timed both the NN and DFTB calculations on the previously generated trajectory using a 0.1 fs timestep. On one core of an Intel Xeon CPU E5-2630 v4 @ 2.20GHz Processor, the DFTB calculation required 2314 h ns^{-1}, while a calculation using the NN models only needed 578 h ns^{-1}. This is a speedup by a factor of four compared to the highly optimized DFTB1 method, which is a considerable improvement in the size in which propagation is possible. Additional improvements can be achieved by reducing the size of the inverse distance representation for the off-diagonal terms to the intermolecular block of the distance matrix, and further optimizing the interfacing code. Additionally, when this scheme is applied to systems with larger monomers, the advantage of the NN model will increase significantly due to the unfavorable N^3_{atoms} scaling of the DFTB diagonalization.

model	depth	n_{neurons}	energy loss weight	regularizer	learning rate
off-diagonals	5	375	21	L2	1×10^{-4}
diagonals	4	460	3940	L1	1×10^{-4}

Table 5.1.: Best parameters for the diagonal and off-diagonal models found in the hyperparameter search.

5.6. Second Generation of Models and Hyperparameter Optimization

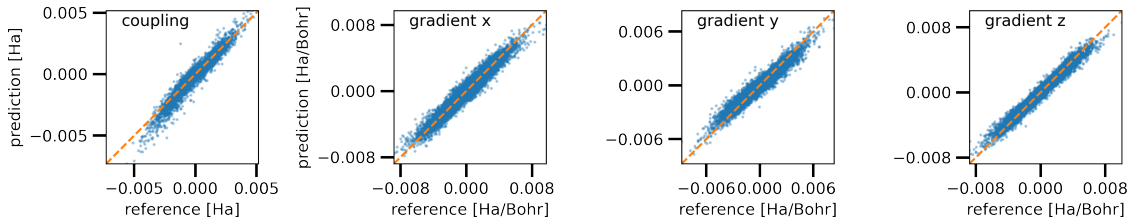
As a next step, I augmented the data set by adding in geometries sampled from higher temperatures, which should increase the geometric diversity of the training set and reduce the frequency of outliers. This resulted in the `mixed_temperatures` data set. At this point, I also began using only the intermolecular terms of the inverse distance matrix as representations for the off-diagonal models in order to further reduce model size and prediction costs. Additionally, having shown the general applicability of the NN models for learning Hamiltonian elements and their gradients, I was now ready to optimize the models in an extensive hyperparameter search.

For the hyperparameter search I used the Hyperband algorithm (section 3.6) as implemented in the `keras-tuner` package. In order to keep computational costs low while ensuring that the training data is sufficient for the model to learn, I used a training set size of 9000 data points during the hyperparameter optimization. Again, I used early stopping to avoid overfitting models to the training set, however, this time I did not couple this to a learning rate schedule and used a longer patience of 20 epochs. I set the maximum number of epochs to 1000 and the factor parameter of the Hyperband search to 3. For the search for the off-diagonal model, I used the merged R^2 coefficient with quadratic scaling (Equation 5.5). In order to push the model for the diagonal elements to an area where both the site energies and the gradients are well-learned, I used the exponential scaling (Equation 5.6).

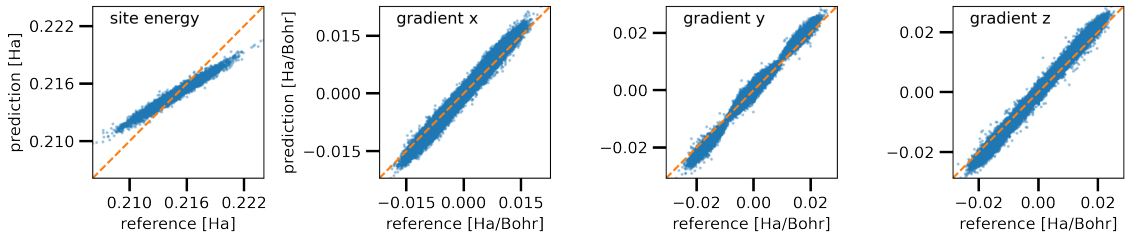
For both the diagonal and the off-diagonal models, the search space was 3, 4 and 5 layers for the depth and 20 to 200 for the number of neurons per layer. For the learning rate, values of 1×10^{-3}, 5×10^{-4} and 1×10^{-4} were possible, and either L1 or L2 weight regularization (Equation 3.21 and 3.22 in section 3.5) was used. The relative loss of the forces was held fixed at the value 1, while the search space for the energy loss weight was 500 to 5000 (diagonal elements) and 1 to 1000 (off-diagonals), respectively.

The results of the search are summarized in Table 5.1, and the results of the evaluation on the test set are visualized in Figure 5.10. When evaluated on a test set of 10 000 held-out structures from the `mixed_temperatures` data set, the model for the off-diagonal elements gave an MAE of $1.6 \times 10^{-4}\,E_\text{h}$ (4.3 meV) for the couplings and $4.7 \times 10^{-5}\,E_\text{h}\,{a_0}^{-1}$ (2.4×10^{-4} eV/Å) for the gradients, resulting in R^2 scores of 0.91 and 0.93, respectively. These values are very similar to those obtained for the model trained on 9000 data points from the `orig` data set.

For the model trained on the diagonal terms, the results are again very similar to those obtained above for the `orig` data set. On the test set, I obtained MAEs of $6.9 \times 10^{-4}\,E_\text{h}$



(a) off-diagonal elements



(b) diagonal elements

Figure 5.10.: Performance of NN models trained to a) off-diagonal and b) diagonal elements of the Hamiltonian and their gradients with hyperparameters optimized using the Hyperband algorithm.

(18.7 meV) for the site energies and $8.8 \times 10^{-4}\, E_\mathrm{h}\, {a_0}^{-1}$ (4.5×10^{-2} eV/Å) for the gradients. The R^2 scores were 0.83 and 0.98 for site energies and gradients, respectively.

While the `mixed_temperatures` data set does have more structural variety, it is unfortunate that the hyperparameter search does not give hyperparameters which give better results. However, this may be due to the fact that no learning rate scheduling was used for this model or the addition of weight regularization pushing the model farther to the mean. As an additional factor, the Hyperband algorithm fundamentally relies on a random sampling of the search space, so running the algorithm multiple times could find better areas[2]. However, this also would be quite costly, making it more and more impractical for the long-term use case of training models between MD equilibration and propagation simulations. Another reason for this lack of improvement the Hyperband algorithm achieves may be the definition of the combined R^2 score, and other objective functions may be compared and evaluated. The upside of the relative lack of success of the automated search, however, is that the surface of the objective seems to be quite flat, so the parameters found by simple manual exploration are already quite good.

5.6.1. Comparison to Diagonal Elements Obtained from LC-DFTB2

Crucially, the automated hyperparameter search for the diagonal element models did not find any areas where site energies and gradients were both learned well. This observation

[2] And of course, moving from Hyperband to Bayesian Optimization may also improve the results, but there, every model is trained to completion, requiring many additional epochs of training.

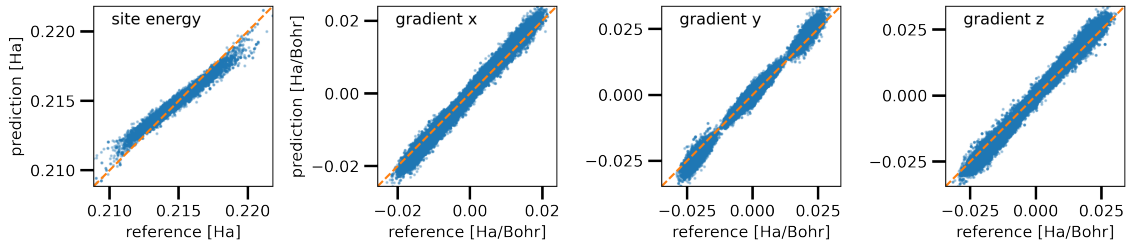


Figure 5.11.: Results for the *delta* model trained on the gradients from LC-DFTB2.

supports the theory that the models' failure to predict both quantities together is not an issue in the model or its configuration, but may be a problem in the data set. If the function–derivative relationship between site energy and gradients is not valid due to an error in the construction of the data set, this could lead to the behavior observed here.

The gradients used until now were calculated from the non-SCC DFTB electronic structure, and the orbital structure of the charged fragment is assumed to be equal to the orbital structure of the neutral molecule aside from the HOMO[3]. This avoids the calculation of the charged fragment completely, as the site energy is the HOMO energy and the gradients for relaxation are those resulting from the HOMO only. I wanted to double-check whether it is this calculation of the gradients which causes a fit conflict due to the approximation breaking down or an error in the implementation of this approximation which is used to calculate training data. To achieve this, I worked with Philipp Dohmen, who re-calculated all site energies and couplings in the `orig` data set using LC-DFTB2 as implemented in `dftb+`. I will refer to the resulting models as *delta* models, as the gradients for a given structure are evaluated as the true difference of the gradients in the neutral and charged states.

I performed the hyperparameter optimization for the delta model in the same manner as described above, hyperparameter grid settings for this search can be found in Table A.4. The best model found had an NN with 4 layers and 430 neurons per layer, the energy:gradient loss weight ratio was 553:2, and L2 weight regularization and a learning rate of 5×10^{-4} were used. The results for the delta model were very different from for the model using the non-SCC DFTB diagonal elements: While the results for the site energies greatly improved (MAE $3.5 \times 10^{-4}\,E_\mathrm{h}$ i.e. 9.6 meV, R^2 =0.94), the gradients were slightly worse, only reaching a MAE of $1.0 \times 10^{-3}\,E_\mathrm{h}\,{a_0}^{-1}$ (5.2×10^{-2} eV/Å, R^2 =0.98). The model showed neither the systematic shift for the site energies, nor the overestimation of the gradients that previous models (Figure 5.5) had, as can be seen in Figure 5.11.

As this model is easily able to capture the site energies and their derivatives just as well as the models for the off-diagonal elements reproduce the values they were trained on, I was able to conclude that something is indeed wrong in the data set of non-SCC diagonal elements. I did not analyze this issue further to determine whether the issue lay in the implementation, approximation, or the use of LC-DFTB2 vs. DFTB1, but in further propagation simulations, I only used the delta models for the site energies and couplings.

[3] or the LUMO for electron transfer (instead of hole transfer)

If the problem is indeed a breakdown of one of the approximations employed in the calculation of the DFTB1 diagonal elements, this inconsistency will be present also in other simulations using this method and has not been noticed before. While finding such errors was not the intended use of the ML model, the nature of the statistical inferences such models make also prove useful for validating non-ML methods. As a final note on this matter, any DFTB2 version has a greatly increased computational cost compared to DFTB1. If the DFTB1 approximation is indeed too harsh to give good results here, the computational cost of the reference method used to perform the propagation simulations will be greatly increased by the use of (LC-)DFTB2, and the speed advantage that the NN prediction provides will only grow.

5.6.2. Re-Evaluating Stability of Simulations using NN Models

Using the optimized models for the delta diagonal terms and the off-diagonal terms from the `mixed_temperatures` data set, new simulations were attempted to estimate whether the models can now give stable simulations. Out of seven 5 ps simulations, two crashed prematurely, again showing large outliers just before the simulation ended. Compared to the previously observed frequency of crashes, this is a significant improvement, indicating that the improved model for the diagonals and the improved sampling for the off-diagonals by using the `mixed_temperatures` data set did indeed make the simulations more stable. However, the frequency of crashes was still too high.

5.7. Third Generation of Models: Sampling the PES of the Charged Molecule

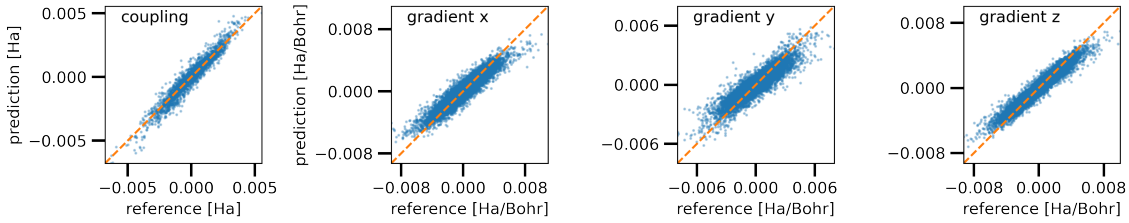
To improve stability and further reduce outliers, I began using data sets for which the geometries had been sampled from both the PES of the neutral and charged systems as explained in section 5.2. I trained models for the DFTB1 and delta diagonal terms and the off-diagonal terms on 30 000 data points each and searched for hyperparameters in the same manner as described above. For this search, I removed the initial learning rate from the search space, opting instead for a learning rate scheduler which would reduce the learning rate at fixed intervals (learning rates of 5×10^{-4}, 1×10^{-4}, 1×10^{-5} and 1×10^{-6} for 50, 100, 400 and 450 epochs, respectively). This should allow the model to converge into smaller local minima. I also increased the bounds of the search space for the number of neurons per layer to 1000 for all models, in the hope that more flexible networks may find better results.

Table 5.2 summarizes the best hyperparameter combinations found in the search, while further details about the search can be found in Figure A.7. Overall, the Hyperband algorithm iterated through 2074 candidate models in each of the three searches. One thing to note here is the fact that both the off-diagonal and diagonal elements converge to a value of 100 for the number of neurons per layer. The added flexibility from a larger network apparently does not result in better models.

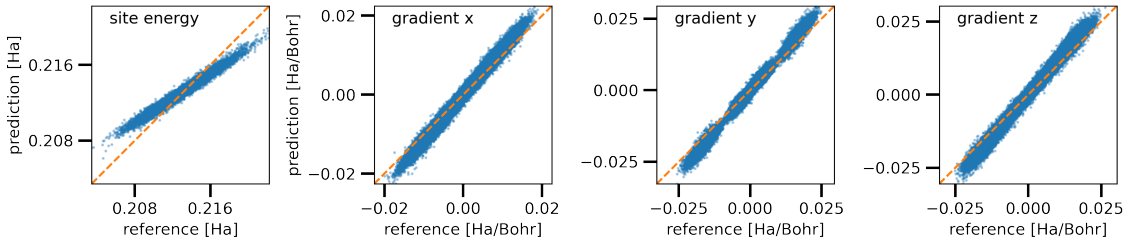
The MAEs and R^2 scores of all models are compared in Table 5.3, while Figure 5.12 visualizes the performance on the 10 000 structure test sets. R^2 scores are overall lower

model	depth	n_{neurons}	energy loss weight	regularizer
off-diagonals	5	100	31	L2
DFTB1 diagonals	4	100	1651	L2
delta diagonals	3	140	3821	L1

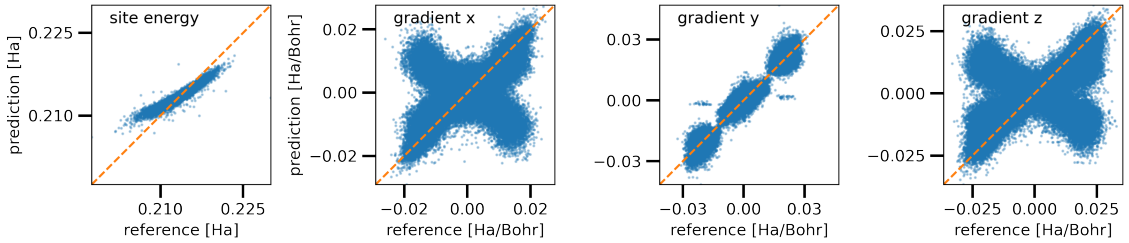
Table 5.2.: Best parameters for the off-diagonal and both DFTB1 and delta diagonal models found in the hyperparameter search.



(a) off-diagonal elements



(b) DFTB1 diagonal elements



(c) delta diagonal elements

Figure 5.12.: Performance of NN models trained to a) off-diagonal, b) DFTB1 diagonal, and c) delta diagonal elements of the Hamiltonian and their gradients from the `charged_uncharged` data set with hyperparameters optimized using the Hyperband algorithm.

	Hamiltonian element			gradients		
model	MAE [E_h]	MAE [eV]	R^2	MAE [$E_\mathrm{h}\,a_0{}^{-1}$]	MAE [eV/Å]	R^2
off-diagonal	8.3×10^{-5}	2.3×10^{-3}	0.92	3.4×10^{-5}	1.7×10^{-3}	0.83
DFTB1 diagonal	8.1×10^{-4}	2.2×10^{-2}	0.87	1.1×10^{-3}	5.5×10^{-2}	0.96
delta diagonal	1.0×10^{-3}	2.7×10^{-2}	0.79	3.3×10^{-3}	1.7×10^{-1}	0.49

Table 5.3.: Comparison of quality metrics for the models trained on the `charged_uncharged` data set.

and MAEs higher than they were on the `orig` and `mixed_temperatures` data sets, which confirms that the `charged_uncharged` data set does indeed have a larger structural variance than the other data sets. The DFTB1 diagonal terms continue to show the same issues as before. As for the delta diagonal elements, the reason for the bad scores is evident in Figure 5.12c: The cross shape in the scatter plots for the gradient elements along the x and z axes shows that their signs appear to be inconsistent and are probably randomly assigned. When confronted with such conflicting information, the model converges in one of two local minima, learning what is effectively one of the principal components of the distribution. In order to properly evaluate the delta models and use them for propagation, this issue must be resolved.

A brief test of the prediction performance of the models on the structures from the `orig` data set showed that the models were not consistently better at predicting the Hamiltonian elements and their gradients at 300 K. Compared to their performance on structures from the `charged_uncharged` data set, neither the MAE nor the R^2 decreased significantly. An exception to this was the model for the delta gradients, which was able to perform far better when the signs of the gradient components were consistent. The maximum errors, however, were significantly lower when the models were evaluated on the `orig` data set – for the couplings they decreased from 60 meV to 39 meV, for the DFTB1 site energies from 22 meV to 18 meV, for the delta site energies from 27 meV to 19 meV. For the gradients, maximum errors decreased from 267 eV Å^{-1} to 197 eV Å^{-1} for the off-diagonal elements and from 488 eV Å^{-1} to 430 eV Å^{-1} for the DFTB1 and 2880 eV Å^{-1} to 829 eV Å^{-1} for the delta gradients, respectively. This indicates that indeed the `charged_uncharged` data set covers an increased part of the conformational space than the `orig` data set.

With the models trained for the off-diagonals and the DFTB1 diagonals, propagation simulations were started to evaluate whether these new methods are finally stable enough to be tested in MD simulations. Indeed, in 600 propagation simulations with a duration of 3 ps (30 000 steps) each performed using these models, none crashed.

5.8. Application of Model Training Procedure for the DATT Molecule

Dianthra[2,3-b:2′,3′-f]thieno[3,2-b]thiophene (DATT)[25] is a compound with promising charge transfer properties. The DATT molecule is twice as large as the anthracene molecule, making it an interesting target for use of NN predicted Hamiltonians versus

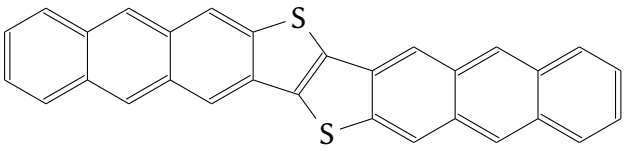
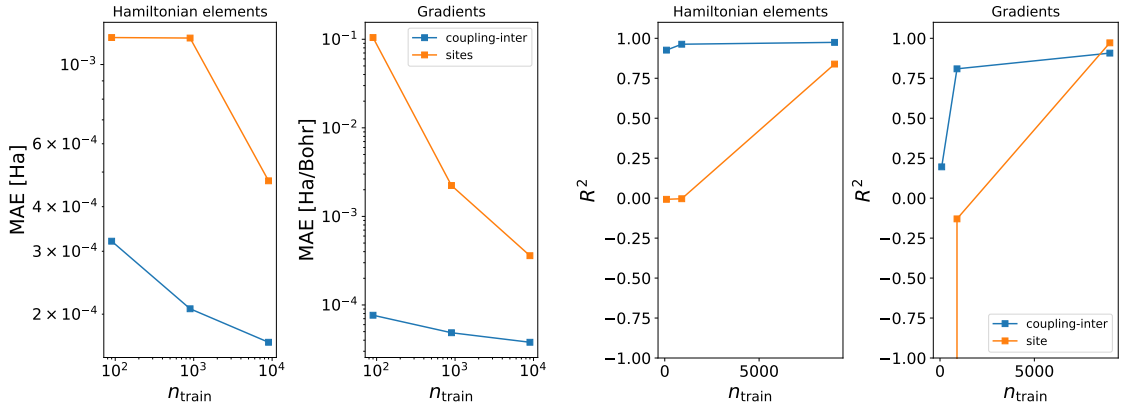


Figure 5.13.: Structure of DATT

DFTB calculations. The training data for DATT was generated in the same manner as the `mixed_temperatures` data set for anthracene by Weiwei Xie. Models were trained using the same parameters as were used for the learning curve for anthracene before (subsection 5.4.3). Only the batch size for model training needed to be reduced to fit in the available GPU memory, from 64 to 32 (diagonal elements) and 16 (off-diagonal elements). For the learning curve, only models using training set sizes of 90, 900 and 9000 structures were trained due to memory limitations from the implementation of the normalization step in the NN architecture. This limitation can easily be circumvented in the future with small changes in the `pyNNsMD` package.

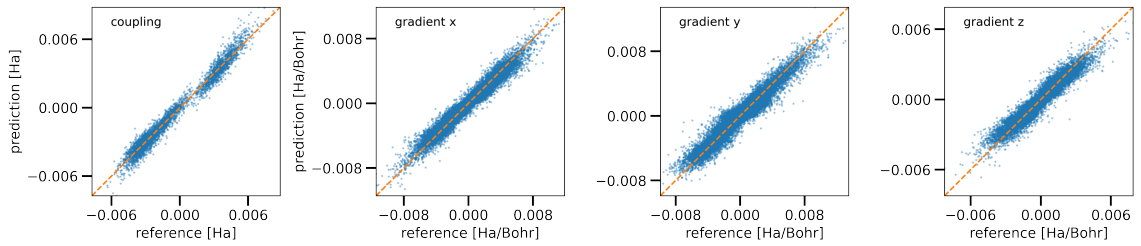
Figure 5.14 shows that for the DATT molecule, the models were able to learn and improve their prediction when given more data points.



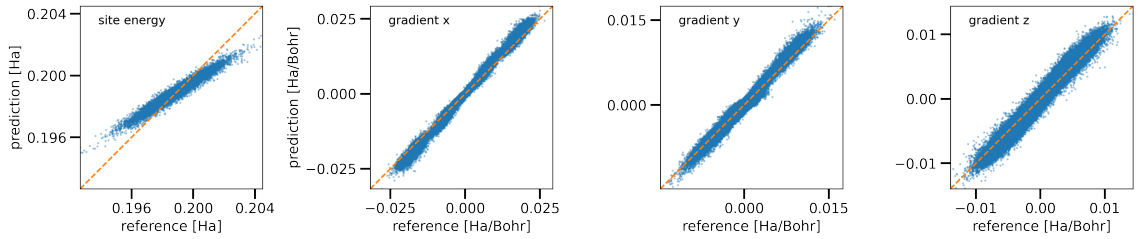
(a) (b)

Figure 5.14.: MAE (a) and R^2 (b) as a function of training set size for the `coupling-full`, `coupling-inter`, and `site` models.

A detailed comparison of the metrics for the largest models can be found in Table 5.4, showing that MAEs and R^2 coefficients were very similar to those reached for the anthracene model. Here, the issue with the site energies calculated from DFTB1 implemented within GROMACS can be seen again, as the model again does not properly learn the site energy (Figure 5.15b), while for the off-diagonal elements there is no such problem (Figure 5.15a). Note, that the hyperparameters which I used here were not optimized but just the same as for the anthracene models trained on the `orig` data set. This hints that the model performance is indeed not strongly dependent on the specific hyperparameters, except for the relative loss weight parameter, which must favor the energy strongly enough for it to compete with the forces.



(a) off-diagonal elements



(b) DFTB1 diagonal elements

Figure 5.15.: Performance of NN models trained to a) off-diagonal, b) DFTB1 diagonal elements of the Hamiltonian and their gradients for the DATT molecule from the `charged_uncharged` data set with hyperparameters optimized using the Hyperband algorithm.

	Hamiltonian element			gradients		
model	MAE [E_h]	MAE [eV]	R^2	MAE [$E_\mathrm{h}\,{a_0}^{-1}$]	MAE [eV/Å]	R^2
off-diagonal	1.7×10^{-4}	4.5×10^{-3}	0.97	3.8×10^{-5}	1.9×10^{-3}	0.91
DFTB1 diagonal	4.7×10^{-4}	1.3×10^{-2}	0.84	3.6×10^{-4}	1.9×10^{-2}	0.97

Table 5.4.: Comparison of quality metrics for the models trained on the 9.0×10^3 structures from the DATT data set.

These results show that this method is easily generalizable to other systems, including molecules far larger than anthracene. The next challenge for these networks will be learning exciton transfer couplings for Bacteriochlorophylls in biological light-harvesting systems, which are twice as large again as DATT.

5.9. Chapter Summary

In this chapter I showed that neural network models are able to learn the Hamiltonian elements for charge transfer propagation, including their gradients and signs (if the latter are consistent in the training data set) in both Anthracene and DATT. The models I obtained for anthracene gave a speedup of a factor of four compared to DFTB during propagation simulations, which can be further increased by several methods: First, the size of the representation can be further reduced, by using automated dimensionality reduction techniques or empirical testing to determine a minimal subset of interatomic distances. Second, the size of the NN model can be included as a factor in the hyperparameter optimization technique, to drive the algorithms to small models of high quality. Finally, the NN model pulls farther ahead if a more costly and accurate method than DFTB1 is used as a reference.

The change of reference may be necessary, as the experiments with the NN model revealed an inconsistency between the site energies and their gradients as calculated from the DFTB1 implementation in the GROMACS code used for data generation and propagation simulations. Using site energies and gradients obtained from LC-DFTB2, the NN was able to learn both the site energy and its gradients well. While investigating the specific reasons for this inconsistency were out of scope for this work, this is another case where an ML model's inability to learn a specific property could be traced to inaccuracies in the reference method.

The hyperparameters of the NN models were not easy to optimize, and more experiments with the optimization algorithm and search space are necessary to determine how to efficiently find working parameters. However, excepting the relative loss weights of energies and gradients, no hyperparameter had a strong influence on the quality of the model, so the hyperparameters may be generalizable enough that hyperparameter optimization is not necessary for every system.

Another important lesson from the work in this chapter was the exceedingly large importance of the quality of the force predictions. If an ML model is to change geometries during a simulation, it is essential to minimize outliers as much as possible to avoid ending up in a self-perpetuating cycle of bad predictions causing bad geometries causing worse predictions. Even close to perfect R^2 scores do not guarantee the model is good enough to achieve this. Ensuring that all thermally accessible geometries are sufficiently represented in both the training and test sets is crucial. Achieving this by sampling at very high temperatures may not be possible in organic systems, but should be attempted as much as possible. If the system should switch PES during the simulation, the model needs to be trained with sufficient data from both PES to give stable simulations.

An efficient method of uncertainty estimation would go a long way here to sanity-check the model outputs and prevent catastrophic failures. Lacking that, the model can indeed be

made sufficiently stable to perform MD simulations by making sure that the training set is as diverse as possible. Given the easy integration of Tensorflow models in GROMACS using the C-API, even an active learning approach on data generated during MD equilibration is feasible.

6. Neural Networks for Expanded Exciton Transfer Hamiltonians

In organic materials and biological systems, the Frenkel states may be insufficient to properly describe the exciton transfer process, and charge-transfer (CT) states must be taken into account in order to get a full picture[136, 137]. Within the formalism for ET simulations introduced in subsection 2.4.3, the site energies and couplings for the CT states available in a system can be included in the Hamiltonian for propagation just like the Frenkel states previously were. The Hamiltonian now must include the couplings of the CT states with each other as well as with the various Frenkel states. However, inclusion of the CT states requires costly excited-state calculations for every dimer of neighboring fragments, as the approximation which resulted in the Coulomb couplings (Equation 2.22) cannot be employed here.

In the spirit of chapter 4, I therefore decided to train an ML model which was able to predict the elements for the Hamiltonian including CT states. The KRR models presented previously for exciton transfer however quickly proved too unwieldy for this task, and I switched to a neural network model. This project is still in its beginning stages as a problem in the training data generation method needed to be fixed, and I have not optimized the model or analyzed its performance in depth.

6.1. Structure of the Expanded Hamiltonian

Including the CT states greatly increases the size of the Hamiltonian. On the diagonal, the excitation energies of the Frenkel states are joined by the excitation energy of each CT state. As the CT state includes two fragments, the CT site energies also depend on pairwise geometries. Additionally, each pair of neighboring fragments results in two different CT states, depending on which fragment carries the electron and which carries the hole. Figure 6.1 schematically shows the Frenkel and CT states for a pair of neighboring molecules. The off-diagonal elements of the Hamiltonian contain the pairwise couplings between the states. Only the lowest-energy Frenkel and CT states are included in the Hamiltonian. The couplings between CT states which do not occupy the same pair of molecules are negligible and are thus excluded from the Hamiltonian, as are the couplings between a CT state and any Frenkel states which do not share a fragment with that CT state.

For a set of three fragments numbered 0, 1, 2 which are arranged in a row with fragment 1 located between fragments 0 and 2, the terms included in the Hamiltonian are shown in Table 6.1. The quantities marked in green are monomer properties – in the fragment-based

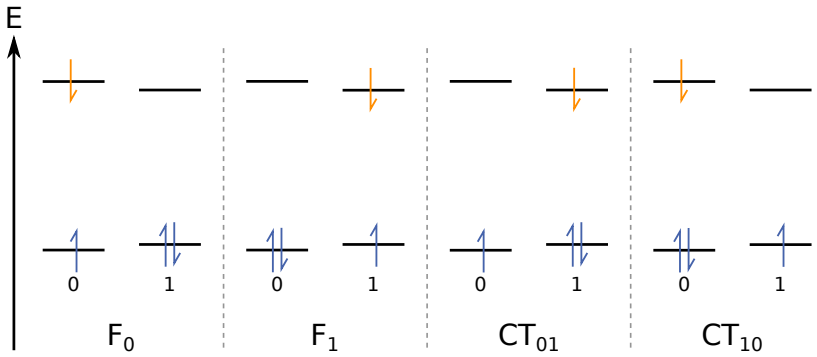


Figure 6.1.: Two Frenkel states and two CT states for two neighboring molecules indexed 0 and 1.

	F_0	CT_{01}	CT_{10}	F_1	CT_{12}	CT_{21}	F_2
F_0	F_0	F_0, CT_{01}	F_0, CT_{10}	F_{01}	0	0	0
CT_{01}	F_0, CT_{01}	CT_{01}	CT_{01}, CT_{10}	F_1, CT_{01}	0	0	0
CT_{10}	F_0, CT_{10}	CT_{01}, CT_{10}	CT_{10}	F_1, CT_{10}	0	0	0
F_1	F_{01}	F_1, CT_{01}	F_1, CT_{10}	F_1	F_1, CT_{12}	F_1, CT_{21}	F_{12}
CT_{12}	0	0	0	F_1, CT_{12}	CT_{12}	CT_{12}, CT_{21}	F_2, CT_{12}
CT_{21}	0	0	0	F_1, CT_{21}	CT_{12}, CT_{21}	CT_{21}	F_2, CT_{21}
F_2	0	0	0	F_{12}	F_2, CT_{12}	F_2, CT_{21}	F_2

Table 6.1.: Hamiltonian elements for three sites indexed 0, 1, 2 arranged in a chain such that fragments 0 and 2 are on either side of fragment 1.

approach, the site energy of the Frenkel state on a monomer only depends on its own geometry. All other entries of the Hamiltonian are dimeric properties.

6.2. Design of the Machine Learning Model

In the ML scheme for predicting Hamiltonian elements as presented in chapter 4, two models were sufficient to predict the entire Hamiltonian: one model for the monomeric site energy, and one for the pairwise couplings. Here, there are multiple types of couplings (per fragment pair, there is one Frenkel-Frenkel, four different types of Frenkel-CT couplings corresponding to each possible occupation, and one CT-CT coupling), as well as the site energies corresponding to the Frenkel and CT states, which are not suited to be learned together in one KRR model because the latter depend on a dimer geometry. While it is possible to predict all elements of this Hamiltonian using only two KRR models (a single-target model for the Frenkel energies and a multi-target KRR model for all dimer properties together), this solution would have problems. Most importantly, the model for the dimer properties would have required very large amounts of training data to give reasonable results for all eight of its targets (two CT site energies and all six couplings) together. KRR models, however, are best suited for problems where few training data points are sufficient, as the computational cost of prediction and the memory footprint

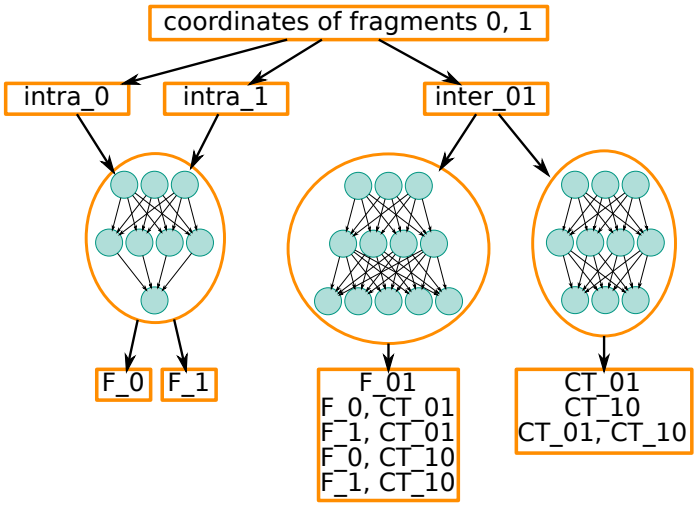


Figure 6.2.: NN architecture for the full Hamiltonian between two fragments numbered 0 and 1. The dimer properties are here predicted by two separate subnetworks depending on whether they involve Frenkel states or not (`split` grouping)

during training can both become prohibitively large with an increased training set size. Additionally, the KRR models had previously shown problems with learning the sign of the couplings, which had required restricting the models to predicting absolute values.

I therefore decided to move away from the KRR description to a neural network. With a neural network, adding more training data has a far smaller effect on the costs of the training (and does not change the cost of prediction at all). Additionally, in chapter 5, I could show that NNs were able to learn the sign of the couplings without issue.

I used the inverse distance representation and feature normalization layers introduced in the `pyNNsMD` package (see chapter 5). The added structural flexibility which NN models add allowed me to experiment with a slightly more involved architecture, which is schematically shown in Figure 6.2. The Cartesian coordinates of the atoms in a fragment pair are used to calculate the matrix of inverse interatomic distances. Then, the distance matrix is split into three parts: The two intramolecular parts containing only distances between the atoms of either fragment 0 or fragment 1 are separated from each other and from the intermolecular terms of the distance matrix, containing all pairwise interatomic distances where one atom belongs to fragment 0 and the other to fragment 1. This separation was motivated by the results from section 5.3 and [100] that the intermolecular part of the distance matrix is sufficient for the prediction of coupling terms.

As the Frenkel energy of a fragment depends only on its geometry, the Frenkel energies of the two fragments are predicted by the same sub-network of the model. This network gets as inputs the interatomic inverse distances of a fragment and predicts the site energy corresponding to the Frenkel state.

The dimer properties can now be grouped in multiple ways. I created two networks with somewhat intuitive groupings of properties to compare, but the network architecture is written in a way that allows for arbitrary groupings. First, I tested to have one sub-network predict all dimer properties together (which I will refer to as the `naive` grouping). The second grouping is the one shown in Figure 6.2, where all dimer properties containing a

	`naive`		`split`		
	Frenkel monomer	all dimer	Frenkel monomer	CT-only dimer	other dimer
neurons	100	2000	100	1000	1000
layers	3	5	3	5	5
loss weight	1	20	1	10	10

Table 6.2.: Summary of the NN architecture for both the `naive` and the `split` grouping.

contribution from a Frenkel state are treated in one sub-net, while the remaining properties which correspond to charge-transfer states only are treated in another. I will refer to this grouping as the `split` grouping.

In both models, the subnetwork for the Frenkel site energies is the same size, three hidden layers with 100 neurons each. For the `naive` model, the dimer subnetwork has five hidden layers á 2000 neurons. The `split` model uses a subnetwork with 1000 neurons arranged in five layers for the CT-only and Frenkel-including dimer properties each. Table 6.2 summarizes the parameters relevant for the model architectures.

All hidden layers use the leaky softplus activation function (Equation 5.1) with default parameters, the final output layer activations are linear. For training, I used mean squared error loss and the Adam optimizer with a learning rate of 0.001 and monitored the loss on a validation set using the early stopping method (maximum 500 epochs, stop if no improvement for 20 epochs) to avoid overfitting. All models were trained on a randomly chosen set of 4500 data points, with 500 additional data points used as the validation set. The models were tested on 10 000 test structures.

6.3. Generation of the Reference Data Set

The data set was generated by Farhad Ghalami using the same method as for the exciton transfer data in subsection 4.1.1 with the added inclusion of CT states. It includes 29 029 geometries of pentacene dimers sampled from an MD simulation in the same way as the anthracene structures were previously sampled. For each dimer, the data set includes both Frenkel site energies, the Frenkel coupling, the site energies of both CT states and the coupling between them, as well as all four distinct Frenkel-CT coupling elements – ten properties in total. All coupling elements are signed (see Figure 6.3), but using a NN instead of KRR model it should be possible to learn the signed value for the coupling as long as it is consistent and the sign is not randomly assigned.

6.4. Model Evaluation

The first model I trained used the naive grouping and the data set as-is, including the signed values for the couplings. As Figure 6.4 shows, this model was able to learn the site energies of both the Frenkel and the CT states, but completely failed to learn the coupling elements beyond their mean of 0.0 eV. Learning the sign of the couplings for hole transport using an NN had not been a problem previously, so I suspected that the problem

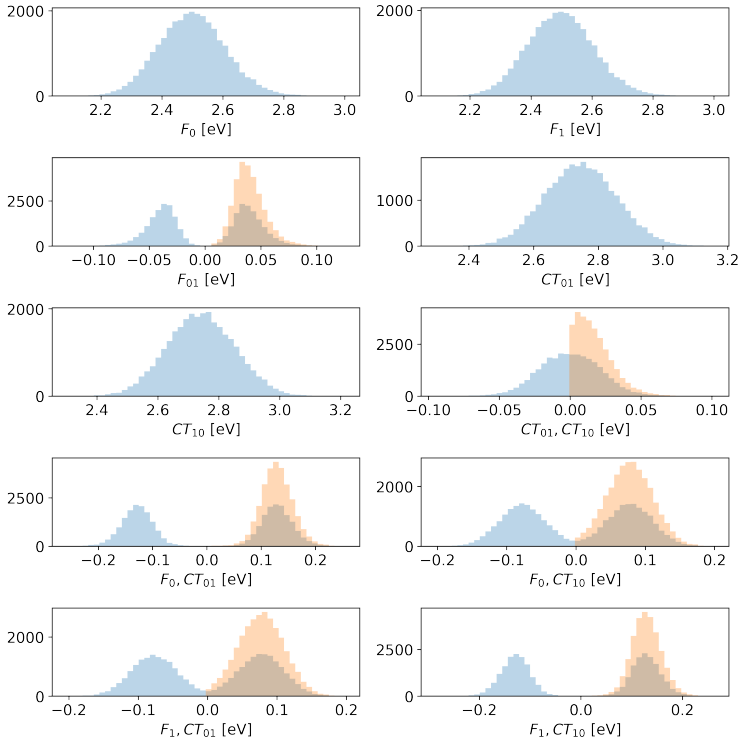


Figure 6.3.: Distributions of the target values in the data set. For couplings, absolute value distributions are shown as well.

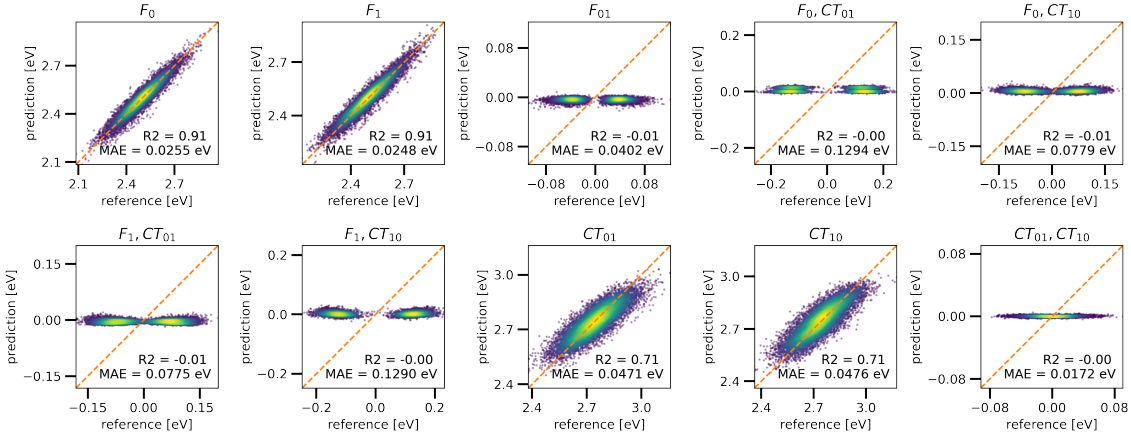


Figure 6.4.: Results for the model trained for the naive grouping on the data from the data set (including the signs of the coupling elements).

lay in this data set. The signs of the couplings are a direct result of the signs of the wave function coefficients (in the charge transfer case) or of the transition charges (for exciton transport). Usually, these are randomly initialized for every calculation, and therefore the initial sign of the couplings is random, as well. During a simulation, only the relative signs of the couplings matter, which can be kept consistent by monitoring the wave function coefficients in every step, as it is done for the charge transfer simulations using GROMACS. However, exciton transport simulations using the same code are currently only possible if the Hamiltonian is provided by an ML method. The reference data used here is therefore the result of many independent LC-TD-DFTB calculations, which leads to random signs for the coupling values.

I trained another model, this time using the absolute values for all coupling elements. This model was indeed able to learn all coupling elements, as can be seen in Figure 6.5. For

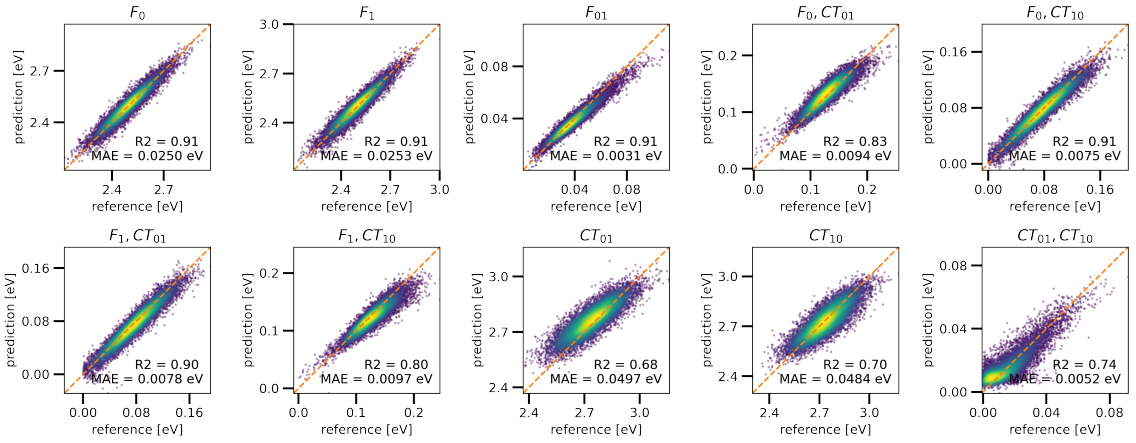


Figure 6.5.: Results for the model trained for the naive grouping on the data from the data set using absolute values for the couplings.

all couplings except for that between the two CT states, the models give very good results, with R^2 coefficients above 0.8. The coupling between the two CT states (CT_{01}, CT_{10}) appears to be more difficult to learn, but this may just be a result of the distorted distribution resulting from taking the absolute value.

A third model using the split grouping and the absolute value gave very similar results, shown in Figure 6.6. This similarity indicates that there may be no significant difference between the two groupings, but further investigation of models with various groupings is needed before more conclusions can be drawn.

In light of the sign inconsistency in the training data, I postponed more detailed experiments and analysis until a scheme for consistent signs could be developed and tested.

6.5. Chapter Summary

In this chapter, I presented a flexible multi-output neural network model which can predict arbitrary combinations of monomeric and dimeric properties for exciton transfer. Given the geometries of a fragment pair, the model is able to predict the energies of the Frenkel

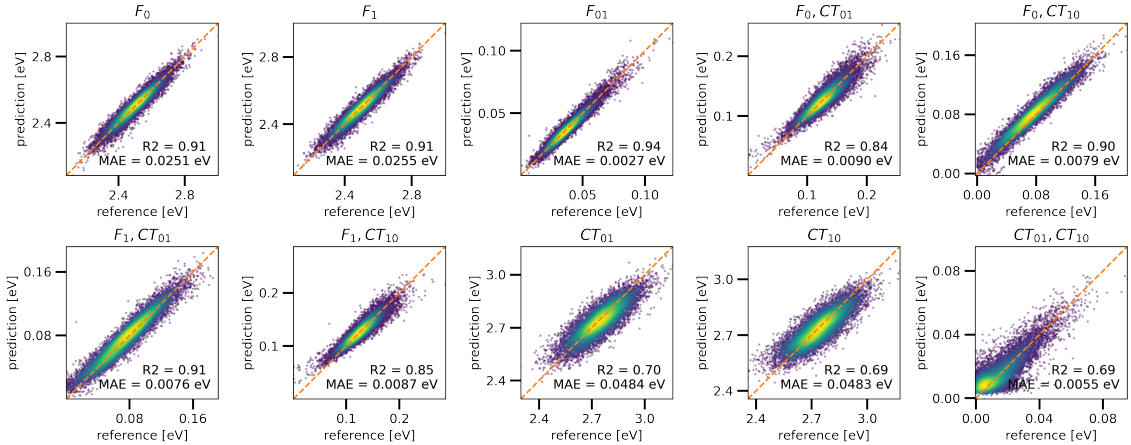


Figure 6.6.: Results for the model trained for the split grouping on the data from the data set using absolute values for the couplings.

and charge-transfer states, as well as any of the couplings between the fragments. The model architecture is divided of sub-units, which allows to balance the flexibility available for the predictions of each individual quantity by varying the number of sub-networks and assignments of properties to them.

In first tests, the model trained on data generated for pentacene was able to learn the couplings only if their absolute values were taken, indicating that the sign of the coupling was inconsistent in the reference data. Therefore, I performed no further optimization of the architecture until a consistent scheme for data set generation could be implemented.

The models presented in this chapter are simple and performant, easily adjustable and have few hyperparameters. Considering the computational costs of the electronic structure calculations for the reference, predicting the expanded exciton transfer Hamiltonian using the ML model will outpace even semi-empirical quantum chemistry methods and enable direct simulations of exciton transfer even in very large systems.

Part III.

Concluding Remarks

7. Summary and Outlook

In this work I investigated the application of ML models for learning electron and exciton transfer Hamiltonians for propagation simulations. The low computational costs for ML predictions compared to electronic structure methods promise to enable the direct simulation of charge and exciton propagation through systems larger than currently feasible.

I first presented simple KRR models trained to predict electron and exciton transfer site energies and couplings in anthracene. Once a good representation for the inputs was found, these models were easy to train to high accuracies, and model training and hyperparameter search was easy to automate. However, only the absolute value of the couplings could be learned in practice, although the scheme used to generate the training data produced a consistent, geometry-dependent sign. The models performed well in reproducing the reference values both when evaluated on a large number of sampled geometries and along the trajectory for an individual fragment or pair of fragments.

During propagation simulations, the models provided the Hamiltonians at a fraction of the computational costs of true *ab initio* methods or time-dependent DFTB, but struggled to significantly improve upon the speed of non-SCC DFTB used for charge transfer. When used in combination with the mean-field Ehrenfest method for propagation, models trained on only 1000 geometries were able to reproduce the hole mobilities in the anthracene crystal to within 8.5 % of the DFTB reference and within 30 % of the experimental values. When a surface-hopping method was used for propagation, the ML model was able to reproduce the equivalent DFTB reference, but compared to experiment the mobilities were greatly overestimated. The reason for this discrepancy to experimental values was the fact that the ML model could provide neither the forces needed to relax occupied fragments, nor the non-adiabatic coupling vectors necessary to properly re-scale the velocities of the nuclei when a hop occurred. To make the comparison to the ML model fair, DFTB also used implicit relaxation, and when the DFTB simulation was repeated using full relaxation, it achieved results far closer to the experimental reference. For exciton transfer, it was more difficult to evaluate the quality of the resulting models in predicting observables, as there was no reference implementation for propagation using LC-TD-DFTB. When the results obtained from a BC-FSSH propagation using the ML Hamiltonian were compared to those obtained from Marcus theory, the diffusion coefficients were overestimated similarly as observed for the hole mobilities.

I subsequently tackled the main limitations of the previously used models: First, the KRR models' inability to learn the sign of the couplings would become problematic in simulations where the charge would be able to move in more than one dimension. Second, for ML Hamiltonians to be usable in surface-hopping based simulations, the model needed to provide occupation-dependent forces and non-adiabatic coupling vectors, two quantities which only become accessible if the gradients of the model w.r.t. the input coordinates can

be evaluated. Due to the difficulty in obtaining gradients from KRR models, as well as the difficulties KRR models had in learning the sign of the couplings and their unsuitability to be used with large training data sets, I moved to a neural network architecture. Using this approach, I was able to obtain models which predicted (signed) charge transfer couplings in anthracene to within ≈2 meV and the gradients required to calculate NACVs with errors less than 0.1 meV $Å^{-1}$. I also applied the method developed here for DATT, an organic semiconductor which is twice as large as a single anthracene molecule, and obtained methods which gave very good results even without further hyperparameter optimization.

Obtaining good models for the diagonal elements of the Hamiltonian (i.e. site energies and their derivatives, which are used for occupation-dependent relaxation) proved more difficult, as I could not find models which were able to learn both the site energy and its gradients equally well. At first, this seemed like a network configuration issue, but an automated hyperparameter search did not improve the results. Indeed, calculating the reference values using LC-DFTB2 instead of the previously used DFTB1 implemented in GROMACS gave an alternative data set for the diagonal elements in which both site energies and gradients could be learned without issue. Therefore, the model's inability to learn the two properties together in a function–derivative relationship is evidence for an inconsistency in the calculation of the gradients implemented in GROMACS. Although I could not determine conclusively whether the issue is in the specific implementation or the approximations used in DFTB1, this issue warrants further investigation.

When the trained models were used in propagation simulations, however, the importance of the sampling quality in the training data set for the models became clear: Initially, every simulation in which forces derived from an ML model's prediction were used, crashed due to individual outliers pushing the system to ever more unrealistic geometries. These outliers occur when the model encounters a structure during the simulation which was not well-represented in the training data set. I therefore successively refined the strategy used for sampling training data from MD simulations. While the initial data set included only geometries sampled at 300 K, also including structures from simulations at 500 K improved the stability of the simulations Including geometries sampled from the PES of the charged molecule at 500 K reduced the frequency of fatally bad predictions to less than one in eighteen million time steps, at which point the models were stable enough to be used in simulations.

A detailed evaluation of these models in propagation was not yet possible, however, as the cause of the issues in the calculation of the diagonal elements could not yet be determined and thus no comparable reference simulation could be run. However, a comparison of the times required to run propagation simulations with the ML models showed that this method outperformed non-SCC DFTB by a factor of four for anthracene. For larger molecules or applications where LC-(TD)-DFTB is required, this advantage would only grow.

Finally, I presented a flexible neural network architecture which is capable of predicting multiple types of excitonic site energies and couplings for a pair of fragments in one pass. This network can be used when the Hamiltonian for propagation contains not only the Frenkel excitons but also charge-transfer states. The architecture consists of multiple sub-networks which can handle different elements of the Hamiltonian, and can be easily

extended to include more types of interactions. Trained on a data set of Hamiltonian elements for pentacene, the network was able to learn both site energies and the absolute values of all couplings quite well. However, the signs for the couplings in the data set for this model were randomly assigned, as there no phase correction scheme was used during training data generation. Therefore, I held off on further improving and evaluating the models until I could obtain a data set with consistent signs.

All ML models I presented in this work can be trained and optimized in an automated fashion, which can be integrated in the standard workflow for the preparation of simulations. The models can be trained on structures generated as part of the system equilibration, although this sampling approach might be insufficient if the model's gradients are to be used in the simulation. The interfacing which I implemented between GROMACS and the Tensorflow API allows for flexible use of ML model predictions in MD simulations, so that adaptive sampling and even the refinement of models during propagation can be included in the MD simulation with little added implementation effort. While any individual model in this work can only give reasonable predictions for the system it was trained on, the entire framework and process is purposefully designed to be as generic as possible. By removing the need for individual models to generalize across chemical space, I was able to keep the models small and easy to train. As the computational cost of generating the necessary training data with DFTB is far lower than the costs incurred by *ab initio* methods, this approach can create robust, easily applicable and automatically trainable models for every specific system which integrate well in existing multi-step materials discovery workflows.

At several points in this work, I encountered situations where the ML models were unable to learn the desired input–target relationship, but detailed analysis of the issues revealed that the error was not in the ML model, but the method used for generating the reference data: When I attempted to learn supermolecular excitonic couplings using KRR in chapter 4, several attempts at training ML models gave very unsatisfactory results. The analysis of these results pointed to problems in the reference data, and indeed after the data was re-generated using the correct formula, the KRR model was able to learn supermolecular couplings as well. The models for diagonal elements of the Hamiltonian in chapter 5 were unable to learn both the site energy and its gradients in a function–derivative relationship when the reference values were calculated using non-SCC DFTB implemented in GROMACS, but there was no problem when LC-DFTB2 was used as a reference. In chapter 6, the fact that the signs of the excitonic couplings in the reference data set were randomized for every geometry became evident when the model failed to learn anything but the mean value for the couplings. These observations point to a rarely-considered application of machine learning models as tools for the validation of other, physically motivated methods. The reliance of the models on the extraction of patterns and connections between variables in large data sets makes them well-suited as detectors for cases when expected patterns are not there (such as properties with randomized signs), or assumed functional relationships do not hold (if one value is not the derivative of another w.r.t. atomic positions). Designing ML models specifically for the evaluation and validation of methods in computational chemistry may therefore be an interesting field for further research and of great benefit to the research community.

Not all models presented here resulted in a significant reduction of computational cost compared to their reference methods: For all exciton transfer applications, the need for time-dependent QM calculations results in a speed improvement compared to LC-TD-DFTB of several orders of magnitude. In charge transfer simulations, the picture was not so clear-cut: While NN methods were indeed several times faster than DFTB during simulations in anthracene, this factor was smaller than expected, and KRR models only improved upon the cost of DFTB if small training sets were used. This highlights the efficiency of the DFTB method and the approximations introduced for the propagation scheme. For its high accuracy when predicting charge transfer couplings, DFTB has computational costs low enough to be competitive with ML methods for systems of up to $\approx$24 atoms per fragment. However, the DFTB cost scales less favorably with the fragment size than the cost of an ML prediction of the site energies and couplings. The anthracene molecule, frequently used for this work, is at the low end of the size spectrum for organic semiconductors whose charge and exciton transfer properties are of interest. For exciton transfer simulations in biochemical systems, targets include very large systems like the Bacteriochlorophyll rings contained in light-harvesting systems like the LH2 complex. In these systems, performing direct exciton transfer dynamics using semiempirical methods is currently beyond the scope of what is feasible, and the ML models designed in this work enable these simulations.

Together, these contributions lay the groundwork for the application of ML methods for use in charge and exciton transfer simulations where the fragment-based Hamiltonian and the gradients of its elements w.r.t. atomic positions are the only quantities required from the electronic structure calculations. The easy and automated training, the use of a generic geometric representation instead of a system-specific one motivated by domain knowledge, and the interfacing with the GROMACS code for propagation all keep the barriers to creating and using ML-driven propagation simulations in ever new systems as low as possible. Using the methods presented in this work, direct simulations of exciton transfer through multi-protein complexes or simulations of charges moving through organic electronics components come within reach.

Bibliography

[1] Dirk Porezag, Th Frauenheim, Th Köhler, Gotthard Seifert, and R Kaschner. "Construction of Tight-Binding-like Potentials on the Basis of Density-Functional Theory: Application to Carbon". In: *Phys Rev B* 51.19 (1995), p. 12947.

[2] Marcus Elstner, Dirk Porezag, G Jungnickel, J Elsner, M Haugk, Th Frauenheim, Sandor Suhai, and Gotthard Seifert. "Self-Consistent-Charge Density-Functional Tight-Binding Method for Simulations of Complex Materials Properties". In: *Phys Rev B* 58.11 (1998), p. 7260.

[3] Thomas A Niehaus, S Suhai, F Della Sala, P Lugli, M Elstner, G Seifert, and Th Frauenheim. "Tight-Binding Approach to Time-Dependent Density-Functional Response Theory". In: *Phys Rev B* 63.8 (2001), p. 085108.

[4] Balint Aradi, Ben Hourahine, and Th Frauenheim. "DFTB+, a Sparse Matrix-Based Implementation of the DFTB Method". In: *J Phys Chem A* 111.26 (2007), pp. 5678–5684.

[5] B. Hourahine et al. "DFTB+, a Software Package for Efficient Approximate Density Functional Theory Based Atomistic Simulations". In: *J. Chem. Phys.* 152.12 (Mar. 2020), p. 124101. ISSN: 0021-9606. DOI: 10.1063/1.5143190.

[6] Michael Feig, Ryuhei Harada, Takaharu Mori, Isseki Yu, Koichi Takahashi, and Yuji Sugita. "Complete Atomistic Model of a Bacterial Cytoplasm for Integrating Physics, Biochemistry, and Systems Biology". In: *J Mol Graph Model* 58 (May 2015), pp. 1–9. ISSN: 1093-3263. DOI: 10.1016/j.jmgm.2015.02.004.

[7] rikenchannel. *All-Atom Molecular Dynamics Simulation of the Bacterial Cytoplasm.* June 2017.

[8] Julia Westermayr, Michael Gastegger, Kristof T. Schütt, and Reinhard J. Maurer. "Perspective on Integrating Machine Learning into Computational Chemistry and Materials Science". In: *ArXiv210208435 Phys.* (Apr. 2021). arXiv: 2102.08435 [physics].

[9] J. S. Smith, O. Isayev, and A. E. Roitberg. "ANI-1: An Extensible Neural Network Potential with DFT Accuracy at Force Field Computational Cost". en. In: *Chem. Sci.* 8.4 (Mar. 2017), pp. 3192–3203. ISSN: 2041-6539. DOI: 10.1039/C6SC05720A.

[10] Bryan R. Goldsmith, Jacques Esterhuizen, Jin-Xun Liu, Christopher J. Bartel, and Christopher Sutton. "Machine Learning for Heterogeneous Catalyst Design and Discovery". en. In: *AIChE J.* 64.7 (2018), pp. 2311–2323. ISSN: 1547-5905. DOI: 10.1002/aic.16198.

[11] Maria Korshunova, Boris Ginsburg, Alexander Tropsha, and Olexandr Isayev. "OpenChem: A Deep Learning Toolkit for Computational Chemistry and Drug Design". In: *J. Chem. Inf. Model.* 61.1 (Jan. 2021), pp. 7–13. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.0c00971.

[12] Xin Yang, Yifei Wang, Ryan Byrne, Gisbert Schneider, and Shengyong Yang. "Concepts of Artificial Intelligence for Computer-Assisted Drug Discovery". In: *Chem. Rev.* 119.18 (Sept. 2019), pp. 10520–10594. ISSN: 0009-2665. DOI: 10.1021/acs.chemrev.8b00728.

[13] Pascal Friederich, Artem Fediai, Simon Kaiser, Manuel Konrad, Nicole Jung, and Wolfgang Wenzel. "Toward Design of Novel Materials for Organic Electronics". en. In: *Adv. Mater.* 31.26 (2019), p. 1808256. ISSN: 1521-4095. DOI: 10.1002/adma.201808256.

[14] Pascal Friederich, Florian Häse, Jonny Proppe, and Alán Aspuru-Guzik. "Machine-Learned Potentials for next-Generation Matter Simulations". en. In: *Nat. Mater.* 20.6 (June 2021), pp. 750–761. ISSN: 1476-4660. DOI: 10.1038/s41563-020-0777-6.

[15] Gabriel R. Schleder, Antonio C. M. Padilha, Carlos Mera Acosta, Marcio Costa, and Adalberto Fazzio. "From DFT to Machine Learning: Recent Approaches to Materials Science–a Review". en. In: *J. Phys. Mater.* 2.3 (May 2019), p. 032001. ISSN: 2515-7639. DOI: 10.1088/2515-7639/ab084b.

[16] Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. "Machine Learning of Molecular Electronic Properties in Chemical Compound Space". en. In: *New J. Phys.* 15.9 (Sept. 2013), p. 095003. ISSN: 1367-2630. DOI: 10.1088/1367-2630/15/9/095003.

[17] Brian Kolb, Levi C. Lentz, and Alexie M. Kolpak. "Discovering Charge Density Functionals and Structure-Property Relationships with PROPhet: A General Framework for Coupling Machine Learning and First-Principles Methods". en. In: *Sci Rep* 7.1 (Apr. 2017), p. 1192. ISSN: 2045-2322. DOI: 10.1038/s41598-017-01251-z.

[18] Yi Zhou, Jiang Wu, Shuguang Chen, and GuanHua Chen. "Toward the Exact Exchange–Correlation Potential: A Three-Dimensional Convolutional Neural Network Construct". In: *J. Phys. Chem. Lett.* 10.22 (Nov. 2019), pp. 7264–7269. DOI: 10.1021/acs.jpclett.9b02838.

[19] Sergei Manzhos. "Machine Learning for the Solution of the Schrödinger Equation". en. In: *Mach. Learn.: Sci. Technol.* 1.1 (Apr. 2020), p. 013002. ISSN: 2632-2153. DOI: 10.1088/2632-2153/ab7d30.

[20] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. "Big Data Meets Quantum Chemistry Approximations: The Δ-Machine Learning Approach". In: *J. Chem. Theory Comput.* 11.5 (May 2015), pp. 2087–2096. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.5b00099.

[21] Junmian Zhu, Van Quan Vuong, Bobby G. Sumpter, and Stephan Irle. “Artificial Neural Network Correction for Density-Functional Tight-Binding Molecular Dynamics Simulations”. en. In: *MRS Communications* 9.3 (Sept. 2019), pp. 867–873. ISSN: 2159-6867. DOI: 10.1557/mrc.2019.80.

[22] Pavlo O. Dral, O. Anatole von Lilienfeld, and Walter Thiel. “Machine Learning of Parameters for Accurate Semiempirical Quantum Chemical Calculations”. In: *J. Chem. Theory Comput.* 11.5 (May 2015), pp. 2120–2125. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.5b00141.

[23] Patrick Bleiziffer, Kay Schaller, and Sereina Riniker. “Machine Learning of Partial Charges Derived from High-Quality Quantum-Mechanical Calculations”. In: *J. Chem. Inf. Model.* 58.3 (Mar. 2018), pp. 579–590. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.7b00663.

[24] Julian J. Kranz, Maximilian Kubillus, Raghunathan Ramakrishnan, O. Anatole von Lilienfeld, and Marcus Elstner. “Generalized Density-Functional Tight-Binding Repulsive Potentials from Unsupervised Machine Learning”. In: *J. Chem. Theory Comput.* 14.5 (May 2018), pp. 2341–2352. DOI: 10.1021/acs.jctc.7b00933.

[25] Kazuki Niimi, Shoji Shinamura, Itaru Osaka, Eigo Miyazaki, and Kazuo Takimiya. “Dianthra[2,3-b:2′,3′-f]Thieno[3,2-b]Thiophene (DATT): Synthesis, Characterization, and FET Characteristics of New π-Extended Heteroarene with Eight Fused Aromatic Rings”. In: *J. Am. Chem. Soc.* 133.22 (June 2011), pp. 8732–8739. ISSN: 0002-7863. DOI: 10.1021/ja202377m.

[26] Jason D. Myers and Jiangeng Xue. “Organic Semiconductors and Their Applications in Photovoltaic Devices”. In: *Polym. Rev.* 52.1 (Jan. 2012), pp. 1–37. ISSN: 1558-3724. DOI: 10.1080/15583724.2011.644368.

[27] Bernard Kippelen and Jean-Luc Brédas. “Organic Photovoltaics”. In: *Energy Env. Sci* 2.3 (2009), pp. 251–261.

[28] Carsten Deibel and Vladimir Dyakonov. “Polymer–Fullerene Bulk Heterojunction Solar Cells”. In: *Rep Prog Phys* 73.9 (2010), p. 096401.

[29] Neal R Armstrong, Weining Wang, Dana M Alloway, Diogenes Placencia, Erin Ratcliff, and Michael Brumbach. “Organic/Organic Heterojunctions: Organic Light Emitting Diodes and Organic Photovoltaic Devices”. In: *Macromol Rapid Commun* 30.9-10 (2009), pp. 717–731.

[30] Rico Meerheim, BjÖrn Lussem, and Karl Leo. “Efficiency and Stability of Pin Type Organic Light Emitting Diodes for Display and Lighting Applications”. In: *Proc IEEE* 97.9 (2009), pp. 1606–1626.

[31] Christos D Dimitrakopoulos and Patrick RL Malenfant. “Organic Thin Film Transistors for Large Area Electronics”. In: *Adv Mater* 14.2 (2002), pp. 99–117.

[32] Antonio Facchetti. “Semiconductors for Organic Transistors”. In: *Mater Today* 10.3 (2007), pp. 28–37.

[33] Harald Oberhofer, Karsten Reuter, and Jochen Blumberger. "Charge Transport in Molecular Materials: An Assessment of Computational Methods". In: *Chem Rev* 117.15 (2017), pp. 10319–10357.

[34] C. Groves. "Simulating Charge Transport in Organic Semiconductors and Devices: A Review". en. In: *Rep. Prog. Phys.* 80.2 (Dec. 2016), p. 026502. ISSN: 0034-4885. DOI: 10.1088/1361-6633/80/2/026502.

[35] Tomáš Kubař, Rafael Gutiérrez, Ulrich Kleinekathöfer, Gianaurelio Cuniberti, and Marcus Elstner. "Modeling Charge Transport in DNA Using Multi-Scale Methods". In: *Phys Status Solidi B* 250.11 (2013), pp. 2277–2287.

[36] Alexander Heck, Julian J Kranz, Tomáš Kubař, and Marcus Elstner. "Multi-Scale Approach to Non-Adiabatic Charge Transport in High-Mobility Organic Semiconductors". In: *J Chem Theory Comput* 11.11 (2015), pp. 5068–5082.

[37] Julian J. Kranz and Marcus Elstner. "Simulation of Singlet Exciton Diffusion in Bulk Organic Materials". In: *J. Chem. Theory Comput.* 12.9 (2016), pp. 4209–4221. DOI: 10.1021/acs.jctc.6b00235.

[38] Christopher J Cramer. *Essentials of Computational Chemistry: Theories and Models*. Second. John Wiley & Sons, 2013.

[39] Frank Jensen. *Introduction to Computational Chemistry*. John Wiley & Sons, 2017.

[40] Sayan Maity, Beatrix M. Bold, Jigneshkumar Dahyabhai Prajapati, Monja Sokolov, Tomáš Kubař, Marcus Elstner, and Ulrich Kleinekathöfer. "DFTB/MM Molecular Dynamics Simulations of the FMO Light-Harvesting Complex". In: *J. Phys. Chem. Lett.* 11.20 (Oct. 2020), pp. 8660–8667. DOI: 10.1021/acs.jpclett.0c02526.

[41] Seth Difley, Lee-Ping Wang, Sina Yeganeh, Shane R. Yost, and Troy Van Voorhis. "Electronic Properties of Disordered Organic Semiconductors via QM/MM Simulations". In: *Acc. Chem. Res.* 43.7 (July 2010), pp. 995–1004. ISSN: 0001-4842. DOI: 10.1021/ar900246s.

[42] P. Hohenberg and W. Kohn. "Inhomogeneous Electron Gas". In: *Phys. Rev.* 136.3B (Nov. 1964), B864–B871. DOI: 10.1103/PhysRev.136.B864.

[43] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". In: *Phys. Rev.* 140.4A (Nov. 1965), A1133–A1138. DOI: 10.1103/PhysRev.140.A1133.

[44] F. W. Averill and G. S. Painter. "Harris Functional and Related Methods for Calculating Total Energies in Density-Functional Theory". In: *Phys. Rev. B* 41.15 (May 1990), pp. 10344–10353. DOI: 10.1103/PhysRevB.41.10344.

[45] J. Harris. "Simplified Method for Calculating the Energy of Weakly Interacting Fragments". In: *Phys. Rev. B* 31.4 (Feb. 1985), pp. 1770–1779. DOI: 10.1103/PhysRevB.31.1770.

[46] G Seifert, D Porezag, and Th Frauenheim. "Calculations of Molecules, Clusters, and Solids with a Simplified LCAO-DFT-LDA Scheme". In: *Int J Quantum Chem* 58.2 (1996), pp. 185–192.

[47] Thomas A Niehaus and Fabio Della Sala. “Range Separated Functionals in the Density Functional Based Tight-Binding Method: Formalism”. In: *Phys Status Solidi B* 249.2 (2012), pp. 237–244.

[48] Julian J. Kranz, Marcus Elstner, Bálint Aradi, Thomas Frauenheim, Vitalij Lutsker, Adriel Dominguez Garcia, and Thomas A. Niehaus. “Time-Dependent Extension of the Long-Range Corrected Density Functional Based Tight-Binding Method”. In: *J. Chem. Theory Comput.* 13.4 (Apr. 2017), pp. 1737–1747. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.6b01243.

[49] J. C. Slater and G. F. Koster. “Simplified LCAO Method for the Periodic Potential Problem”. In: *Phys. Rev.* 94.6 (June 1954), pp. 1498–1524. DOI: 10.1103/PhysRev.94.1498.

[50] T. A. Niehaus. “Approximate Time-Dependent Density Functional Theory”. en. In: *Journal of Molecular Structure: THEOCHEM*. Time-Dependent Density-Functional Theory for Molecules and Molecular Solids 914.1 (Nov. 2009), pp. 38–49. ISSN: 0166-1280. DOI: 10.1016/j.theochem.2009.04.034.

[51] Julian J Kranz, Marcus Elstner, Bálint Aradi, Thomas Frauenheim, Vitalij Lutsker, Adriel Dominguez Garcia, and Thomas A Niehaus. “Time-Dependent Extension of the Long-Range Corrected Density Functional Based Tight-Binding Method”. In: *J Chem Theory Comput* 13.4 (2017), pp. 1737–1747.

[52] Monja Sokolov, Beatrix M. Bold, Julian J. Kranz, Sebastian Höfener, Thomas A. Niehaus, and Marcus Elstner. “Analytical Time-Dependent Long-Range Corrected Density Functional Tight Binding (TD-LC-DFTB) Gradients in DFTB+: Implementation and Benchmark for Excited-State Geometries and Transition Energies”. In: *J. Chem. Theory Comput.* 17.4 (Apr. 2021), pp. 2266–2282. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.1c00095.

[53] Beatrix Mirinda Bold, Monja Sokolov, Sayan Maity, Marius Wanko, Philipp Michael Dohmen, Julian Joan Kranz, Ulrich Kleinekathöfer, Sebastian Höfener, and Marcus Elstner. “Benchmark and Performance of Long-Range Corrected Time-Dependent Density Functional Tight Binding (LC-TD-DFTB) on Rhodopsins and Light-Harvesting Complexes”. In: *Phys Chem Chem Phys* (2020), in press.

[54] Ryogo Kubo, Morikazu Toda, and Natsuki Hashitsume. *Statistical Physics II: Nonequilibrium Statistical Mechanics*. Vol. 31. Springer Science & Business Media, 2012.

[55] Paul Ehrenfest. “Bemerkung Über Die Angenäherte Gültigkeit Der Klassischen Mechanik Innerhalb Der Quantenmechanik”. In: *Z Phys* 45 (1927), pp. 455–457.

[56] AD McLachlan. “A Variational Solution of the Time-Dependent Schrodinger Equation”. In: *Mol Phys* 8.1 (1964), pp. 39–44.

[57] John C. Tully. “Mixed Quantum–Classical Dynamics”. en. In: *Faraday Discuss.* 110.0 (Jan. 1998), pp. 407–419. ISSN: 1364-5498. DOI: 10.1039/A801824C.

[58] Clarence Zener. “Non-Adiabatic Crossing of Energy Levels”. In: *Proc R Soc Lond. Ser A* 137.833 (Sept. 1932), pp. 696–702. ISSN: 1364-5021. DOI: 10.1098/rspa.1932.0165.

[59] Curt Wittig. "The Landau-Zener Formula". In: *J Phys Chem B* 109.17 (May 2005), pp. 8428–8430. ISSN: 1520-6106. DOI: 10.1021/jp040627u.

[60] John C Tully. "Molecular Dynamics with Electronic Transitions". In: *J Chem Phys* 93.2 (July 1990), pp. 1061–1071. ISSN: 0021-9606. DOI: 10.1063/1.459170.

[61] Antoine Carof, Samuele Giannini, and Jochen Blumberger. "Detailed Balance, Internal Consistency, and Energy Conservation in Fragment Orbital-Based Surface Hopping". In: *J. Chem. Phys.* 147.21 (Dec. 2017), p. 214113. ISSN: 0021-9606. DOI: 10.1063/1.5003820.

[62] Michael D. Hack, Ahren W. Jasper, Yuri L. Volobuev, David W. Schwenke, and Donald G. Truhlar. "Quantum Mechanical and Quasiclassical Trajectory Surface Hopping Studies of the Electronically Nonadiabatic Predissociation of the Ã State of NaH2". In: *J. Phys. Chem. A* 103.32 (Aug. 1999), pp. 6309–6326. ISSN: 1089-5639. DOI: 10.1021/jp9912049.

[63] Weiwei Xie, Marin Sapunar, Na\dja Došlić, Matthieu Sala, and Wolfgang Domcke. "Assessing the Performance of Trajectory Surface Hopping Methods: Ultrafast Internal Conversion in Pyrazine". In: *J Chem Phys* 150.15 (2019), p. 154119.

[64] Weiwei Xie and Wolfgang Domcke. "Accuracy of Trajectory Surface-Hopping Methods: Test for a Two-Dimensional Model of the Photodissociation of Phenol". In: *J Chem Phys* 147.18 (Nov. 2017), p. 184114. ISSN: 0021-9606. DOI: 10.1063/1.5006788.

[65] Weiwei Xie, Daniel Holub, Tomáš Kubař, and Marcus Elstner. "Performance of Mixed Quantum-Classical Approaches on Modeling the Crossover from Hopping to Bandlike Charge Transport in Organic Semiconductors". In: *J. Chem. Theory Comput.* 16.4 (Apr. 2020), pp. 2071–2084. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.9b01271.

[66] Adam Kubas, Felix Hoffmann, Alexander Heck, Harald Oberhofer, Marcus Elstner, and Jochen Blumberger. "Electronic Couplings for Molecular Charge Transfer: Benchmarking CDFT, FODFT, and FODFTB against High-Level Ab Initio Calculations". In: *J Chem Phys* 140.10 (2014), p. 104105.

[67] Adam Kubas, Fruzsina Gajdos, Alexander Heck, Harald Oberhofer, Marcus Elstner, and Jochen Blumberger. "Electronic Couplings for Molecular Charge Transfer: Benchmarking CDFT, FODFT and FODFTB against High-Level Ab Initio Calculations. II". In: *Phys Chem Chem Phys* 17.22 (2015), pp. 14342–14354.

[68] Tomáš Kubař, P Benjamin Woiczikowski, Gianaurelio Cuniberti, and Marcus Elstner. "Efficient Calculation of Charge-Transfer Matrix Elements for Hole Transfer in DNA". In: *J Phys Chem B* 112.26 (2008), pp. 7937–7947.

[69] Tomáš Kubař and Marcus Elstner. "A Hybrid Approach to Simulation of Electron Transfer in Complex Molecular Systems". In: *J R Soc Interface* 10.87 (2013), p. 20130415.

[70] Alexander Heck, Julian J Kranz, and Marcus Elstner. "Simulation of Temperature-Dependent Charge Transport in Organic Semiconductors with Various Degrees of Disorder". In: *J Chem Theory Comput* 12.7 (2016), pp. 3087–3096.

[71] Kazuo Kitaura, Eiji Ikeo, Toshio Asada, Tatsuya Nakano, and Masami Uebayasi. “Fragment Molecular Orbital Method: An Approximate Computational Method for Large Molecules”. In: *Chem Phys Lett* 313.3-4 (1999), pp. 701–706.

[72] Jacov Frenkel. “On the Transformation of Light into Heat in Solids. I”. In: *Phys Rev* 37.1 (1931), p. 17.

[73] Jacov Frenkel. “On the Transformation of Light into Heat in Solids. II”. In: *Phys Rev* 37.10 (1931), p. 1276.

[74] Volkhard May and Oliver Kühn. *Charge and Energy Transfer Dynamics in Molecular Systems*. Vol. 2. Wiley Online Library, 2011.

[75] Nils Schieschke, Beatrix M. Bold, Philipp M. Dohmen, Daniel Wehl, Marvin Hoffmann, Andreas Dreuw, Marcus Elstner, and Sebastian Höfener. “Geometry Dependence of Excitonic Couplings and the Consequences for Configuration-Space Sampling”. en. In: *J. Comput. Chem.* 42.20 (2021), pp. 1402–1418. ISSN: 1096-987X. DOI: 10.1002/jcc.26552.

[76] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009.

[77] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. “Dimensionality Reduction: A Comparative”. In: *J Mach Learn Res* 10.66-71 (2009), p. 13.

[78] C. O. S. Sorzano, J. Vargas, and A. Pascual Montano. “A Survey of Dimensionality Reduction Techniques”. In: *ArXiv14032877 Cs Q-Bio Stat* (Mar. 2014). arXiv: 1403.2877 [cs, q-bio, stat].

[79] David Harville. “Extension of the Gauss-Markov Theorem to Include the Estimation of Random Effects”. In: *Ann. Stat.* 4.2 (1976), pp. 384–395. ISSN: 0090-5364.

[80] Christopher M Bishop. *Pattern Recognition and Machine Learning*. springer, 2006.

[81] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[82] J Baker, M Douma, and S Kotochigova. *The 2018 CODATA Recommended Values of the Fundamental Physical Constants*. http://physics.nist.gov/constants. 2020.

[83] William M Haynes. *CRC Handbook of Chemistry and Physics*. Ninety-fifth. CRC press, 2014.

[84] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. “Network Dissection: Quantifying Interpretability of Deep Visual Representations”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6541–6549.

[85] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. “The Building Blocks of Interpretability”. en. In: *Distill* 3.3 (Mar. 2018), e10. ISSN: 2476-0757. DOI: 10.23915/distill.00010.

[86] Albert P. Bartók, Risi Kondor, and Gábor Csányi. “On Representing Chemical Environments”. In: *Phys. Rev. B* 87.18 (May 2013), p. 184115. DOI: 10.1103/PhysRevB.87.184115.

[87] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. "Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning". In: *Phys. Rev. Lett.* 108.5 (Jan. 2012), p. 058301. DOI: 10.1103/PhysRevLett.108.058301.

[88] Bing Huang and O. Anatole von Lilienfeld. "Communication: Understanding Molecular Representations in Machine Learning: The Role of Uniqueness and Target Similarity". In: *J. Chem. Phys.* 145.16 (Oct. 2016), pp. 161102–161102. DOI: 10.1063/1.4964627.

[89] Jörg Behler and Michele Parrinello. "Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces". In: *Phys. Rev. Lett.* 98.14 (Apr. 2007), p. 146401. DOI: 10.1103/PhysRevLett.98.146401.

[90] Felix A. Faber, Anders S. Christensen, Bing Huang, and O. Anatole von Lilienfeld. "Alchemical and Structural Distribution Based Representation for Improved QML". In: (Dec. 2017). DOI: 10.1063/1.5020710.

[91] Anders S. Christensen, Lars A. Bratholm, Felix A. Faber, David R. Glowacki, and O. Anatole von Lilienfeld. "FCHL Revisited: Faster and More Accurate Quantum Machine Learning". In: *ArXiv190901946 Phys.* (Sept. 2019). arXiv: 1909.01946 [physics].

[92] Felix A. Faber, Luke Hutchison, Bing Huang, Justin Gilmer, Samuel S. Schoenholz, George E. Dahl, Oriol Vinyals, Steven Kearnes, Patrick F. Riley, and O. Anatole von Lilienfeld. "Prediction Errors of Molecular Machine Learning Models Lower than Hybrid DFT Error". In: *J. Chem. Theory Comput.* 13.11 (Nov. 2017), pp. 5255–5264. DOI: 10.1021/acs.jctc.7b00577.

[93] Jonas Lederer, Waldemar Kaiser, Alessandro Mattoni, and Alessio Gagliardi. "Machine Learning–Based Charge Transport Computation for Pentacene". In: *Adv. Theory Simul.* 2.2 (Feb. 2019), pp. 1800136–1800136. DOI: 10.1002/adts.201800136.

[94] Matthias Rupp. "Machine Learning for Quantum Mechanics in a Nutshell". In: *International Journal of Quantum Chemistry* 115.16 (Aug. 2015), pp. 1058–1073. ISSN: 0020-7608. DOI: 10.1002/qua.24954.

[95] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization". In: *J. Mach. Learn. Res.* 18.1 (2017), pp. 6765–6816.

[96] Félix Musil, Sandip De, Jack Yang, Joshua E. Campbell, Graeme M. Day, and Michele Ceriotti. "Machine Learning for the Structure–Energy–Property Landscapes of Molecular Crystals". en. In: *Chem. Sci.* 9.5 (Jan. 2018), pp. 1289–1300. ISSN: 2041-6539. DOI: 10.1039/C7SC04665K.

[97] Onur Çaylak, Anil Yaman, and Björn Baumeier. "Evolutionary Approach to Constructing a Deep Feedforward Neural Network for Prediction of Electronic Coupling Elements in Molecular Materials". In: *J. Chem. Theory Comput.* 15.3 (Mar. 2019), pp. 1777–1784. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.8b01285.

[98] Florian Häse, Stéphanie Valleau, Edward Pyzer-Knapp, and Alán Aspuru-Guzik. "Machine Learning Exciton Dynamics". In: *Chem. Sci.* 7.8 (July 2016), pp. 5139–5147. DOI: 10.1039/C5SC04786B.

[99] Florian Häse, Christoph Kreisbeck, and Alán Aspuru-Guzik. "Machine Learning for Quantum Dynamics: Deep Learning of Excitation Energy Transfer Properties". en. In: *Chem. Sci.* 8.12 (Nov. 2017), pp. 8419–8426. ISSN: 2041-6539. DOI: 10.1039/C7SC03542J.

[100] Chun-I Wang, Mac Kevin E. Braza, Gil C. Claudio, Ricky B. Nellas, and Chao-Ping Hsu. "Machine Learning for Predicting Electron Transfer Coupling". In: *J. Phys. Chem. A* 123.36 (Sept. 2019), pp. 7792–7802. ISSN: 1089-5639. DOI: 10.1021/acs.jpca.9b04256.

[101] Mila Krämer, Philipp M. Dohmen, Weiwei Xie, Daniel Holub, Anders S. Christensen, and Marcus Elstner. "Charge and Exciton Transfer Simulations Using Machine-Learned Hamiltonians". In: *J. Chem. Theory Comput.* 16.7 (July 2020), pp. 4061–4070. ISSN: 1549-9618. DOI: 10.1021/acs.jctc.0c00246.

[102] Junmei Wang, Romain M. Wolf, James W. Caldwell, Peter A. Kollman, and David A. Case. "Development and Testing of a General Amber Force Field". en. In: *J. Comput. Chem.* 25.9 (2004), pp. 1157–1174. ISSN: 1096-987X. DOI: 10.1002/jcc.20035.

[103] Junmei Wang, Wei Wang, Peter A. Kollman, and David A. Case. "Automatic Atom Type and Bond Type Perception in Molecular Mechanical Calculations". In: *J. Mol. Graph. Model.* 25.2 (2006), pp. 247–260. ISSN: 1093-3263. DOI: 10.1016/j.jmgm.2005.12.005.

[104] U. Chandra Singh and Peter A. Kollman. "An Approach to Computing Electrostatic Charges for Molecules". en. In: *J. Comput. Chem.* 5.2 (1984), pp. 129–145. ISSN: 1096-987X. DOI: 10.1002/jcc.540050204.

[105] Brent H. Besler, Kenneth M. Merz, and Peter A. Kollman. "Atomic Charges Derived from Semiempirical Methods". en. In: *J. Comput. Chem.* 11.4 (1990), pp. 431–439. ISSN: 1096-987X. DOI: 10.1002/jcc.540110404.

[106] G. A. Petersson, Andrew Bennett, Thomas G. Tensfeldt, Mohammad A. Al-Laham, William A. Shirley, and John Mantzaris. "A Complete Basis Set Model Chemistry. I. The Total Energies of Closed-Shell Atoms and Hydrides of the First-Row Elements". In: *J Chem Phys* 89.4 (1988), pp. 2193–2218. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.455064.

[107] G. A. Petersson and Mohammad A. Al-Laham. "A Complete Basis Set Model Chemistry. II. Open-Shell Systems and the Total Energies of the First-Row Atoms". In: *J. Chem. Phys.* 94.9 (1991), pp. 6081–6090. ISSN: 0021-9606, 1089-7690. DOI: 10.1063/1.460447.

[108] Michael J. Frisch et al. *Gaussian 09*. Wallingford, CT, USA: Gaussian, Inc., 2009.

[109] H. J. C. Berendsen, D. van der Spoel, and R. van Drunen. "GROMACS: A Message-Passing Parallel Molecular Dynamics Implementation". In: *Comput. Phys. Commun.* 91.1 (Sept. 1995), pp. 43–56. ISSN: 0010-4655. DOI: 10.1016/0010-4655(95)00042-E.

[110] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. "GROMACS: High Performance Molecular Simulations through Multi-Level Parallelism from Laptops to Supercomputers". In: *SoftwareX* 1–2 (Sept. 2015), pp. 19–25. ISSN: 2352-7110. DOI: 10.1016/j.softx.2015.06.001.

[111] R. Mason. "The Crystallography of Anthracene at 95$\^\circ$K and 290$\^\circ$K". In: *Acta Crystallogr.* 17.5 (May 1964), pp. 547–555. DOI: 10.1107/S0365110X64001281.

[112] D. J. Evans and B. L. Holian. "The Nose–Hoover Thermostat". en. In: *J. Chem. Phys.* 83.8 (1985), p. 4069. ISSN: 00219606. DOI: 10.1063/1.449071.

[113] S Fratini, S Ciuchi, D Mayou, G Trambly De Laissardière, and A Troisi. "A Map of High-Mobility Molecular Semiconductors". In: *Nat Mater* 16.10 (2017), pp. 998–1002.

[114] Anders S. Christensen, Felix A. Faber, Bing Huang, Lars A. Bratholm, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. *QML: A Python Toolkit for Quantum Machine Learning*. Jan. 2017.

[115] Sandip De, Albert P. Bartók, Gábor Csányi, and Michele Ceriotti. "Comparing Molecules and Solids across Structural and Alchemical Space". In: *Phys. Chem. Chem. Phys.* 18.20 (May 2016), pp. 13754–13769. DOI: 10.1039/C6CP00415F.

[116] Felix A Faber, Anders S Christensen, Bing Huang, and O Anatole Von Lilienfeld. "Alchemical and Structural Distribution Based Representation for Universal Quantum Machine Learning". In: *J Chem Phys* 148.24 (2018), p. 241717.

[117] Fabian Pedregosa et al. "Scikit-Learn: Machine Learning in Python". In: *J. Mach. Learn. Res.* 12 (Oct. 2011), pp. 2825–2830. ISSN: 1533-7928.

[118] Norbert Karl and Jörg Marktanner. "Electron and Hole Mobilities in High Purity Anthracene Single Crystals". In: *J Mol Cryst Liq Cryst Sci Technol Sect A* 355.1 (2001), pp. 149–173.

[119] Szilárd Páll, Mark James Abraham, Carsten Kutzner, Berk Hess, and Erik Lindahl. "Tackling Exascale Software Challenges in Molecular Dynamics Simulations with GROMACS". In: *International Conference on Exascale Applications and Software*. Springer, 2014, pp. 3–27.

[120] Vera Stehr, Reinhold F Fink, Bernd Engels, Jens Pflaum, and Carsten Deibel. "Singlet Exciton Diffusion in Organic Crystals Based on Marcus Transfer Rates". In: *J Chem Theory Comput* 10.3 (2014), pp. 1242–1255.

[121] Rudolph A Marcus. "On the Theory of Oxidation-Reduction Reactions Involving Electron Transfer. I". In: *J Chem Phys* 24.5 (1956), pp. 966–978.

[122] Rudolph A Marcus. "Electron Transfer Reactions in Chemistry. Theory and Experiment". In: *Rev Mod Phys* 65.3 (1993), p. 599.

[123] Sopheak Meangkhorl Seng. "Prediction of Exciton Transfer Parameters of LH2 from Rhodoblastus Acidophilus with Machine Learning". Bachelor's Thesis. Karlsruhe: Karlsruhe Institute of Technology, 2020.

[124] Joseph E. Subotnik, Amber Jain, Brian Landry, Andrew Petit, Wenjun Ouyang, and Nicole Bellonzi. "Understanding the Surface Hopping View of Electronic Transitions and Decoherence". In: *Annu Rev Phys Chem* 67.1 (May 2016), pp. 387–417. ISSN: 0066-426X. DOI: 10.1146/annurev-physchem-040215-112245.

[125] Samuele Giannini, Antoine Carof, and Jochen Blumberger. "Crossover from Hopping to Band-Like Charge Transport in an Organic Semiconductor Model: Atomistic Nonadiabatic Molecular Dynamics Simulation". In: *J Phys Chem Lett* 9.11 (2018), pp. 3116–3123.

[126] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015.

[127] The Theano Development Team et al. "Theano: A Python Framework for Fast Computation of Mathematical Expressions". In: *ArXiv160502688 Cs* (May 2016). arXiv: 1605.02688 [cs].

[128] Julia Westermayr, Michael Gastegger, Maximilian F. S. J. Menger, Sebastian Mai, Leticia González, and Philipp Marquetand. "Machine Learning Enables Long Time Scale Molecular Photodynamics Simulations". en. In: *Chem. Sci.* 10.35 (Sept. 2019), pp. 8100–8107. ISSN: 2041-6539. DOI: 10.1039/C9SC01742A.

[129] Julia Westermayr and Philipp Marquetand. "Machine Learning and Excited-State Molecular Dynamics". en. In: *Mach. Learn.: Sci. Technol.* 1.4 (Sept. 2020), p. 043001. ISSN: 2632-2153. DOI: 10.1088/2632-2153/ab9c3e.

[130] Julia Westermayr, Michael Gastegger, and Philipp Marquetand. "Combining SchNet and SHARC: The SchNarc Machine Learning Approach for Excited-State Dynamics". In: *J. Phys. Chem. Lett.* 11.10 (May 2020), pp. 3828–3834. DOI: 10.1021/acs.jpclett.0c00527.

[131] Jingbai Li, Patrick Reiser, André Eberhard, Pascal Friederich, and Steven Lopez. "Nanosecond Photodynamics Simulations of a Cis-Trans Isomerization Are Enabled by Machine Learning". en. In: (Oct. 2020). DOI: 10.26434/chemrxiv.13047863.v1.

[132] Jingbai Li, Patrick Reiser, Benjamin R. Boswell, André Eberhard, Noah Z. Burns, Pascal Friederich, and Steven A. Lopez. "Automatic Discovery of Photoisomerization Mechanisms with Nanosecond Machine Learning Photodynamics Simulations". en. In: *Chem. Sci.* 12.14 (2021), pp. 5302–5314. DOI: 10.1039/D0SC05610C.

[133] Zhenwei Li, James R. Kermode, and Alessandro De Vita. "Molecular Dynamics with On-the-Fly Machine Learning of Quantum-Mechanical Forces". In: *Phys. Rev. Lett.* 114.9 (Mar. 2015), p. 096405. DOI: 10.1103/PhysRevLett.114.096405.

[134] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller. "SchNet – A Deep Learning Architecture for Molecules and Materials". In: *J. Chem. Phys.* 148.24 (Mar. 2018), p. 241722. ISSN: 0021-9606. DOI: 10.1063/1.5019779.

[135] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *ArXiv14126980 Cs* (Jan. 2017). arXiv: 1412.6980 [cs].

[136] Vladimir I. Novoderezhkin, Roberta Croce, Md Wahadoszamen, Iryna Polukhina, Elisabet Romero, and Rienk van Grondelle. "Mixing of Exciton and Charge-Transfer States in Light-Harvesting Complex Lhca4". en. In: *Phys. Chem. Chem. Phys.* 18.28 (2016), pp. 19368–19377. DOI: 10.1039/C6CP02225A.

[137] Flurin D. Eisner, Mohammed Azzouzi, Zhuping Fei, Xueyan Hou, Thomas D. Anthopoulos, T. John S. Dennis, Martin Heeney, and Jenny Nelson. "Hybridization of Local Exciton and Charge-Transfer States Reduces Nonradiative Voltage Losses in Organic Solar Cells". In: *J. Am. Chem. Soc.* 141.15 (Apr. 2019), pp. 6362–6374. ISSN: 0002-7863. DOI: 10.1021/jacs.9b01465.

List of Figures

List of Tables

A. Appendix

A.1. Synthetic Data and Code for the Photoelectric Effect Example

```
#!/usr/bin/env python

import numpy as np
import matplotlib.pyplot as plt
from numpy.random import default_rng
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_absolute_error
rg =default_rng(42)

phi = 2.36 #eV (sodium)
npoints = 50
h_true = 4.136e-15 # eVs
x = np.linspace(6e14, 1e15, npoints)
sign = rg.choice([-1,1], size=npoints)
noisy = np.random.normal(0, .05, x.shape)
f = (h_true*x - phi)+sign*noisy
print(x)
print(f)
lin = LinearRegression()
lin.fit(x.reshape(-1,1), f)
p = lin.predict(x.reshape(-1,1))
m = lin.coef_
b = lin.intercept_
print(m, b)
rss = ((f-p)**2).sum()
print(rss)
r2 = r2_score(f,p)
print(r2)

ntest = 3*npoints

sign = rg.choice([-1,1], size=ntest)
bigx = np.linspace(6e14, 1e15, ntest)
noisy = np.random.normal(0, 1e13, bigx.shape)
```

```
xtest = bigx+noisy
yref = (h_true*xtest - phi)
ypred = lin.predict(xtest.reshape(-1,1))

r2_pred = r2_score(yref, ypred)
mae = mean_absolute_error(yref, ypred)
print(r2_pred, mae)
```

A.2. Kernel Ridge Regression for Charge and Exciton Transfer

Detailed Information on Trained Models

Charge transfer

dataset	n_{train}	λ	σ	R^2	MAE (eV)	Variance	max. error (eV)	rel. error (%)
short	100	10^{-6}	15	0.49	1.1×10^{-2}	1.1×10^{-4}	1.2×10^{-1}	35
	1000	10^{-3}	10	0.83	6.0×10^{-3}	3.5×10^{-5}	7.4×10^{-2}	20
	5000	10^{-2}	10	0.91	4.1×10^{-3}	1.9×10^{-5}	6.9×10^{-2}	14
	10000	10^{-2}	10	0.93	3.7×10^{-3}	1.6×10^{-5}	8.1×10^{-2}	12
	25000	10^{-2}	5	0.95	3.1×10^{-3}	1.0×10^{-5}	6.7×10^{-2}	10
long	100	10^{-2}	10	0.49	3.3×10^{-3}	8.5×10^{-5}	1.2×10^{-1}	58
	1000	10^{-6}	5	0.79	2.0×10^{-3}	3.4×10^{-5}	1.1×10^{-1}	34
	5000	10^{-2}	10	0.93	1.3×10^{-3}	1.2×10^{-5}	6.5×10^{-2}	23
	10000	10^{-2}	10	0.95	1.1×10^{-3}	8.9×10^{-6}	6.0×10^{-2}	19
	25000	10^{-2}	10	0.97	8.7×10^{-4}	6.0×10^{-6}	6.3×10^{-2}	15
full	100	10^{-6}	5	0.53	3.2×10^{-3}	8.3×10^{-5}	1.3×10^{-1}	57
	1000	10^{-2}	10	0.85	1.8×10^{-3}	2.7×10^{-5}	9.4×10^{-2}	32
	5000	10^{-2}	5	0.92	1.3×10^{-3}	1.4×10^{-5}	7.5×10^{-2}	23
	10000	10^{-2}	10	0.95	1.1×10^{-3}	9.0×10^{-6}	6.8×10^{-2}	19
	25000	10^{-2}	10	0.96	8.8×10^{-4}	6.3×10^{-6}	7.1×10^{-2}	15
sites	100	10^{-4}	50	0.97	6.8×10^{-3}	2.9×10^{-5}	4.6×10^{-2}	0.12
	1000	10^{-6}	50	0.99	4.1×10^{-3}	1.1×10^{-5}	3.6×10^{-2}	0.070
	5000	10^{-6}	25	1.00	2.5×10^{-3}	4.3×10^{-6}	2.8×10^{-2}	0.043
	10000	10^{-6}	10	1.00	1.8×10^{-3}	2.2×10^{-6}	1.6×10^{-2}	0.030
	25000	10^{-6}	10	1.00	1.2×10^{-3}	1.0×10^{-6}	1.4×10^{-2}	0.020

Table A.1.: Grid search results for CT models. λ and σ are the KRR hyperparameters, R^2 is the coefficient of determination as calculated during training. Mean absolute and maximum errors calculated on test set of 60000 held out data points.

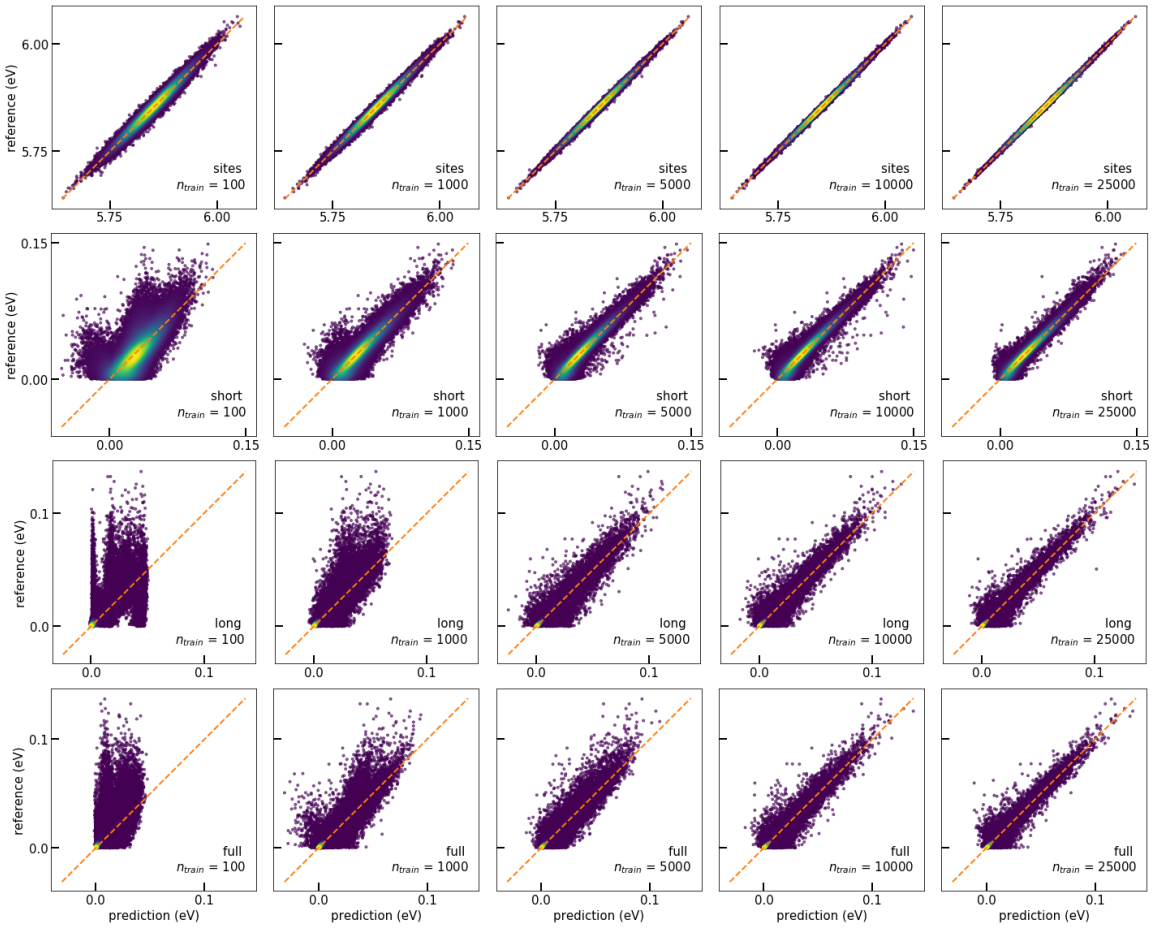


Figure A.1.: Prediction vs reference CT couplings for the 60000 data points in the test set for all models and training set sizes.

Exciton Transfer

dataset	n_{train}	λ	σ	R^2	MAE (eV)	Variance	max. error (eV)	rel. error %
short	100	5×10^{-10}	20	0.74	5.1×10^{-3}	1.7×10^{-5}	3.5×10^{-2}	23
	1000	10^{-2}	15	0.93	2.4×10^{-3}	5.4×10^{-6}	3.6×10^{-2}	11
	5000	10^{-4}	25	0.96	1.6×10^{-3}	3.3×10^{-6}	3.8×10^{-2}	7.1
	10000	10^{-4}	30	0.97	1.4×10^{-3}	2.7×10^{-6}	3.6×10^{-2}	6.0
	25000	10^{-3}	20	0.98	1.1×10^{-3}	2.3×10^{-6}	3.3×10^{-2}	5.0
long	100	10^{-2}	15	0.78	3.1×10^{-3}	9.9×10^{-6}	3.4×10^{-2}	26
	1000	10^{-2}	10	0.88	2.2×10^{-3}	5.2×10^{-6}	2.6×10^{-2}	18
	5000	5×10^{-10}	7.5	0.96	1.3×10^{-3}	2.1×10^{-6}	2.5×10^{-2}	11
	10000	10^{-3}	7.5	0.97	1.1×10^{-3}	1.7×10^{-6}	2.7×10^{-2}	8.9
	25000	10^{-3}	7.5	0.98	8.2×10^{-4}	1.3×10^{-6}	2.4×10^{-2}	6.8
full	100	10^{-2}	20	0.83	1.5×10^{-3}	5.5×10^{-6}	3.8×10^{-2}	38
	1000	10^{-2}	7.5	0.88	1.2×10^{-3}	3.8×10^{-6}	3.1×10^{-2}	30
	5000	10^{-2}	5	0.93	9.2×10^{-4}	2.4×10^{-6}	3.1×10^{-2}	24
	10000	10^{-2}	7.5	0.96	7.6×10^{-4}	1.2×10^{-6}	2.5×10^{-2}	20
	25000	10^{-2}	7.5	0.98	6.0×10^{-4}	7.2×10^{-7}	2.5×10^{-2}	16
sites	100	10^{-2}	20	0.98	1.5×10^{-2}	1.5×10^{-4}	1.1×10^{-1}	0.45
	1000	10^{-5}	20	0.99	7.4×10^{-3}	3.8×10^{-5}	6.9×10^{-2}	0.22
	5000	10^{-6}	20	1.00	3.7×10^{-3}	1.0×10^{-5}	3.7×10^{-2}	0.11
	10000	10^{-9}	20	1.00	2.0×10^{-3}	2.7×10^{-6}	1.9×10^{-2}	0.059
	25000	10^{-9}	20	1.00	8.7×10^{-4}	5.4×10^{-7}	1.2×10^{-2}	0.026

Table A.2.: Grid search results for ET models.

Details on Results for Charge Propagation

		Exp.	DFTB	ML-100	ML-1000	ML-5000	ML-10000	ML-25000
MFE	*a*	1.1	1.4	3.2	1.5	1.7	1.7	1.6
	b	2.9	3.4	3.7	3.7	3.5	3.1	3.2
BC-FSSH	*a*	1.1	8.0	10	7.7	8.7	7.6	8.1
	b	2.9	13	14	11	11	12	11

Table A.3.: Hole mobility in $cm^2\,V^{-1}\,s^{-1}$ as calculated from the averaged MSD in *a*- and *b*-direction using the MFE and BC-FSSH methods for charge propagation with DFTB and ML-models with various training-size.

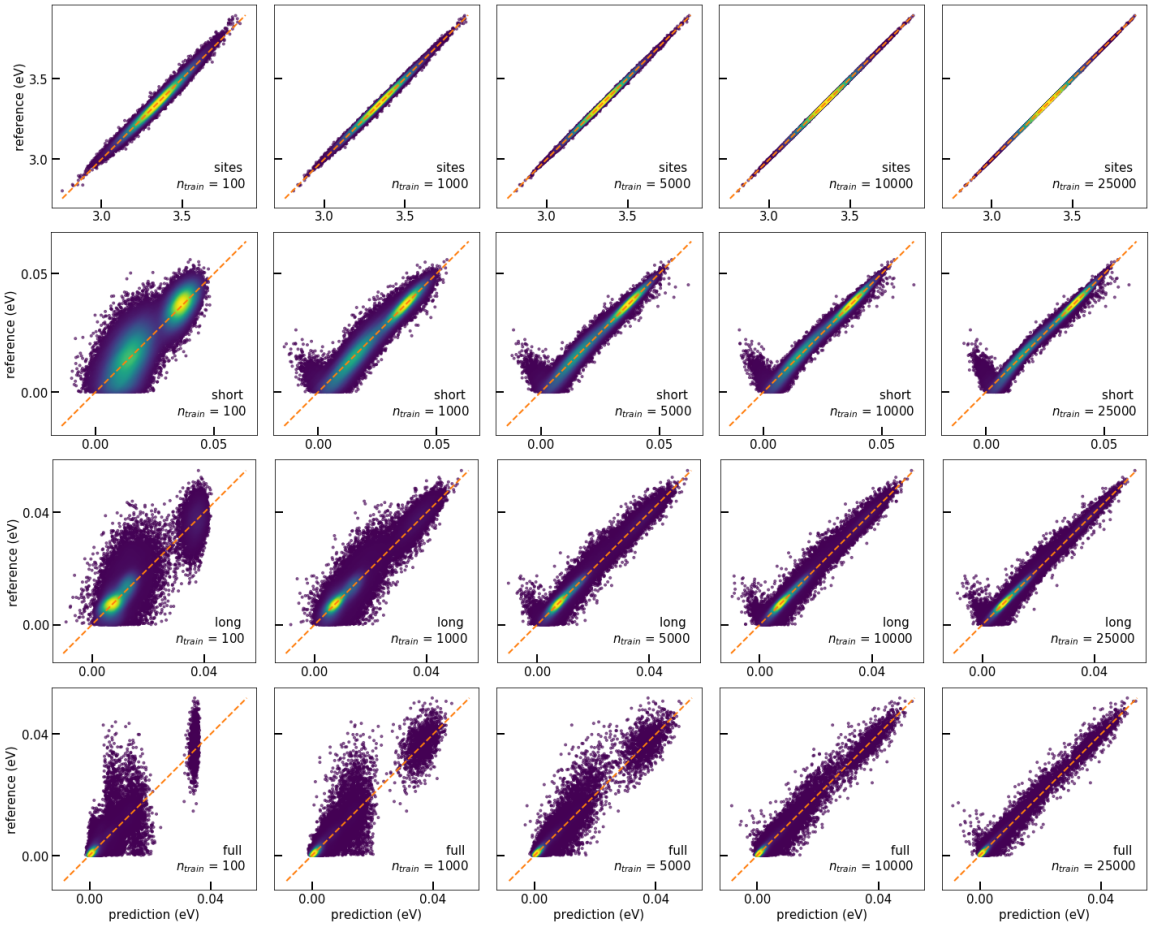


Figure A.2.: Prediction vs reference ET couplings for the 60000 data points in the test set for all models and training set sizes.

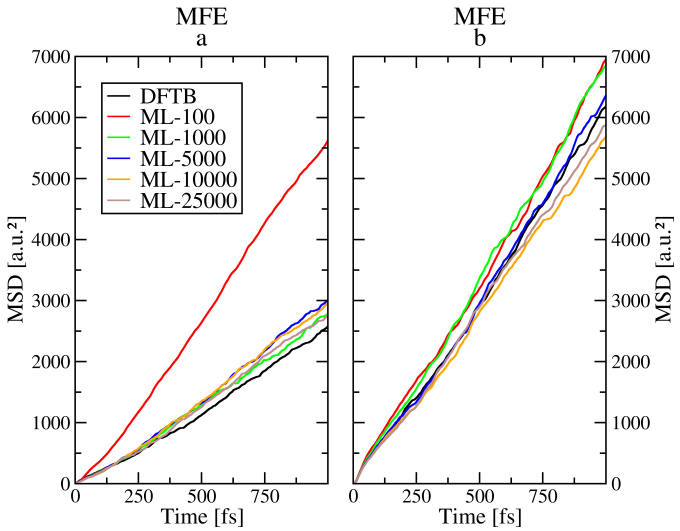


Figure A.3.: Time evolution of the averaged MSD in *a*- and *b*-direction using the MFE method for charge propagation with DFTB and ML-models with various training-size.

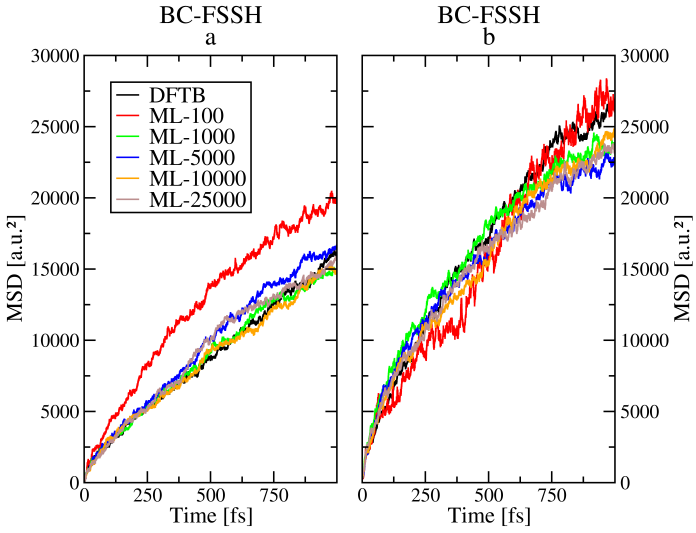


Figure A.4.: Time evolution of the averaged MSD in *a*- and *b*-direction using the BC-FSSH method for charge propagation with DFTB and ML-models with various training-size.

Details on Results for Exciton Propagation

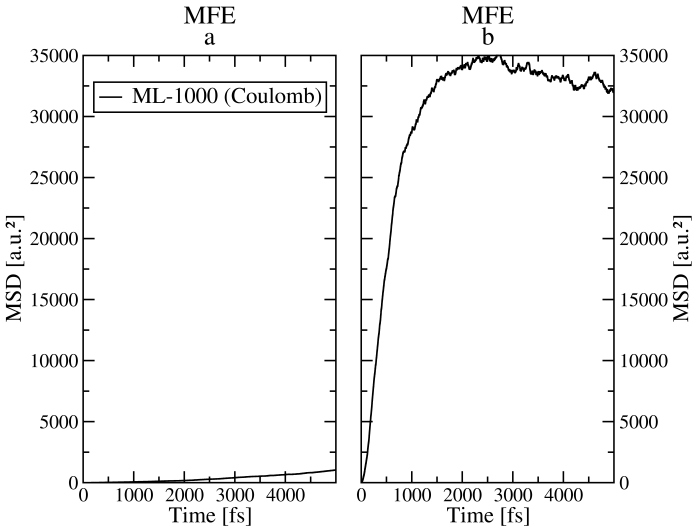


Figure A.5.: Time evolution of the averaged MSD in *a*- and *b*-direction using the MFE method for exciton propagation with an ML-model with a training-size of 1000.

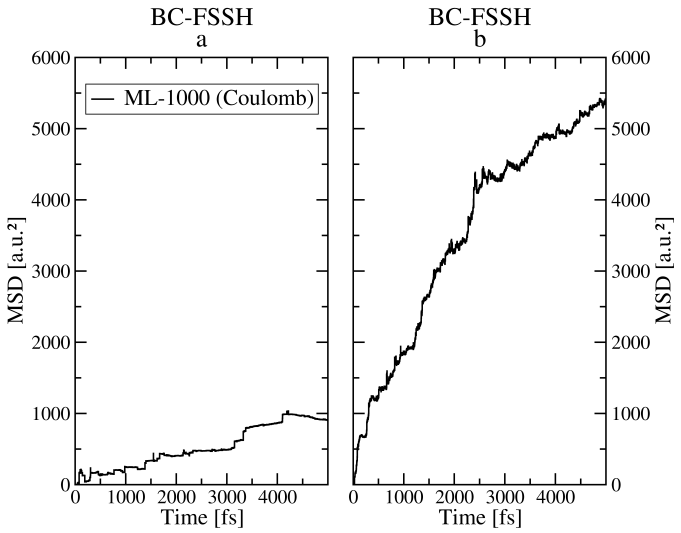
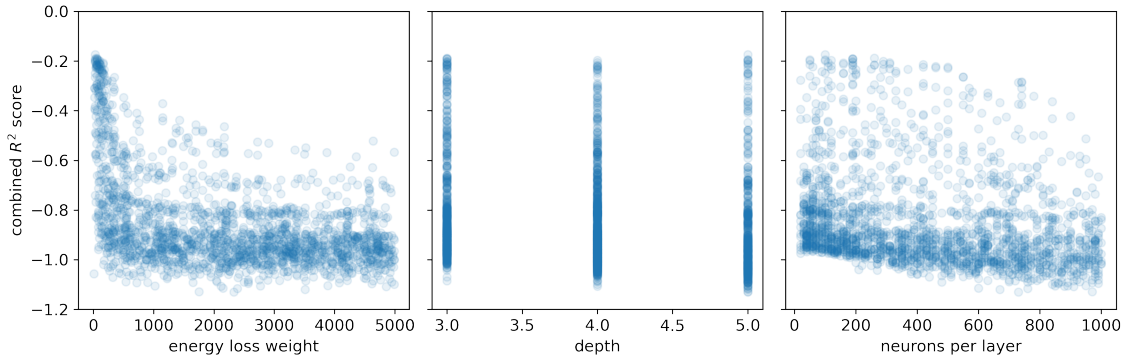


Figure A.6.: Time evolution of the averaged MSD in *a*- and *b*-direction using the BC-FSSH method for exciton propagation with an ML-model with a training-size of 1000.

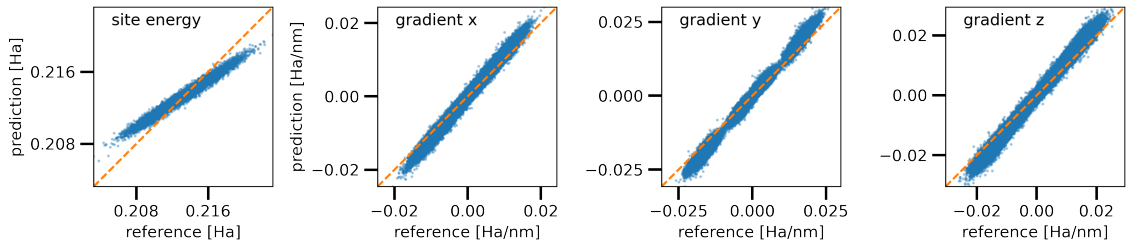
A.3. Hyperparameter Searches using the Hyperband algorithm

parameter	search space
depth	3, 4 and 5
neurons per layer	20 to 500
regularization	L_1, L_2
energy loss weight	1 to 1000
force loss weight	1 to 1000
learning rate	1×10^{-3}, 5×10^{-4} and 1×10^{-4}

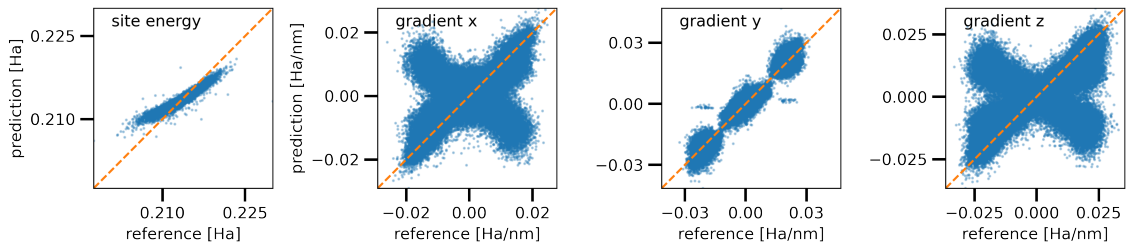
Table A.4.: Hyperparameter search space for the delta diagonal model.



(a) off-diagonal elements



(b) DFTB1 diagonal elements



(c) delta diagonal elements

Figure A.7.: Hyperparameters sampled during the Hyperband optimization algorithm vs the score of resulting model for a) the off-diagonal, b) DFTB1 diagonal and c) LC-DFTB2 diagonal elements.