

Development of Neural Network Models for Prediction of Molecular Properties

John E. Herr

Publication Date

21-04-2020

License

This work is made available under a Public Domain Mark 1.0 (No Copyright) license and should only be used in accordance with that license.

Citation for this work (American Psychological Association 7th edition)

Herr, J. E. (2020). *Development of Neural Network Models for Prediction of Molecular Properties* (Version 1). University of Notre Dame. <https://doi.org/10.7274/j3860577b2v>

This work was downloaded from CurateND, the University of Notre Dame's institutional repository.

For more information about this work, to report or an issue, or to preserve and share your original work, please contact the CurateND team for assistance at curate@nd.edu.

DEVELOPMENT OF NEURAL NETWORK MODELS FOR PREDICTION OF
MOLECULAR PROPERTIES

A Dissertation

Submitted to the Graduate School
of the University of Notre Dame
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

John E. Herr

Olaf Wiest, Co-Director

Paul Helquist, Co-Director

Graduate Program in Chemistry and Biochemistry

Notre Dame, Indiana

April 2020

This document is in the public domain.

DEVELOPMENT OF NEURAL NETWORK MODELS FOR PREDICTION OF MOLECULAR PROPERTIES

Abstract

by

John E. Herr

Machine learning (ML) has seen renewed interest with the advent of modern computational accelerators such as GPUs and the availability of larger scale datasets. ML potential energy models are promising the accuracy of quantum methods at significantly reduced cost. High accuracy potential methods which scale well for long simulations are desirable for many research areas. This dissertation describes methods for collecting datasets of nonequilibrium geometries for training ML potential energy models and shows that enhanced sampling methods or methods which do not follow Boltzmann statistics are necessary for diversely sampling geometries. Several potential models are described. The first is a combination of an ML model with the many-body expansion for condensed phase systems. The result is a highly efficient and accurate method for running large simulations of liquid phase systems. The method is implemented for methanol, but extension to other systems or mixed non-covalent fragments is trivial. The next two chapters describe the development of TensorMol, and improvements made to the model to accommodate more elements. TensorMol is a general NN potential for small organic molecules which includes explicit long-range interactions. The Coulomb and van der Waals energies included with TensorMol are crucial for running accurate simulations with correct long-range behavior. The follow-up work improves upon TensorMol by reframing the parame-

terization such that it is constant with the number of unique elements in the training data. The result is a model which is able to treat a vast expansion of organic molecules by including eleven unique elements in the training data. This dissertation also covers the early stages and future directions of a project using graph neural network models to predict reactions products, yields, and stereoselectivities. Finally, a collaborative project to model the Stokes shift observed in photoluminescence experiments of lead-halide perovskites describes the electronic structure of these materials to elucidate the cause of the Stokes shift. A low-lying gap state is observed which follows the trends of size-dependence in experimental results. Electronic structure calculations verify this low-gap state is responsible for the size-dependence of the Stokes shift.

”...a good education is gold.”

For my grandfather

CONTENTS

Figures	vi
Tables	viii
Chapter 1: Introduction	1
1.1 Introduction	1
1.2 Neural network potential models	2
1.3 Conclusions	3
Chapter 2: The many-body expansion combined with neural networks	4
2.1 Introduction	4
2.1.1 Coulomb matrix representation	4
2.1.2 Depth map representation	6
2.1.3 The many-body expansion	7
2.2 Methods	9
2.3 Results	13
2.3.1 MBE-NN accuracy	13
2.3.2 Inverse design of depth map	22
2.4 Conclusions	23
Chapter 3: Metadynamics for generating off-equilibrium geometries	25
3.1 Introduction	25
3.2 Sampling schemes	26
3.2.1 Molecular dynamics and metadynamics	26
3.2.2 Normal mode sampling	29
3.3 Methods	31
3.3.1 Data collection	31
3.3.2 Model training	33
3.4 Results	34
3.4.1 Determining bias potential parameters	34
3.4.2 Accuracy of models trained across different sampling schemes	37
3.5 Conclusions	43

Chapter 4: TensorMol model with long-range physics	45
4.1 Introduction	45
4.2 Methods	46
4.2.1 HDNNP model	46
4.2.2 Trained models	55
4.3 Results	56
4.3.1 Water model	56
4.3.2 Chemspider model	60
4.4 Conclusions	63
Chapter 5: Fully transferable high-dimensional neural network potentials	65
5.1 Introduction	65
5.2 Methods	67
5.2.1 Deriving the elemental modes	67
5.2.2 Learning formation energies of elpasolites	69
5.2.3 Changes to HDNNP model	71
5.2.4 Adjustments to ACSFs	73
5.2.5 Data generation	75
5.2.6 Simulating alchemical intermediates	79
5.3 Results	80
5.3.1 Elpasolite model	80
5.3.2 HDNNP model	80
5.3.3 Alchemical intermediate simulation	85
5.4 Conclusions	87
Chapter 6: Graph networks for reaction outcome prediction	90
6.1 Introduction	90
6.2 Methods	94
6.2.1 Graph model	94
6.2.2 Predicting bond changes as graph operations	95
6.2.3 Future implementations	98
6.3 Results	98
6.4 Conclusions	100
Chapter 7: Stokes shifts in lead halide perovskites	101
7.1 Introduction	101
7.2 Methods	104
7.2.1 Bulk material characterization	104
7.2.2 Generation of defect models	104
7.2.3 Determination of electronic structure	107
7.3 Results	109
7.4 Conclusions	116

Appendix A: Terminology	117
Bibliography	119

FIGURES

2.1	Model of the many-body expansion and generative adversarial network	12
2.2	Errors associated with each many-body expansion order	14
2.3	Error of a methanol cluster and error per methanol for increasing size clusters.	16
2.4	Comparison of the accuracy of MP2, the many-body expansion, the NN model, and AMOEBA09.	17
2.5	Energy of solvation when breaking the hydrogen bond of a methanol dimer.	20
2.6	Two-body, three-body, and total energy of a methanol trimer bond stretch.	21
3.1	Principal component analysis (PCA) of the distance matrix along a 4 ps trajectory of a water cluster	36
3.2	Variance of the distance matrix components over a 10 ps trajectory for MD and MetaMD.	37
3.3	Errors for models tested on data from the same sampling method as their training data.	38
3.4	Errors for models tested on data from the other sampling methods than their training data.	39
3.5	Potential curve for a nicotine bond stretch from MD model, MetaMD model, and from Q-Chem.	40
3.6	MetaMD method from this work used to find conformers of a water cluster.	42
4.1	Visualization of the TensorMol model.	49
4.2	Bonding curve of a water trimer calculated with DFT and TensorMol.	57
4.3	Rotation breaking the hydrogen bond of a water dimer, and plots of the dipole components.	58
4.4	Simulated IR spectra for water clusters calculated with DFT and TensorMol.	59
4.5	Harmonic and real-time simulated IR spectra with TensorMol for morphine.	61

4.6	Harmonic IR spectrum of four molecules calculated by TensorMol. . .	62
4.7	Non-covalent binding calculated between DNA base pairs with DFT and TensorMol.	63
5.1	Heat maps of encoded physical properties and their encoded representation.	69
5.2	Visualization of the learned encoding of the physical properties. . . .	70
5.3	Formation energy predictions for the test set of elpasolites.	81
5.4	Distribution of embedded atomic energies from models trained with and without chlorine.	83
5.5	Simulation of an alchemical transformation with the improved TensorMol model.	86
7.1	Photoluminescence spectra and observed Stokes shift for CsPbBr ₃ . . .	102
7.2	TEM images of cuboidal CsPbBr ₃ nanocrystals.	105
7.3	Bandstructure, density of states, and model Jablonski diagram for the electronic structure of CsPbBr ₃	109
7.4	Molecular orbitals for four sizes of the CsPbBr ₃ nanocrystal model systems.	110
7.5	Energies of VBES and CBES relative to CHS from the model systems and Stokes shifts from experiment and theory.	112

TABLES

2.1	Mean absolute error (MAE) and mean signed error (MSE) given in kcal/mol of one-body, two-body, and three-body energies.	15
2.2	Relative energies (mE_h) of three most stable geometries of methanol trimers.	18
5.1	Information about the abundance of each atomic species in the training data set for the HD-NNP.	78
6.1	The accuracy of the most probable bond change predictions from Coley et al.[30] and this work.	99
7.1	Calculated crystallographic parameters for bulk CsPbBr ₃	105
7.2	Absorption, emission, and Stokes shift data for CsPbBr ₃ nanocrystal ensembles.	113
7.3	Stokes shift and band gap data for theoretical model systems.	114
A.1	Commonly used abbreviations	117

CHAPTER 1

INTRODUCTION

1.1 Introduction

Machine learning (ML) has experienced a recent renaissance because neural networks (NNs) have been shown to make significant improvements over previous state-of-the-art models. One of the earliest examples that garnered significant attention was the convolutional neural network (CNN) model trained on the Imagenet dataset by Krizhevsky and coworkers,[1] which achieved error rates 10-15% lower than all other entries on two tasks in the Large Scale Visual Recognition Challenge 2012 (ILSVRC2012).[2] Accordingly, NNs have received much attention within the ML community and in a number of other fields, including the physical sciences. NNs are said to be "universal approximators," but it should be noted that the algorithmic learnability of the parameters is not guaranteed.[3] The flexibility afforded by NNs has been shown to make them a powerful way to establish models for a wide variety of tasks, including classification and regression, as well as generative modeling.[1, 4, 5]

The chemical sciences are no exception. There have been many recent examples of ML methods being used to predict molecular[6–11] or materials properties,[12–18] to accelerate quantum methods,[19–24] and to generate force field parameters.[25, 26] Graph networks and sequence transformer models have been used to predict reaction products or to perform retrosynthetic analysis.[27–36] Others have used machine learning together with automated organic reaction robots to search for new reactivity.[37] There is much interest from the pharmaceutical and chemical industries to

explore ML methods for better virtual screening of drug candidates from libraries, or to generate new drug candidates never before synthesized.[38–41] There is also much interest in accelerating sampling methods.[42, 43] Protein folding has also seen some interesting approaches using NN methods.[44, 45]

1.2 Neural network potential models

There has been wide interest in developing potential models using ML methods to fit to quantum data.[5, 46–63] This interest is due to the ability to train models that achieve near quantum accuracy with orders of magnitude less computational effort.[19, 48, 57–59] The implications of having a potential energy model with *ab initio* accuracy that is competitive in computational cost with classical force fields are far-reaching.[46, 57, 64–68] Furthermore, these models employ an ansatz which parameterizes the total energy of a system as a sum of contributions from each individual atom based on a feature vector which represents the local environment of the atom.[47, 69] This makes handling changes in bonding easier, unlike classical force fields where changing any bonds causes issues with conservation of energy.

Because these models are purely data-driven, it is necessary to spend significant efforts toward designing a training and testing dataset. There is much research dedicated to designing and collecting datasets for training.[10, 58, 60, 61, 70–72] Sampling the entire potential energy surface of a molecule is intractable, so often practitioners borrow from sampling methods used for fitting classical force fields,[10, 58, 71] or from enhanced sampling techniques.[70] Other approaches build an initial dataset by using reasonable assumptions of diversity and then employ active learning methods to improve their models by finding samples where the error is anticipated to be large.[60, 73]

While these models have shown great promise to transform many aspects of research, there still remain many issues yet to be solved. Many such works are only

able to treat a few elements at a time due to the scaling of parameters with respect to unique elements.[57, 58, 69, 74, 75] Other issues can arise from the completeness of the feature vector representation.[76] Other works have been able to incorporate known invariances through domain knowledge, but at a cost of increased computational effort.[77, 78]

1.3 Conclusions

NN potential models will be an essential tool for more accurate simulations of moderate and large systems. This dissertation covers four projects aimed at developing potential models with NNs. The first uses a many-body expansion in combination with NNs to develop a potential model for condensed phase systems.[79] The next chapter covers a project which addresses issues with common methods for collecting training data, and introduces an improved method for collecting more diverse datasets.[70] The following two chapters introduce TensorMol, a NN potential model which includes explicit long-range physics, and an improved version of TensorMol which is able to treat eleven unique elements.[57, 59]

Following this, the final two chapters divert from potential models with NNs. The first outlines the early progress of a project implementing graph NNs to predict reaction products and yields. Finally, I present a collaborative project with an experimental group where we investigated the electronic structure of lead-halide perovskite materials to determine the cause of an apparent size-dependent Stokes shift observed during photoluminescence experiments.[80]

CHAPTER 2

THE MANY-BODY EXPANSION COMBINED WITH NEURAL NETWORKS

2.1 Introduction

2.1.1 Coulomb matrix representation

The input features for an ML model must encode the relevant information needed to evaluate the function we wish to approximate using the model. In the case of predicting potential energies of molecules, the model is attempting to approximate solutions to Schrödinger’s equation. As such, it is natural to use features which enter into the electronic Hamiltonian, i.e. nuclear charges and atomic positions.

One way to encode this information is through the Coulomb matrix,[81] which is defined as,

$$M_{ij} = \begin{cases} 0.5Z_i^{2.4} & i = j \\ \frac{Z_i Z_j}{|R_i - R_j|} & i \neq j \end{cases}, \quad (2.1)$$

where i and j correspond to indices of atoms in the molecule, and Z_i and R_i are the nuclear charge and coordinates of atom i , respectively. The diagonal elements of the Coulomb matrix encode the stoichiometry of the molecule, while the off-diagonal elements correspond to the Coulomb repulsion between atoms in the molecule.

The definition of the Coulomb matrix affords translational and rotational invariance, which is crucial to developing ML models which respect conservation of energy[47]. Since there is no well defined ordering of the atom indices, however, the Coulomb matrix is not permutation invariant; that is, permuting the indices of any

two atoms in the molecule leads to two distinct but valid Coulomb matrix representations. Unlike translational and rotational transformations, of which there are infinitely many ways to transform the molecule while remaining invariant with respect to the energy, permutations are defined by the number of atoms in the molecule. Given a molecule with n atoms, then there will be $n!$ unique Coulomb matrix representations of that molecule. One method to treat these permutations is to train the model with all possible permutations of the Coulomb matrix for a molecule if the molecule is small enough for this to be tractable.[8]

One additional problem that must be considered is the dimensionality of the features. The size of the Coulomb matrix depends on the number of atoms in the molecule. In a NN model, information is passed from layer to layer by applying a linear transformation, which is posed as matrix multiplication, followed by applying an element-wise non-linear function. This precludes the possibility of having a different number of features per sample, so the Coulomb matrix must be defined to be constant size. One possibility is to pad the Coulomb matrix with zeros up to the largest molecule in the dataset such that all samples will have the same size representation.[9] In practice this works for molecules with a limited number of atoms, but quickly becomes impractical as the largest molecule in the dataset grows, since the number of Coulomb matrix elements scales quadratically with the number of atoms.

The Coulomb matrix is not limited to small molecules, however. Often times molecular simulations involve, for example, solvent molecules which are trivially divided into discrete subunits of the entire system. When performing ab initio calculations for such an application, due to the cubically scaling cost with respect to the number of basis functions, it can be advantageous to use a many-body expansion.[82] In this work we show that combining such fragment based methods with a NN model provides a way of ensuring the input to the model is constant in size without the need for padding the Coulomb matrix.

2.1.2 Depth map representation

Chemists measure or calculate properties of molecules with the aim of discovering a new molecule that has some set of desired properties, such as a novel drug-candidate with high binding affinities/selectivities for a protein target. Unfortunately, thus far there is no way to perform the inverse of this search, whereby a chemist would define some set of criteria to calculate molecules which fit those criteria. This is referred to as inverse molecular design. While ML models may be a promising way to achieve this goal, one of the basic requirements would be to develop a representation which is directly invertible with the geometry. In this work, I also introduced a novel representation which is referred to as the depth map. The depth map is a depth of field image of a ball-and-stick structure. Depth of field images are commonly used as feature representations for ML models in the computer vision field.[83, 84] Atom sizes are given according to their van der Waals radii, and pixel intensity corresponds to the depth from the plane of the depth map.

Because a molecule can be viewed from any orientation, then the depth map is not a unique representation. When dealing with a constant system, like a single molecule, using a canonicalized orientation can alleviate all or most of this by eliminating degrees of freedom corresponding to translation or rigid rotation. This must be decided on a case-by-case basis, but in general setting an origin and defining a plane parallel to the field of the depth map based on the position of one or several atoms.

The resolution of the depth map can be defined by simply increasing the size of the image and scaling the length per voxel appropriately. Higher resolution affords less noisy inputs to the network, but lower resolutions provide a more compact representation, which keeps computational cost down. Choosing a resolution is a matter of trade-off between these two aspects.

Recently, generative adversarial networks (GANs) were proposed, which are a type of NN that learn to generate realistic samples which fall within a true dataset

distribution.[4] GANs consist of two NNs; a generator and a discriminator. The generator learns to generate realistic samples in the distribution of the real data, and the discriminator learns to predict whether an image comes from the real data or from the generator network. The approach works by first collecting random vectors, z , from a probability distribution $p(z)$. Then a generator network, G , is defined that maps z to the space of the real data by $G(z)$. Also, a discriminator network, D , is defined which maps the data space to a scalar value. The scalar is interpreted as the probability that an image comes from the real data. The discriminator is trained to maximize the probability of assigning the correct label to the real and generated samples, while the generator simultaneously trains to minimize $\log(1 - D(G(z)))$. The effect is that the generator learns to map the probability distribution, $p(z)$, to the distribution of real data. We can then explore the distribution of the real data by exploring the probability distribution $p(z)$.

2.1.3 The many-body expansion

The many-body expansion is an approximation where the total energy of a system is expanded as a sum of terms by

$$E_{total} = \sum_i^{N_i} E_i + \sum_{i < j}^{N_{ij}} \Delta E_{ij} + \sum_{i < j < k}^{N_{ijk}} \Delta E_{ijk} + \dots, \quad (2.2)$$

where E_i is the monomer energy of fragment i in the system, ΔE_{ij} is the two-body interaction energy between monomers i and j , and ΔE_{ijk} is the three-body interaction energy between monomers i , j , and k . The n-body interaction energies are given by taking the total energy of the n-body system and subtracting the lower-order many-body terms. Explicitly, the two-body interaction energies are given by

$$\Delta E_{ij} = E_{ij} - E_i - E_j, \quad (2.3)$$

and the three-body interaction energies are given by,

$$\Delta E_{ijk} = E_{ijk} - \Delta E_{ij} - \Delta E_{ik} - \Delta E_{jk} - E_i - E_j - E_k, \quad (2.4)$$

where E_{ij} is the energy of the dimer of fragments i and j , and E_{ijk} is the energy of the trimer of fragments i , j , and k .

Continuing the expansion of this sum up to the total number of fragments defined in the system leads to an exact calculation of the full system. In practice, Equation 2.2 may be truncated with fewer n-body terms included without significant loss in accuracy, depending on the system at hand. For systems where fragments are defined such that no covalent bonds will be cleaved, it is often enough to include up to three-body terms in Equation 2.2.[85]

When calculating the n-body terms in the many-body expansion, it is important to correct for basis set superposition effects (BSSE).[86] For example, in Equation 2.3, E_{ij} is calculated with the full basis set of both monomers. If the monomers are in close enough proximity, then calculating the dimer energy allows for each monomer to be further stabilized by borrowing the basis functions from the other monomer. This leads to an inconsistency in the treatment of the monomers in isolation as compared to in the dimer system. In practice, the magnitude of this effect depends on the size of the basis set employed. Larger basis sets are more complete, and therefore additional basis functions will result in less stabilization of the monomers in the dimer.

Aside from using a larger basis set, one method for approximating BSSE is to use the counterpoise correction of Boys and Bernardi.[87] The counterpoise correction method approximates the BSSE by calculating lower order many-body terms with the full basis set of the n-body order term being calculated. For example, the BSSE of monomer i in the dimer of monomers i and j is given by

$$E_i^{BSSE} = E_i^{ij} - E_i^i. \quad (2.5)$$

On the right hand side of Equation 2.5, subscripts denote the system being considered and superscripts denote the basis set being used. Because a monomer is able to access the rest of the basis functions in the dimer, then E_i^{ij} must necessarily be lower than E_i^i . This result is subtracted from Equation 2.3 for both monomers, which after cancellation results in,

$$\Delta E_{ij}^{CP} = E_{ij}^{ij} - E_i^{ij} - E_j^{ij}. \quad (2.6)$$

Applying the same method to correct the three-body interaction energies results in,

$$\Delta E_{ijk} = E_{ijk}^{ijk} - \Delta E_{ij}^{ijk} - \Delta E_{ik}^{ijk} - \Delta E_{jk}^{ijk} - E_i^{ijk} - E_j^{ijk} - E_k^{ijk}. \quad (2.7)$$

2.2 Methods

In this project, I chose methanol as the system to showcase the NN model. Previous works have shown that the many-body expansion converges rapidly for water clusters and other molecular clusters with strong van der Waals interactions.[88–90] I fragment the system into individual methanol molecules, and include up to three-body terms in the many-body expansion.

To generate a data set, I first simulated a cluster of 108 methanol molecules at 330 K using the general Amber force field.[91, 92] I also ran an ab initio simulation of three methanol molecules at 500 K. The simulations provide 844,800 samples of single methanol molecules. Previous works of similar systems have suggested using a cutoff distance to limit the number of dimers and trimers required to be included, since the combinatorial nature of including these is the largest computational cost for such a system. I found that a cutoff of 10 Å was sufficient to converge the total energy for a cluster of 108 methanol molecules, so I used this limit for collecting eligible dimers and trimers. This provided 74,240 samples of methanol dimers, and 36,864 samples of methanol trimers. More details for determining an appropriate cutoff distance are

provided in the Supplementary Information.

After collecting the sample geometries, I ran RI-MP2 calculations using the cc-pVTZ basis set for each sample. BSSE was corrected using the k-mer centered basis set approach.[90] All calculations were run using the Q-Chem package.[93] One-body energies, two-body, and three-body interaction energies were calculated according to the scheme presented in Section 2.1.3. These energies, along with the corresponding monomers, dimers, or trimers constitute the training data for the model. Calculating total energies of a system of methanol then follows the typical many-body expansion scheme by adding up the contributions from each n-body system. I note that over the course of the simulations, no bonds were broken or formed, and as such the model trained herein is limited to non-reactive samples. The data set was then randomly split into ratios of 80:20 for training and testing data, respectively.

The model uses a separate fully-connected feed-forward NN for each n-body order of the many-body expansion; one-body, two-body, and three-body energies are predicted by separately parameterized NNs. By summing all the respective n-body interactions in the system, the model should predict the total energy of the system by approximating a many-body expansion on the system. This allows for us to easily provide constant size features to the NNs because each network will only be shown a constant number of atoms, while being able to make predictions for systems of methanol clusters of arbitrary size. This model is trivially extensible to similar systems of clusters of molecules where a many-body expansion can be favorable over a full treatment of the electronic structure. The number of parameters for each NN are different based on the many-body order the NN is being trained on. The one-body NN has one hidden layer with 50,000 hidden neurons in that layer. The two-body NN has two hidden layers with 10,000 and 5,000 hidden neurons respectively, and the three-body NN has three hidden layers with 1,000 hidden neurons in the first layer and 2,000 neurons in the second and third layers. The ReLU activation function was

applied to introduce non-linearities into the model.

Choosing a set of features plays a significant role in the accuracy of the model. Above, I introduced two features I will use in this work. In Section 2.3.1, I first develop the model with the Coulomb matrix. Because the samples will always be stoichiometric, I do not include the diagonal elements of the Coulomb matrix as shown in Equation 2.1, as they will be constant across each sample. Since the Coulomb matrix is also symmetric, then I only include the elements in either the upper or lower triangle of the matrix to avoid unnecessary duplication of the input features. For dimers and trimers, the Coulomb matrix is simply expanded to include the whole dimer or trimer as the input features for their respective NNs.

To account for permutation invariance, I augment the training data by adding all permutations of hydrogen atoms connected to carbon for the monomers, which leaves six possible permutations. For dimers, permutations of the hydrogen on both methyl groups must be included, as well as permuting the methanol molecules themselves, which leads to 72 possible permutations. I include all 72 permutations in the dataset to train for this. Similarly, for trimers it can be shown that there are 1296 possible permutations. It is difficult to include such a large number of permutations in the training data. Instead, I model the three hydrogen atoms on each methyl group as one composite "imaginary" atom attached to carbon as a superposition of the three hydrogen. The model effectively learns the mean interaction of the three constituent hydrogens. This simplifies the number of permutations for trimers, since now only permutation of whole molecules needs to be included, of which there are six possible permutations.

I finish Section 2.3.1 by developing the model with the depth map features, but only did so for the three-body interactions. the reasons for using the depth map were not to compete with the Coulomb matrix in terms of accuracy, but rather to provide an alternative descriptor which is invertible with the geometry. I only set out to show

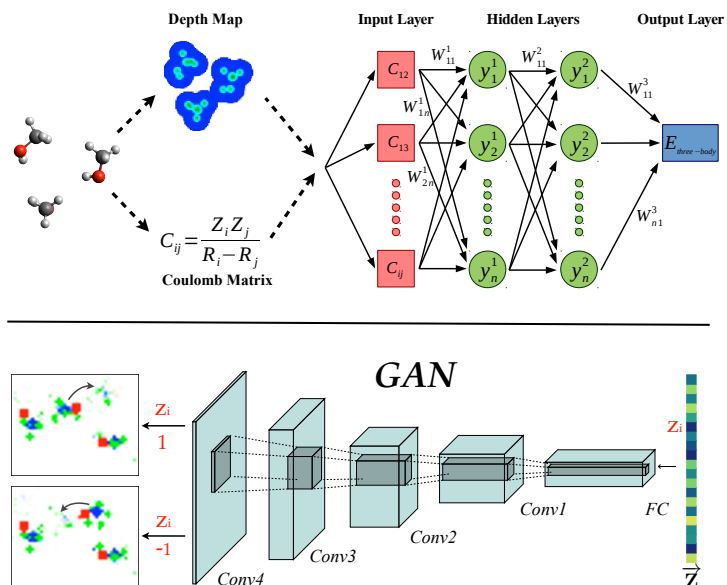


Figure 2.1. Top: The interaction energy of each n-body fragment is predicted by embedding the geometry of the n-body system into the Coulomb matrix or depth map, which is then fed as input to a NN. Bottom: Generative network which takes a vector, z , and outputs depth maps of methanol trimers.

that the depth map can provide suitable accuracy for property prediction, and this is why I elected not to pursue the full MBE-NN model with the depth map. The three-body depth map NN has five convolution layers with 64, 128, 256, 256, and 128 filters in the five layers, and filter sizes of 5×5 for the first layer, 4×4 for the second layer, and 3×3 for the last three layers. Following the convolutional layers there are three feed-forward fully-connected layers each with 1024 hidden neurons. The ReLU activation was used to introduce non-linearities to the model.

To construct a depth map for a sample methanol trimer, first I define the center of the depth map image to be the Cartesian center of all atoms in the trimer, and I rotate the system such that all oxygen atoms are in the plane of the depth map. This removes translational invariance, and limits rotational invariance to one dimension; rotation

about the axis normal to the plane of the depth map. Similarly to permutation invariance, I train the model such that it will learn rotational invariance by training with random rotations about this axis of each sample. The appropriate number of random rotations to add depends on the convergence of the network. In this work, I found that adding 20 rotations of each sample resulted in well-converged energies.

The scheme for the three-body network is shown in the top panel of Figure 2.1. A methanol trimer is first transformed to either a Coulomb matrix or depth map representation. Those features are then fed into the three-body network, which outputs the prediction for the three-body interaction energy of the methanol trimer. Similarly, one-body and two-body networks are set up for the Coulomb matrix representation.

Finally, in Section 2.3.2 I trained a GAN to generate depth maps of methanol trimers. The discriminator was fed images from the generator network and from the set of depth maps of methanol trimers used to train the MBE-NN to discern between images which are real, and those which are generated. The generator network learned to generate depth maps of methanol trimers which it had never seen before. The scheme of the GAN is shown in the bottom panel of Figure 2.1, with examples of depth maps from the generator network shows on the left.

2.3 Results

2.3.1 MBE-NN accuracy

I begin by assessing the errors of the interaction energy predictions. First, I discuss the results for NNs trained using the Coulomb matrix and will return to a comparison with the depth map after. Figure 2.2 shows plots of the interaction energies predicted by the NNs as compared to the energies calculated with MP2 within the many-body expansion (MP2-MBE) for the reserved test data, as well as the distribution of errors for each n-body interaction order. The NN predictions agree well with the calculated

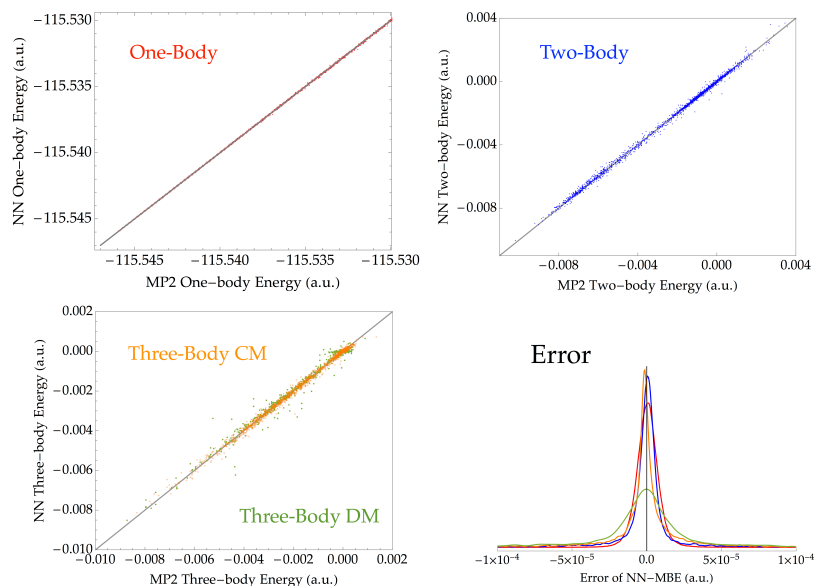


Figure 2.2. The top left, top right, and bottom left panels plot the calculated MP2 energies versus the energy predicted by the networks for the one-body, two-body, and three-body energies from the independent test set. The bottom right panel shows the distribution of errors for each n-body order from the test set.

MP2 energies. Mean absolute errors (MAE) and mean signed errors (MSE) for each network are given in Table 2.3.1. It can be seen that the energy errors are significantly smaller than the expected error of the model chemistry, thus the accuracy is limited by the quality of the underlying calculations used to generate the training data.

It should be noted that the MAE for the two-body and three-body networks is somewhat larger than the one-body network. This is due to these interactions being more complex, and thus harder to model, as well as a smaller amount of training data for these n-body orders. More data could likely improve the errors associated with the two-body and three-body networks, however, generating data for higher order terms is more computationally expensive, and higher order terms in the many-body expansions generally contribute orders of magnitude less energy to the total energy of the system, so the errors associated with their prediction are smaller as well. This

TABLE 2.1

MEAN ABSOLUTE ERROR (MAE) AND MEAN SIGNED ERROR (MSE) GIVEN IN KCAL/MOL OF ONE-BODY, TWO-BODY, AND THREE-BODY ENERGIES.

Error	1-body	2-body	3-body (CM)	3-body depth map
MSE	1.5×10^{-4}	5.6×10^{-4}	-7.3×10^{-4}	-2.5×10^{-3}
MAE	3.8×10^{-3}	9.8×10^{-3}	0.013	0.024

one-body and two-body errors are from the Coulomb matrix model. Three-body errors are given for both the Coulomb matrix (CM) and depth map.

can be seen by considering that the MSEs associated with the one-body, two-body, and three-body terms in Table 2.3.1 are relatively balanced; that is, in each case the associated errors are on the same order of magnitude, despite having a factor of ten more training data for one-body energies than two-body interaction energies, and a factor of two more training data for two-body as compared to three-body interaction energies.

The distribution of energy errors shown in the bottom right panel of Figure 2.2 are uncorrelated. This implies that, as system size is increased, and thus the number of one-body, two-body, and three-body terms contributing to the total energy increases, the average error per contribution will tend towards zero. Thus, when larger and larger system sizes are treated with the model, the error-per-methanol should decrease. To demonstrate this notion, I plot the energy error of the NN model relative to the MP2-MBE treatment for increasing sizes of methanol clusters. I show the error associated with the whole cluster, as well as the error-per-methanol in the system. It can be seen that the error-per-methanol tends towards zero for larger systems, while

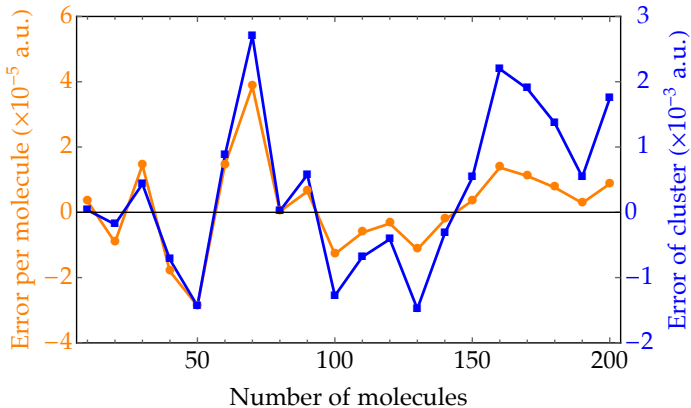


Figure 2.3. Error-per-methanol and total error of the energy for increasing system sizes of methanol clusters. The error is calculated as the difference in energy between the predictions from the NN-MBE and MP2-MBE.

the error of the overall cluster scales independent of system size. Thus, the model will improve in accuracy-per-molecule for larger systems.

The methanol trimer is known to have three minimal energy geometries, referred to as chair, bowl, and chain configurations.[94] In Table 2.2 I show the energies calculated by the model, and with MP2, Hartree-Fock, and B3LYP, for each configuration. Energies are given relative to the chair configuration, which is the lowest energy geometry of the three. the NN-MBE model predicts energy differences between each configuration within 10% of those from MP2. The energy difference errors of Hartree-Fock and B3LYP relative to MP2 are both larger than the model, while each method is significantly more costly to perform these calculations as well.

While the model has been shown to give accuracy close to the underlying ab initio data used to train the parameters while providing orders of magnitude faster results, classical force fields remain more computationally efficient and often give sufficient accuracy for many applications. I show that the model, which approaches the computational cost of a classical force field, retains higher accuracy and is therefore

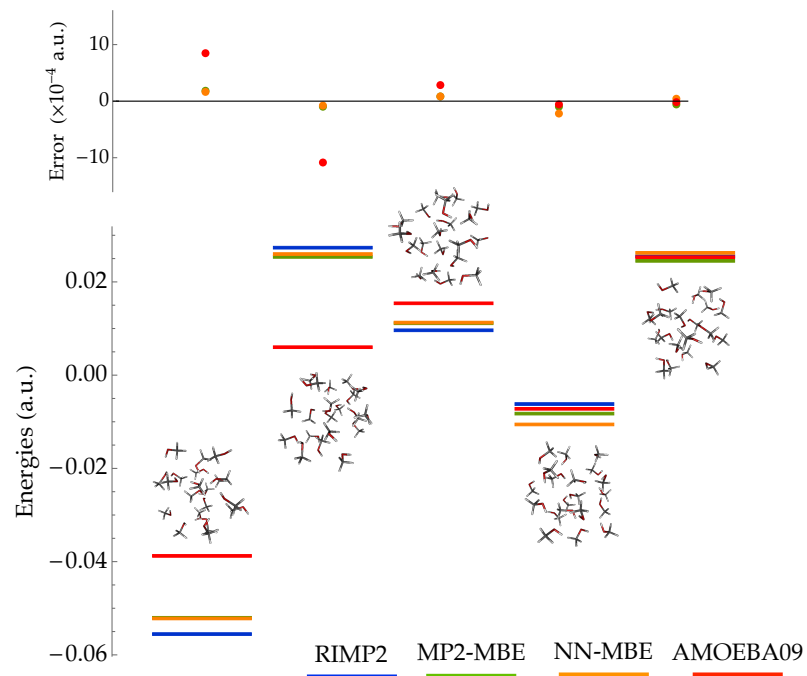


Figure 2.4. Bottom: Relative energies for five clusters of twenty methanol calculated with RIMP2 (blue), MP2-MBE (green), NN-MBE (orange), and AMOEBA09 (red). Top: Error of the energies relative to RIMP2. Shown are the errors for MP2-MBE (green), NN-MBE (orange), and AMOEBA09 (red).

TABLE 2.2

RELATIVE ENERGIES (ME_h) OF THREE MOST STABLE
GEOMETRIES OF METHANOL TRIMERS.

Geometry	RIMP2	MBE-NN	HF	B3LYP
chair	0	0	0	0
bowl	1.50	1.40	1.75	2.02
chain	4.54	4.10	2.56	5.29

MP2, HF and B3LYP energies are extrapolated to a complete basis. The NN-MBE model shows smaller errors relative to MP2 compared to Hartree-Fock and B3LYP. The NN-MBE model shows smaller errors relative to MP2 compared to Hartree-Fock and B3LYP.

useful when the accuracy required to model some chemical phenomena of interest exceeds what force fields are capable of. In Figure 2.4, I have plotted the energies of five random clusters of 20 methanol molecules. I calculated the total energy of each cluster with RIMP2, MP2-MBE, the NN-MBE model, and AMOEBA09. In the bottom panel, energies are given relative to the average of all five clusters. The errors associated with the MP2-MBE results relative to RIMP2 can be considered the error associated with the MBE treatment, and the errors associated with the NN-MBE model compared to MP2-MBE can be considered as the error of the NN. The NN-MBE model matches most closely with MP2-MBE calculations, which is expected because this model chemistry was used to train the NNs. The energies from the NN-MBE model and MP2-MBE deviate slightly compared to a full RIMP2 calculation of the system, while energies from AMOEBA09 deviate even further from RIMP2 results. The top panel shows the energy difference of each cluster for MP2-MBE, the NN-MBE model, and AMOEBA09 relative to the RIMP2 energy of the same

cluster. The energy differences between the model and MP2-MBE overlap such that it is hard to distinguish the green markers for MP2-MBE, while AMOEBA09 shows larger differences. The MAEs of the five clusters for MP2-MBE and the NN-MBE relative to RIMP2 are 0.063 and 0.069 kcal/mol respectively, while AMOEBA09 has a MAE of 0.28 kcal/mol. Thus, the model can provide an alternative which is nearly as efficient as a classical force field and nearly as accurate as ab initio methods.

Treatment of solvent effects is an important part of modeling most chemical processes. I calculated the energy change of breaking a hydrogen bond between a methanol dimer with and without the presence of a solvation shell with MP2-MBE and with the NN-MBE model to see if the NN-MBE model is able to accurately model how a solvation affects such processes. The system is shown in Figure 2.5; the top shows the methanol dimer in a vacuum while the bottom shows the same dimer in the presence of a 10 Å solvation shell. The energy of breaking the hydrogen-bond is calculated by the difference in energy of the system when there is a hydrogen bond between the methanol dimer, and when one methanol is rotated by 180 °to break the hydrogen bond. I note that MP2-MBE shows an energy difference of 7.7 kcal/mol between the system in vacuum and in a solvation shell, while the difference for the NN-MBE is 8.8 kcal/mol, which is 1.1 kcal/mol larger. This suggests that the model shows strong potential to be used to simulate condensed phase phenomena efficiently and accurately.

It is important that the predicted potential energy surface of a system is smooth, if such a model is to be used for simulations. I used the model to calculate energies of a methanol trimer as one methanol gradually moves away from the other two. In Figure 2.6, I plot the total energy, two-body, and three-body interaction energies of the trimer using out model as well as MP2-MBE calculations. The model is shown to exhibit a smooth potential energy surface for the system in good agreement MP2-MBE. The energies show the poorest agreement at the energy minimum (around 2.6

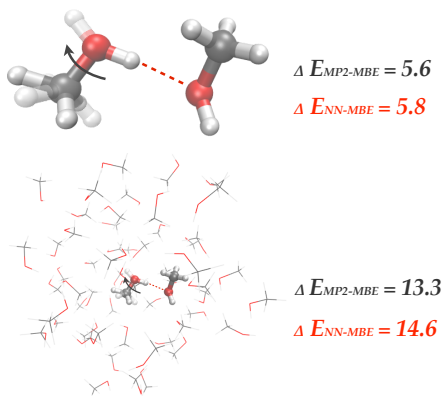


Figure 2.5. Total energy(solid), two-body (dashed), and three-body (dotted) interaction energies of a methanol trimer as one methanol gradually moves away from the other two. Energies are shown for the model (orange) and from MP2-MBE calculations (blue). The model is shown to be smooth and matches well with an MP2-MBE treatment.

Å) and at longer distances (greater than about 4.5 Å), likely due to sampling of the training data with molecular dynamics, which will provide most samples near energetic minima, an issue discussed further in Chapter 3.

Turning our attention to the three-body network trained using the depth map as input, it can be seen that the MSE and MAE are about a factor of four and two, respectively, higher than for the Coulomb matrix in Table 2.3.1. It was not my intention to provide an alternative representation which would be competitive in terms of accuracy with the Coulomb matrix, but instead that it would provide suitable accuracy such that its direct invertibility with the geometry can be leveraged. In this regard, the depth map is successful in that important information about the geometry of the system can be captured in a representation which is invertible.

Analysis of the errors for the depth map and Coulomb matrix in the bottom left panel of Figure 2.2, it can be noted that generally, the depth map performs nearly as well as the Coulomb matrix, but in several cases the energy predicted by the

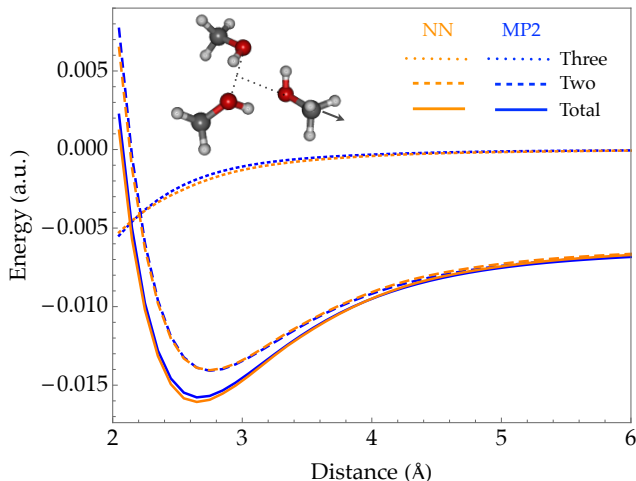


Figure 2.6. Total energy(solid), two-body (dashed), and three-body (dotted) interaction energies of a methanol trimer as one methanol gradually moves away from the other two. Energies are shown for the model (orange) and from MP2-MBE calculations (blue). The model is shown to be smooth and matches well with an MP2-MBE treatment.

depth map network deviates significantly more than the Coulomb matrix relative to the underlying MP2 energies. By examining the input in these cases, it is observed that often these larger errors are associated with a methyl group which eclipses the hydroxyl group from the plane of view based on the canonical orientation described above. This is an obvious pitfall of using such a representation, and in the future this issue would need to be addressed for more complex systems. In the short term, a solution for this specific case could be to provide a second depth map as input from the perspective opposite the first. This would be a trade-off between accuracy and computational cost.

Furthermore, a deviation in the energies predicted by the depth map model occurs near zero where the model tends to predict interaction energies which have a smaller magnitude (i.e. are closer to zero) than the true MP2 energy. This is likely caused by a combination of machine precision and the resolution of the information the network

is able to extract from the depth map. Because the depth map is a voxelized image, the precision with which the distance of two atoms can be determined depends on the resolution of the depth map. When the energies predicted fall within a certain range of zero, the NN is making predictions for which the resolution of the depth map is not sufficient to determine these small differences in energy. The NN is unable to distinguish between these cases near zero energy, and cases where the energy is significantly closer to zero, so it has learned to average the predictions in this region. The resolution of the depth map can be increased to improve this, but this is again a trade-off between accuracy and computational efficiency.

2.3.2 Inverse design of depth map

Next, I discuss the GAN trained using the depth map representation. As I discussed before, the generator network learns to map a vector z drawn from a probability distribution to the space of the training data by attempting to fool an ever-improving discriminator network. After a training period, the competing networks have converged. I can then explore what the generator has learned by initializing vectors from the probability distribution and examining how the generated depth maps change as one component of the vector is continuously changed.

As an example of this, Figure 2.1 provides two example outputs from the generator. The outputs were made by taking one randomly initialized vector and varying one component, z_i , of that vector over the range of the distribution. The two samples are the cases at the extremes, when z_i equals one or negative one. The change of the depth map over this range shows an apparent methanol flip end-over-end relative to its positioning with respect to the other two methanol. This shows that it is possible to model continuous changes to a system as continuous changes to a vector. If the mapping of the properties of the system with the dimensions of the vector used to generate the depth maps can be influenced, then it would be possible to develop a

model where the vector components correspond to desired properties of a system, and a search over the space of the vector could reveal systems with a desired set of properties. Examples of using GANs in this way for image generation include aging of faces in photos,[95] and adding of other features, such as sunglasses.[96]

2.4 Conclusions

Machine learning is gaining traction as a new tool for developing models of potential energy surfaces. Fundamentally, there is a requirement of a constant size input or feature set for many of these machine learning algorithms that must be dealt with for cases of molecules, materials, and systems which come in innumerable shapes and sizes. For systems which can readily be broken down into common fragments, taking advantage of this partitioning of the system lends itself well to machine learning potential energy methods since networks can be defined for each type of fragment, or collection of fragments, to predict additive energies which total the energy of the whole system. Fragmentation methods are well studied, and can be extended to include fragmentation schemes which cleave covalent bonds, but typically require higher order n-body contributions to converge the total energy in such cases. The extension of this model to other systems is trivial in that the NN model is only a replacement for ab initio calculations within a fragmentation method. Thus a NN-MBE model can be developed in any case where a traditional MBE fragmentation method is known to work as well.

The NN-MBE model has been shown to accurately reproduce the potential energy surface of methanol clusters. The error associated with the NN is smaller than the error associated with a MBE method, and thus the model is limited in accuracy by the underlying training data and not the design of the model itself. The potential energy surface for the model is smooth and accurately reproduces the energetic differences between known local minima of methanol trimers. The energies predicted

by the model are shown to be more accurate than classical force fields while orders of magnitude faster than ab initio calculations. The effect of solvation on chemical processes is shown to be reproduced faithfully for condensed phase phenomena. Furthermore, I show that the model becomes more accurate for increasing system sizes. In the future, this model would need to be extended to fragmentation methods of covalent systems and higher order many-body expansion orders. Some of this work may be trivial, but there are likely to be unexpected issues which arise that may require reassessing the input features, or the design of the NN models.

Machine learning has recently seen rejuvenated interest since deep learning has invigorated the field. There is hope that machine learning methods may enable the development of models for inverse design. Early attempts will require system representations which are invertible with the geometry of the system, but suitable for property prediction. I introduced the depth map, which is a field of vision image of molecules. The depth map was shown to provide suitable accuracy for the prediction of methanol trimer interaction energies, such that the possibility was explored of using this representation for inverse design models. I trained a recently introduced GAN model to learn a mapping for a probability distribution to the space of the methanol trimer depth maps. The generator was able to create realistic depth maps which could be continuously controlled by continuously varying the vector determining the output of the generator. GANs are in the early days of development themselves, and future improvements of these models, along with improvements to representation and learning algorithms will all lead to many new questions to explore for molecular inverse design.

CHAPTER 3

METADYNAMICS FOR GENERATING OFF-EQUILIBRIUM GEOMETRIES

3.1 Introduction

Data-driven methods for developing potential energy surface models rely heavily on representative sampling of a reference potential energy surface. Well sampled regions of the potential energy surface will be reproduced accurately by the model, while regions which are sampled more sparsely suffer larger inaccuracies in the final model. Typically, the highest accuracy is desired for low-energy regions of the potential energy surface, but higher-energy regions should be adequately sampled such that simulations faithfully reproduce the desired statistics of the system. A balance must be struck such that data is sampled in relevant regions of the potential energy surface and is distributed relatively evenly throughout the sampled regions, while avoiding unnecessary extremes of the potential energy surface with exceedingly small probabilities of being visited during simulations.

Researchers typically rely on well known sampling methods from statistical mechanics (e.g. NVT molecular dynamics) or other methods, such as normal mode sampling (NMS).[10, 49, 71] There is as of yet little research into the problem of how to best design datasets for these applications. In this work, I compare different methods used by practitioners of molecular simulation for developing such datasets. I also suggest metadynamics[97] as a way to overcome some of the issues with commonly used methods. Enhanced sampling is an active field of research.[98, 99] Indeed, many current state-of-the-art enhanced sampling techniques could be used to construct datasets for ML models, but in this work I focus on comparing metadynamics

to methods which have been commonly used by practitioners developing ML potential models in recent literature.[10, 49–51, 71] I also note a common pitfall that is easy to overlook when making comparisons of the accuracy of models trained on such datasets.

3.2 Sampling schemes

3.2.1 Molecular dynamics and metadynamics

One of the most common methods for sampling geometry configurations of a molecule is standard molecular dynamics (MD).[50, 68, 100, 101] While this method is implemented in nearly every computational chemistry software package, the major problem is that the frequency of sampling high-energy vs. low-energy regions of the potential energy surface is based on the Boltzmann statistics of the system. Low-energy configurations will have exponentially more samples than high-energy configurations. It is possible to increase the temperature of the simulation to more efficiently explore higher-energy conformations, but many organic molecules break apart at relatively modest temperatures, while extreme temperatures are needed to achieve reasonable sampling efficiency.

I propose to use metadynamics[97] as an alternative method for generating sample geometries, because of its simplicity as an extension to MD, and the ability to reweight the distribution of sampling systems in high and low energy regions of the potential surface. Metadynamics was designed to efficiently explore energy surfaces by adding a bias potential based on the previous configurations of the system. Biasing against previous configurations increases sampling of rare events in the dynamics such as traversing a transition state between local minima. If the simulation continues, the potential well around a new local minima can be explored by the simulation. This is not possible with NMS, which is designed to sample the region of the potential

energy surface around the current local minimum of the potential surface. I discuss in Section 3.2.2 that performing NMS on many different molecules likely accounts for this problem when training a NN model on this data, but using metadynamics, it is more certain that multiple local minima of a molecule will be sampled. Furthermore, to develop a model for a system with relatively few unique molecules, multiple local minima must be sampled explicitly for each molecule. Metadynamics is therefore likely a better choice for this purpose.

Metadynamics adds a bias potential based on a collective variable of the system. Typically, this collective variable is hand-chosen, because metadynamics is often used for large systems to encourage the sampling of rare events for these computationally expensive molecules. Designing a collective variable to bias against directly affects how many iterations of the simulation will be needed to sample the events, so in these cases it is imperative to carefully consider the collective variable used. In choosing a collective variable for metadynamics, one must consider that the collective variable distinguishes between the reactant and product state of the system, that the collective variable contains the modes which are too slow for the timescale of the simulation, and that there be relatively few collective variables in all. In this work, I am trying to efficiently sample many regions of the potential energy surface without oversampling those regions. A collective variable which is valid for any system is more appropriate for this purpose because I consider all relatively low energy regions of the potential surface which are feasibly accessible during long simulation times to be of roughly equal importance for sampling.

I propose a collective variable which is based on the components of the distance matrix because it is translation invariant for the system and is well defined for all systems. The distance matrix is defined for all systems in all configurations. It is an internal coordinate representation, which means biasing against the components is

translationally and rotationally invariant. The bias potential is given by

$$V_{\text{bias}}(\vec{R}) = \sum_{\alpha} \lambda e^{-\sum_{ij} (D_{ij}(\vec{R}) - D_{ij}^{\alpha})^2 / (2\sigma^2)} \quad (3.1)$$

where D_{ij} is the distance between the i th and j th atoms, \vec{R} is the configuration of the system at the current time step of the simulation, and α references previous configurations the potential is biasing against. λ and σ are parameters of the bias Gaussian potential, with units of energy and distance respectively. Larger values for λ increase the energy of the bias potential and how strongly the bias force will push the system away from previous configurations. σ controls the width of the bias potential, and larger values mean the system will be subject to the bias for configurations with slightly larger deviations in the distance matrix than the previous configuration.

The distance matrix scales quadratically with system size, but for small organic molecules with a few hundred atoms or less this is a reasonable cost for modern computational hardware and software. There is evidence to suggest that these NN potential models may not display significant increases in error for systems which are larger than the data they were trained on.[58, 102] For larger systems, it would be trivial to use sparse components of the distance matrix, such as only components which begin below some cutoff distance.

One issue with the distance matrix collective variable which is more difficult to overcome is avoiding problems resulting from symmetries in the system. Liquid and gas phase systems may exhibit symmetries by permutation of identical molecules which would not be recognized by the bias potential unless this permutation is explicitly accounted for. This could be dealt with by applying all bias potentials across all identical fragments of a system, but the present work is limited to the case of small organic molecules.

Typically it is preferred to use a high quality DFT functional/ab initio method

and basis set for the training data. Simulating the dynamics of a system with a computationally expensive reference method is slow because the Schrödinger equation must be solved at each time step. To calculate the reference data more efficiently when using MD or metadynamics for sampling, one may use an inexpensive method, such as a classical force field, to run the simulation and then reevaluate each sample with the high-quality reference method. Doing so makes it possible to leverage the hardware of modern computational systems more efficiently for these calculations.

3.2.2 Normal mode sampling

Others have relied on NMS for designing datasets.[10, 58, 71] The process of NMS begins with a molecule at a minimum energy geometry. First, the set of normal modes for the molecule are calculated at some desired level of theory (ab initio, DFT). The corresponding normal mode coordinates are collected into a set $Q = \{q_1, q_2, q_3, \dots, q_N\}$, along with the corresponding force constants, $K = \{k_1, k_2, k_3, \dots, k_N\}$, where N is the number of degrees of freedom for the system. Then a displacement for each normal mode is calculated by,

$$R_i = \pm \sqrt{\frac{3c_i N_a k_b T}{K_i}} \quad (3.2)$$

where k_b is Boltzmann’s constant, N_a is the number of atoms in the molecule, T is temperature, and c_i is from a set of N uniformly distributed pseudo-random numbers such that $\sum_i^N c_i$ is in the range $[0, 1]$. Here, a harmonic potential has been set to the c_i scaled average energy of the system at temperature T . The sign of R_i is randomly determined with equal probability of being positive or negative to ensure that both sides of the harmonic potential are sampled evenly. Then the displacements calculated are used to scale each normal mode coordinate by,

$$q_i^R = R_i q_i \quad (3.3)$$

then the minimum geometry of the molecule is displaced by Q^R , which is the superposition of all q_i^R . This new geometry is then added to the collection of samples to be used in the dataset. The process can be repeated until the desired number of points for the molecule have been sampled.

The advantage of NMS is that samples are generated in a window of the potential energy surface around the minimum geometry and one can be reasonably sure that interactions up to some desired temperature are well captured by those samples. It is easy to balance the number of high-energy and low-energy conformations sampled in this way as well, since samples are not correlated through simulation time. Additionally, after computing the normal modes for the system, collecting geometries is fast and single-point calculations can be run concurrently for each geometry.

Normal mode sampling is not without its own issues, though. First, NMS requires a costly calculation of the full Hessian matrix to determine the normal modes. For small molecules this is not a major concern, but the Hessian matrix scales quadratically with the number of atoms, so this prefactor quickly becomes a significant part of the computational cost for larger molecules. Furthermore, NMS only explores the potential energy surface in a window around the local minima at which the normal modes were calculated. Depending on what is desired of the potential to be created from the data, this may be insufficient when other minima are to be explored if NMS has not been performed from these minima as well.

The authors of the ANI-1 model used NMS for their data generation,[58] and they do make note that this is the case, but they suggest that while some interactions may be under-sampled or missed entirely, that NMS is best used when many different molecules will be included in the training set such that these other interactions will be well represented by samples from other molecules. Their assertion is likely correct, and there is good evidence for this when these models perform well on molecules not included in their training data. In this work, however, I limit the scope to sampling

geometries for a single molecule. It is not uncommon to build models with specific systems in mind, and thus an analysis of how best to collect samples in these cases is desirable. Furthermore, it is difficult to perform a quantitative analysis of the amount to which a molecule benefits from interactions learned by the model from other molecules. A good starting point for such an analysis would be to begin from the case of sampling geometries for single molecules.

3.3 Methods

3.3.1 Data collection

I ran simulations of small systems with different values of λ and σ in Equation 3.1 to determine acceptable values for each parameter. The bias potential should be strong enough to be able to push the configuration of the system out of a potential well without applying forces strong enough to break bonds in the molecule. The width of the bias potential must also be large enough to bias against configurations which are similar without punishing configurations which are significantly different. I ran simulations to observe how different values of λ and σ change measures of the diversity of configurations over simulation time, and made comparisons with standard MD as well.

First, I ran several simulations of a water cluster and observed the principal components of the distance matrix over the course of a short simulation. One MD simulation and two metadynamics simulations were run for 4 ps. Observing the principal components over the simulations compares the diversity explored by the simulations and puts the most diverse axis into the first principal components. Simulations which explore more diverse configurations will exhibit a larger variance in the values of their principal components. Next, I ran four simulations of nicotine with metadynamics and one with MD for 10 ps to observe the variance of the distance

matrix components. Judging how these two properties compare will guide how I use the parameters for sampling configurations in further analysis.

To compare the sampling methods outlined in Sections 3.2.1 and 3.2.2, I sampled 50,000 geometries with each method for a nicotine molecule in vacuum. For MD and metadynamics, I ran simulations of nicotine at a temperature of 600 K. This choice in temperature is commonly used by practitioners generating datasets for building ML models of potential surfaces.[50, 68, 100, 103] The autoignition temperature of organic molecules is typically around 500-600 K. Above the autoignition temperature for a molecule it can spontaneously combust in atmosphere. This makes sampling at temperatures in the range of 500-600 K a reasonable upper-limit temperature when sampling data of small organic molecules.[104] The Andersen thermostat was used because it stochastically rescales atomic velocities to thermostat the system.[105] MD was run using the *ab initio* MD implementation in Q-Chem[93] while metadynamics simulations were run using my own implementation in the TensorMol package.[106] The total simulation time was 25 ps and the equations of motion were integrated every 0.5 fs. Reference configurations were collected every 10 fs of simulation time to store for biasing against future configurations. The geometry at each step was collected for the training data, which gives a total of 50,000 geometries from the simulations. The values for λ and σ chosen in this simulation were 1.0 hartree and 2.0 Å. The reasons for this choice are discussed in Section 3.4.1.

For NMS, I follow the procedure outlined in Section 3.2.2. The value of k_bT in Equation 3.2 was chosen to be 2 mHa, which equates to a temperature of about 630 K. This choice gives some reasonable certainty that the samples from each method are able to access roughly the same regions of the potential surface, but the distribution of those samples within that region of the potential surface will be different based on the sampling method. I performed NMS on the nicotine molecule until 50,000 geometries were collected to match those from MD and metadynamics.

All calculations were performed at the ω B97X-D/6-311G** level of theory with Q-Chem 4.[93, 107] In Section 3.2.1, I mentioned that it is possible to use a computationally less demanding method to run the MD and metadynamics simulations. Because this work was limited to a single molecule, I chose to run the simulations at the level of theory at which the training data is to be calculated. To generate a dataset of many different molecules, the increased computational efficiency would be much more significant. MD simulations were run with the ab initio molecular dynamics (AIMD) implementation in Q-Chem 4. For metadynamics simulations, I used a dynamics implementations within our TensorMol software package with an interface to Q-Chem for energies and forces.[57]

3.3.2 Model training

The goal of this work is to determine the relative merits and pitfalls of different methods for generating samples of a molecule to be used in training ML potential models. One of the easiest ways to do so is to train a model using data from each method, and to compare the accuracy of these models. For this I chose the high-dimensional NN potential model of Behler and Parinello.[5] I use the ANI-1 variant of the atom-centered symmetry functions for our feature representation.[47, 58] Though I are making a direct comparison of identical models trained with different datasets, I expect that the results should be applicable to many types of potential models which are fit using strictly data-driven methods because these models are prone to overfitting.

The networks are trained identically with the exception of the training data used. Only total molecular energies are included in the loss function during training. Including atomic forces in the loss function can improve the accuracy of the model, particularly when training data may be sampled sparsely in the configuration space of a molecule. In this work, the generated data should sample the potential surface

densely enough that including atomic forces in the loss function is unlikely to make a significant difference.

For each sampling method I trained multiple models including progressive amounts of training data. Observing the differences in convergence of the model accuracy on an independent test set with respect to the amount of training data should elucidate the amount of oversampling from each method. Each model has three hidden layers with 500 neurons per layer. The activation function chosen was the exponential linear unit (ELU). The data were split by to an 80:20 ratio for training and testing data respectively. Each model was trained for 2000 epochs in total. For each sampling method, I trained models six models. First, samples from the training data were collected into sets of 2000, 4000, 8000, 16,000 and 32,000 samples which are randomly selected from the full training set. I trained models with each of these subsets, as well as the full training set of 40,000 geometries. Doing this for each sampling method means that in all 18 different models were trained.

3.4 Results

3.4.1 Determining bias potential parameters

First, it is important to determine appropriate values for λ and σ from Equation 3.1. Another factor to consider is how frequently snapshots of the distance matrix should accumulate for reference configurations to bias against. If reference configurations accumulate frequently, the Gaussian bias can be relatively weaker and more narrow because the bias force will accumulate from each reference configuration. This allows for a well distributed sampling of the potential surface, because the simulation will always bias against a recent configuration which is correlated in time. On the other hand, storing many reference configurations will increase memory requirements.

If reference configurations accumulate slowly, then memory usage for the simu-

lation can be kept at a minimum. The caveat is that because fewer configurations will be biased against, then the strength and width of the Gaussian potential must be large enough to push the simulation away from larger regions of the potential surface. This can lead to samples which form more localized clusters throughout the potential surface. In my experience a 10 fs interval for accumulating reference bias configurations with a 0.5 fs time step gives a good compromise between clustering of samples and the memory constraints of modern GPUs. Evaluating the bias potentials on CPUs would allow one to access larger amounts of system RAM at the cost of the computational acceleration GPUs can provide.

First I simulated a cluster of ten water molecules for 4 ps with 0.5 fs time steps. I ran three such simulations, one following AIMD, and two with metadynamics using values for λ and σ of 0.4 hartree and 0.8 Å, respectively in one simulation and 1.0 hartree and 2.0 Å in the other. For each step of the simulation, I performed a principal component analysis on the distance matrix at that step in the simulation. If a simulation explores many diverse configurations, the principal components would be expected to vary greatly over the entire simulation. On the other hand, a simulation that over-samples the same configurations would exhibit much less variance in the principal components. Figure 3.1 shows the first and second principal components over the course of the whole simulation. It is apparent that the AIMD simulation drastically over-samples the same region, while the metadynamics simulations both exhibit more configurational diversity.

Next, I ran five simulations of nicotine in vacuum. The root mean square variance of the distance matrix components up to the current step in the simulation is plotted in Figure 3.2. Again, these simulations were run for 10 ps with a 0.5 fs time step. One simulation was run with AIMD, and the other four each were run with metadynamics. The values for λ and σ in each simulation are given in the legend of Figure 3.2. The variance for the AIMD simulation stays mostly flat over the course of simulation

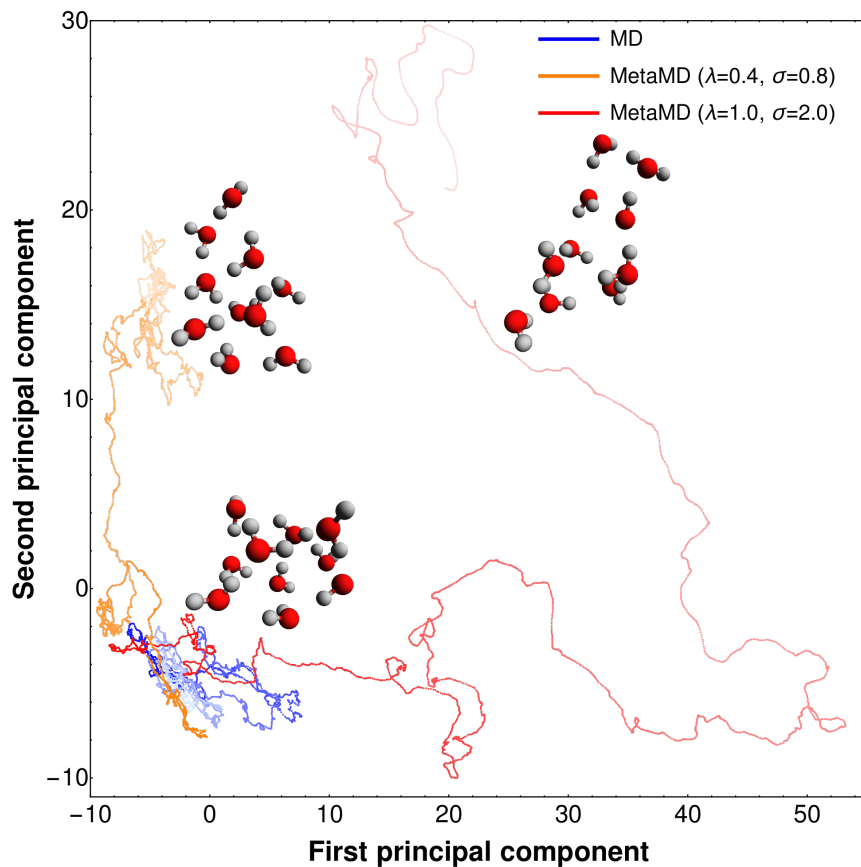


Figure 3.1. Principal component analysis (PCA) of the distance matrix along a 4 ps trajectory with MD (blue) and MetaMD with bump height (width) parameters of 0.4 hartree (0.8 Å) (orange) and 1.0 hartree (2.0 Å) (red) at a temperature of 300 K. All simulations start with a cage-like minimal energy geometry. The color of each line fades as the trajectory time increases. Inserted figures are the final geometries after the 4 ps simulation.

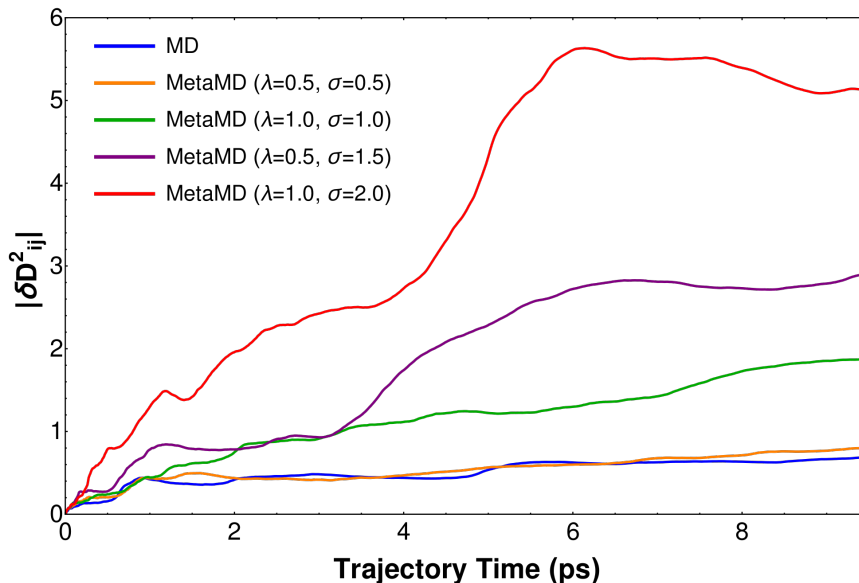


Figure 3.2. Running expectation of the distance matrix variance with ordinary Andersen dynamics and MetaMD $\langle |(\delta D_{ij})^2| \rangle(t)$. Larger λ and σ values provide more bias against previous snapshots of the distance matrix.

time. Similarly, a very weak bias potential (λ and σ values of 0.5 hartree and 0.5 Å) does not significantly increase the variance either. Larger values for λ and σ show a greater increase in the variance of the distance matrix, with the greatest variance occurring when λ and σ are set to 1.0 hartree and 2.0 Å, respectively. Larger values cause covalent bonds to be broken during the 10 ps simulation. Considering the results of these simulations, along with the water cluster simulations discussed in the previous paragraph, I used 1.0 hartree and 2.0 Å for the remainder of this work.

3.4.2 Accuracy of models trained across different sampling schemes

In Section 3.3.2 I described 18 models that were trained; six models trained with increasing amounts of data for each sampling method. For each sampling method there is also a reserve of 10,000 samples which are used as test data. Figure 3.3 shows the mean absolute error (MAE) of each network on test data from the same sampling

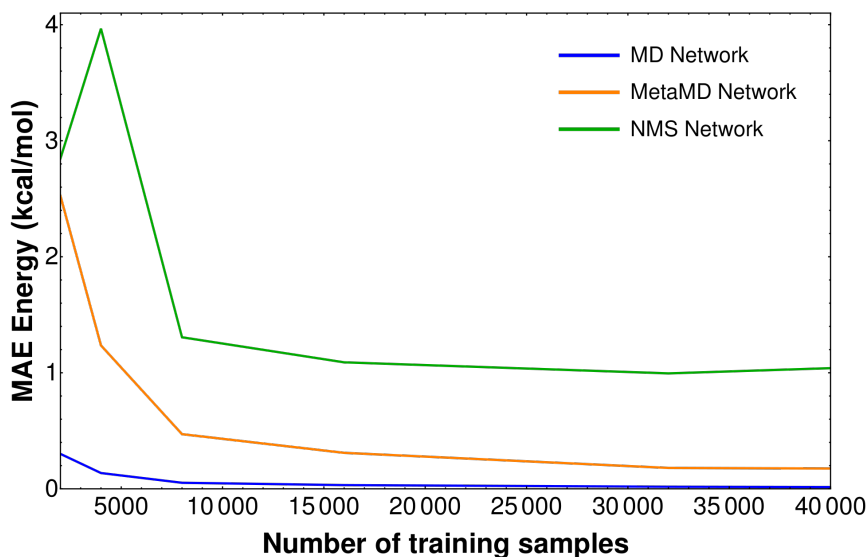


Figure 3.3. MAE of the energy prediction evaluated on the same dataset as the number of training samples increases. AIMD converges with relatively fewer training samples, while MetaMD and NMS both require more samples before converging.

method the network was trained with. With as few as 2000 samples for training data, the accuracy of the MD network is already well under 1 kcal/mol on a test set of 10,000 samples. This means as few as 2000 samples is already enough to cover the span of the entire MD sampling data which aligns with the results in Figures 3.1 and 3.2. For metadynamics and NMS it takes significantly more training data before the model accuracies converge.

To further elucidate this point, I have calculated MAEs for each network on test data from other sampling methods. These results are shown in Figure 3.4. It becomes more apparent now that the MD networks have high accuracy on data from the same sampling method, but the errors are considerably higher on data from methods which explore more diverse configurations. Metadynamics and NMS both have significantly lower errors on data from the other sampling methods.

Finally, I calculated the potential of nicotine with the C-C bond stretch which

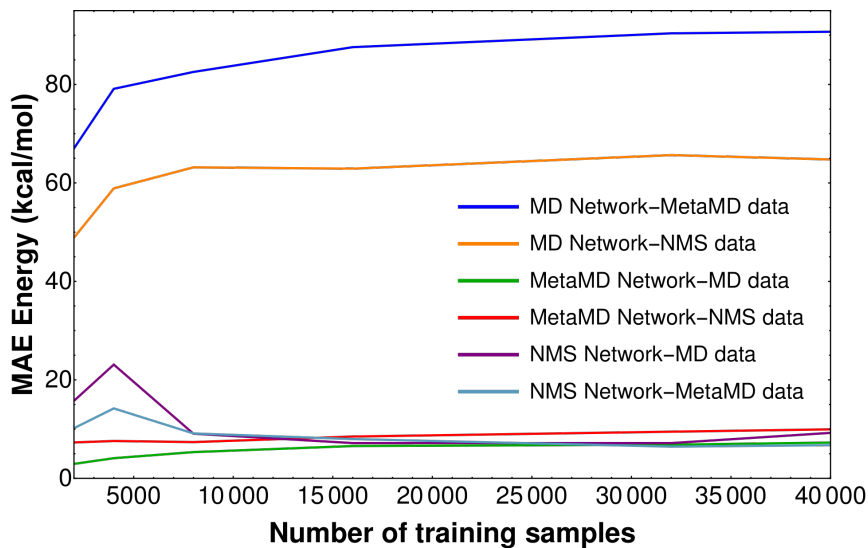


Figure 3.4. Mean absolute errors in the energy prediction cross evaluated on other dataset generation methods as the number of training samples increases. Labels are listed as training data-evaluation data. AIMD networks rarely see samples similar to geometries from MetaMD or NMS, while MetaMD and NMS networks both train on a larger variance of geometry samples, thus generalizing better across other data generation methods.

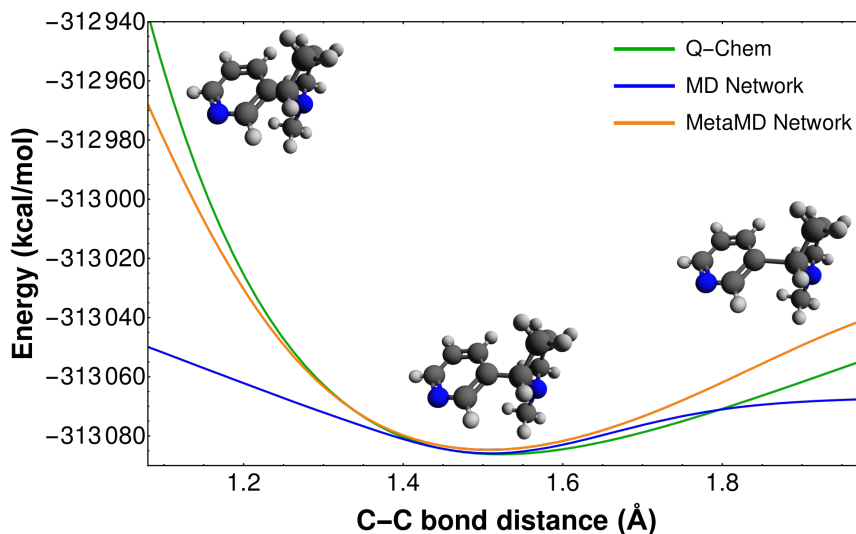


Figure 3.5. Potential energy curve for stretching and shrinking of the C-C bond connecting the rings of nicotine. Potential energies calculated from Q-Chem provided as a reference. The potential from a MetaMD network shows the desired sharp increase of the potential energy as the carbons become increasingly closer. The potential from a AIMD networks shows an increase in the potential energy, but does not rise as sharply as Q-Chem or the MetaMD network.

connects the 1-methylpyrrolidine and pyridine rings from the minimal energy geometry. These energies were calculated using Q-Chem with the ω B97X-D/6-311G** model chemistry, and with the MD and metadynamics networks which were trained on the complete set of 40,000 samples. The data are plotted in Figure 3.5. It is apparent that the MD network is unable to accurately predict the total energy beyond a small well near the local minimum. The network trained with metadynamics data has significantly better accuracy at regions much further from the local minimum.

A key result of this paper is that practitioners looking to develop ML potential models should use caution when collecting sample data. MD is insufficient, even at high temperatures. To overcome the unfavorable probabilities of sampling high energy configurations, more advanced sampling methods must be used. Using a test

dataset that is collected in the same way as the training data can make models appear more accurate than they might be. If the model is to be used for simulations, then it must be stable over high energy regions of the potential surface, as well as multiple local minima.

The best way to collect training data likely comes from a combination of methods. NMS gives a reasonable assurance of evenly sampling the potential window around the local minima, but suffers from the inability to sample different conformers. NMS also tends to exhibit some configurations which are highly unstable and do not provide much useful information to the model. Some examples of this can be seen in the Supplementary Information. Metadynamics is able to traverse transition states and provides significantly improved sampling over MD, but does not give the same distribution of sampling around local minima.

Ideally, it is desirable to sample the potential windows around the local minima of all conformers of the molecule, and the relevant transition states between them. Of course, there exist many symmetries of the high-dimensional potential surface which means that many regions of the potential surface may be well inferred from other regions. Thus it may be enough to sample several local minima and the transition states between them.

Metadynamics is capable of sampling many diverse configurations quickly, and is designed to traverse transition states to sample multiple conformers of a molecule. The water hexamer exhibits several low-energy conformers of interest. I started from the "bag" conformer, and ran a metadynamics simulation using a water ML potential model which I trained in a previous work.[57] Every 100th time step, I took the current configuration and performed a geometry optimization. Within the first 5 ps of the simulation, the resulting minimized geometries had resulted in four other well known low-energy conformers of the water hexamer; the prism, ring, boat, and cage. At around 16 ps of simulation time the book conformer was also found by

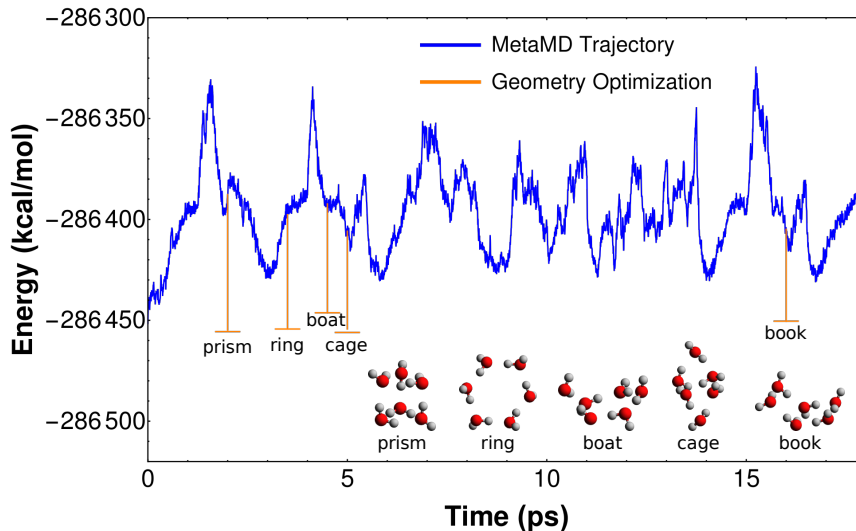


Figure 3.6. MetaMD trajectory starting from the bag geometry of a water hexamer with geometry optimization every 100th time step. MetaMD parameters were 1.0 hartrees and 2.0 Å for the bump height and width respectively, with a time step of 0.5 fs and bumps added every 5.0 fs. Trajectory time was 20 ps. The blue curve follows the energy of the MetaMD trajectory, orange curves follow the energy of a geometry optimization starting from the respective time step of the MetaMD trajectory. Geometry optimizations are only shown when for the first appearances of other low-energy water hexamer geometries. Inserted figures are the results from the converged geometry optimizations.

another geometry optimization. The potential over the simulation, along with the point at which each conformer was first found from the geometry optimization is shown in Figure 3.6.

It is simple to collect the normal modes for each conformer to perform NMS and sample the potential well around these minima. This would assure the potential well around each conformer is well sampled, as well as transitions states between them. The data from both methods can be combined into a single training set for a potential model. Ultimately a practitioner can be reasonably certain that they have sampled sufficiently diverse data to learn the breadth of the surface which could be visited

during long simulation times. This is an extremely important consideration for ML potential models. Their place in the hierarchy of potential energy methods relies on their accuracy exceeding what classical force fields can achieve, their computational cost scaling better than ab initio and DFT methods, and their ansatz which is based on local atomic environments instead of explicit bonding parameters. To develop a successful model, it is imperative to ensure that the accuracy remains high over the relevant parts of the potential surface.

3.5 Conclusions

With a wider adoption of ML potential models, it is good to take caution with these highly data-driven methods. The practice of developing a dataset is not as simple as collecting enough geometries to achieve a large amount of training data. A smaller diverse sampling of the potential energy surface is superior to massive amounts of samples. The data from straight-forward MD sampling is lacking in diversity, even at relatively high temperatures for the molecule.

The biases of the potential surface are not favorable for sampling a training set. While the highest accuracy is typically desired in the most over-sampled regions of the potential surface, there must be some balancing to the sampling of configurations which have exponentially less probability of appearing, but still need to be inferred accurately by the ML potential model. The methods presented here can be combined with active learning techniques, where a practitioner would develop multiple models trained on overlapping subsets of the full training data, and then would run a simulation of the system for which it was developed.[60] If the agreement between the set of models goes beyond a threshold, then the assumption is that the region of the potential surface is under-sampled so the configuration is added to the training data and each model is retrained. To avoid frequently retraining each model during this active learning phase, it is desirable to begin with a dataset which already covers

most of the potential surface.

The lack of interpretability of many data-driven models should be a caution to rigorously testing and developing these models. Aside from the ability to encode patterns into the structure of the model, then the accuracy relies on the underlying training data. The sampling and curation of the dataset should of considerable importance to the entire process.

CHAPTER 4

TENSORMOL MODEL WITH LONG-RANGE PHYSICS

4.1 Introduction

Data-driven models of potential energy surfaces have received a growing interest in chemical and material sciences recently. In particular, for organic molecules the high-dimensional neural network potential (HDNNP) model of Behler and Parrinello has achieved considerable success and is the standard for NN potential models.[5, 46, 47] The robust ANI-1 models are based on the same extensive scheme but use a modified version of the atom centered symmetry functions (ACSFs) to incorporate an encoding of the atomic number of atoms in the local environment of one another.[58, 60, 61]

These models have been successful because of the approximation of the total energy as a summation of contributions from each atom in the system based on a description of the local environment of the atom. An extensive scheme is necessary to accommodate arbitrary sizes of molecules by the model, and limiting the sensory range of the feature representation keeps the computational cost competitive with classical force fields. The assumption that total energies are only based on local atomic environments, however, means that the model is unable to learn well known long-range physical interactions commonly employed in molecular mechanics and density functional theory (DFT).

Electrostatic interactions are some of the longest-range interactions that must be considered in molecular simulation. They fall off as $1/r$ where r is the distance

separating two charged particles. Practitioners of classical force field simulations employ either fixed charge schemes or depend on more costly approaches to explicitly include polarizability.[108] Because of the slow decay of Coulombic interactions, then typical feature representations do not contain the sensory range to incorporate them into their prediction of atomic energies. This is a failure of these models to faithfully reproduce the potential surface.

Incorporating known physical interactions constrains the learning problem in training an ML potential model. The training procedure attempts to learn patterns which correspond with changes in the energy of a system. By incorporating known physical interactions with a small computational cost, the model is able to ignore signals which inform these physical interactions to focus on learning the interactions which are more expensive to calculate.

In this work, TensorMol is introduced, a NN potential model which is augmented with long-range physics. TensorMol features a linear-scaling inductive dynamic charge model and van der Waals interactions along with an HDNNP model. The charge model is used to incorporate long-range Coulomb interactions and enables the calculation of infrared (IR) spectra with TensorMol. The added physical interactions provide necessary long-range interactions for running accurate simulations and keep the computational cost of the model reasonable.

4.2 Methods

4.2.1 HDNNP model

The model is represented in Figure 4.1. The atoms in a molecule are projected onto the ACSF basis to describe their local environment. The ACSFs only encode geometric information, so the features are split into channels based on the atomic number of the atom in the local environment. First there is a charge network which

predicts the partial charge for each atom in the molecule. The partial charges of all atoms are summed to calculate the total charge and the error in total charge of the molecule is subtracted evenly from all atoms to neutralize the charge. The molecular dipole is also calculated by assuming point charges at each atomic coordinate. The partial charges are also used to calculate a Coulomb energy for each atom in the molecule. The energy network predicts an embedded atomic energy for each atom in the molecule. In previous works these embedded atomic energies were summed to give the total energy of the molecule. In this work the Coulomb energy is also added to these predictions.

Each atom in a molecule or system encodes its local environment through the ACSFs, which are a basis of two-body and three-body correlation functions.[47] The developers of the ANI-1 model used a slightly modified version which was used in this work.[71] The ACSFs contain two components; a radial component given by,

$$G_m^R = \sum_{j \neq i} \exp \left[-\eta (R_{ij} - R_s)^2 \right] f_c(R_{ij}), \quad (4.1)$$

and an angular component given by,

$$G_m^A = 2^{1-\zeta} \sum_{j,k \neq i} (1 + \cos(\theta_{ijk} - \theta_s))^\zeta \times \exp \left[-\eta \left(\frac{R_{ij} + R_{ik}}{2} - R_s \right)^2 \right] f_c(R_{ij}) f_c(R_{ik}), \quad (4.2)$$

where i , j , and k are indices over atoms in the system, R_{ij} is the distance between atoms i and j , and θ_{ijk} is the angle between \vec{R}_{ij} and \vec{R}_{ik} . Summing over j in Equation 4.1 and over pairs of j and k in Equation 4.2 ensures that the ACSFs are invariant

to permutation of atom indexing. The distance cutoff function, $f_c(R_{ij})$ is given by,

$$f_c(R_{ij}) = \begin{cases} 0.5 \times \cos\left(\frac{\pi R_{ij}}{R_c}\right) + 0.5 & R_{ij} \leq R_c \\ 0 & R_{ij} > R_c \end{cases}, \quad (4.3)$$

where R_c is the cutoff distance. This function scales from one at $R_{ij} = 0$ to zero at $R_{ij} = R_c$. Scaling the ACSFs this way gives higher value to signals from atoms which are closer in the local environment and which should have a greater effect on the predictions by the NN. Furthermore, it smoothly scales signals to zero at the limit of the sensory range which prevents an abrupt change in the environment so that predictions by the NN will vary smoothly as one atom moves beyond the sensory range. A similar cutoff is used in the Tersoff potential.[109]

In Equation 4.1, m indexes over a set of η and R_s parameters. R_s shifts the center of the Gaussian peak and η controls the width. In the original work from Behler and Parrinello multiple η were used with a few R_s to include multiple scales of probing the radial environment from each Gaussian center. In the ANI-1 model and in this work, one value for η is used but more values for R_s . This makes each radial function probe narrow regions of the radial environment. I used 4.0 for η and a total of 32 R_s evenly spaced to a distance of 4.45 Å for each Gaussian center and a cutoff distance of 4.6 Å.

In Equation 4.2, m instead indexes over a set of four parameters: η , ζ , R_s , and θ_s . Here η and R_s work similarly to Equation 4.1 but they probe the average distance of a pair of atoms in the local environment. I use the same value for η but only use 8 R_s spaced evenly to 2.7 Å with a cutoff distance of 3.1 Å. θ_s probes the angular environment analogously to the way R_s probes the radial environment, with ζ controlling the width of peaks in the angular environment. I used 8.0 for ζ and 8 θ_s evenly spaced over 2π radians.

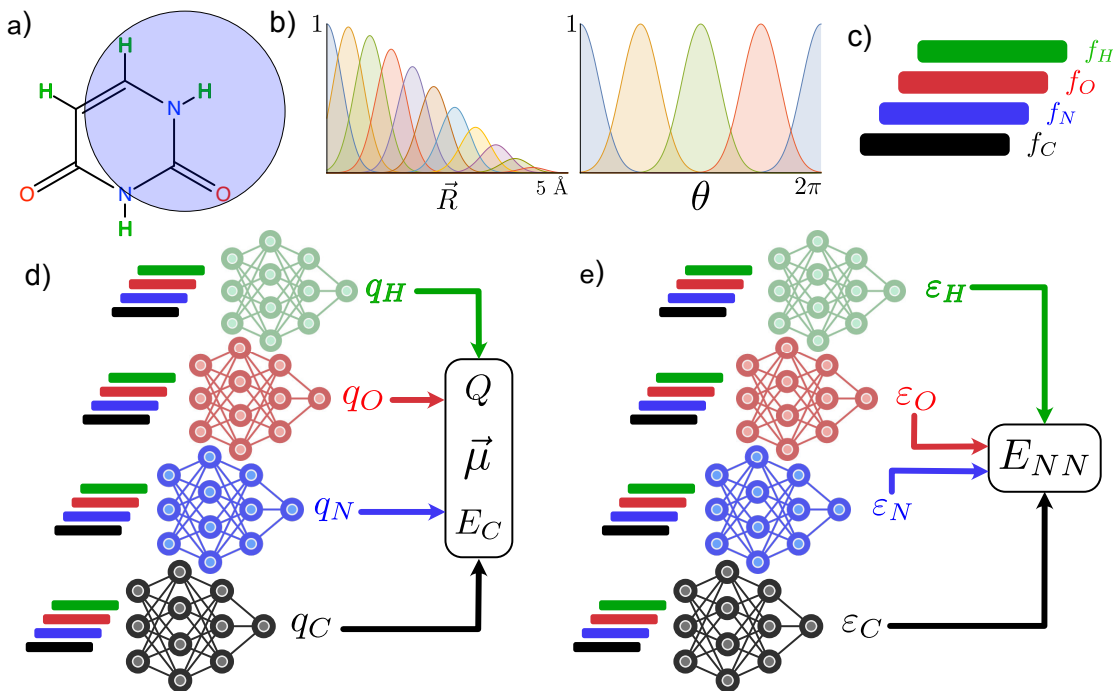


Figure 4.1. a) Atoms are color-coded by element which determines which networks they use and which channel their features go in. b) The symmetry functions are a set of two-body and three-body correlation functions which describe the local environment of an atom. c) The symmetry functions are split into channels to encode the atomic number of the atoms in the local environment. d) The charge network is trained first. Each atom is assigned a partial charge based on its feature vector. The charges are neutralized for charge equality, and the molecular dipole is calculated by assuming point charges on each atom. The loss function is the mean square error of the molecular dipole. When charge training has converged, the charges are used with a Coulomb kernel to calculate a Coulomb energy as part of the process of training the energy network. e) The energy network works the same as the original model by adding atomic contributions to the total energy. This energy, E_{NN} , only accounts for short range interactions. E_C and the van der Waals energy are added to E_{NN} to calculate the total energy.

The ACSFs do not encode the atomic number of atoms in the local environment. To remedy this, the ACSFs are split into channels similar to red-green-blue channels in a color image. The radial functions are split into N channels where N is the number of unique elements in the training data. The angular functions must encode the identity of both atoms being probed in the local environment. Furthermore, the way the pair of atomic numbers is encoded must be invariant to permutation of the two neighbor atoms. The angular functions are thus split into channels for each unique unordered set of atomic number pairs. The number of channels for this encoding will be $N(N + 1)/2$. The ACSFs are split into these channels based on atomic number before summing over j in Equation 4.1 and over j and k in Equation 4.2. The radial and angular features are then concatenated into one vector to make the feature vector for an atoms local environment.

The model in this work consists of two sub-networks; a charge network and energy network. Analogously to previous work each sub-network contains a NN for each unique element in the training data. The feature vectors are fed as input to the network corresponding to the atomic number of the atom they are encoding the local environment of. This gives the model the ability to parameterize its predictions separately for each element.

The charge network uses the feature vector of an atom to predict the partial charge based on its local environment. The learning target for the charge network is the molecular dipole. By assuming point charges at each atomic positions, the dipole is calculated by,

$$\vec{\mu}^{\text{NN}} = \sum_i q_i \vec{r}_i \quad (4.4)$$

where q_i is the partial charge predicted for atom i and \vec{r}_i is the vector to the position

of atom i . The loss function for the charge network is given by,

$$L_{dipole} = \sum_A \left(\frac{\vec{\mu}_A^{\text{DFT}} - \vec{\mu}_A^{\text{NN}}}{N_{\text{atom}}} \right)^2 \quad (4.5)$$

where $\vec{\mu}^{\text{DFT}}$ is the molecular dipole calculated at a reference model chemistry for the training data, and A indexes over a mini-batch of training data during the training procedure. The molecular dipole is used as the learning target because it is a quantum mechanical observable. Because the charge network is trained with a loss function on the dipole vector, then it works similarly to the original model in that the direct output of each network is only constrained such that the eventual learning target should be correct. Given highly correlated configurations, however, the model will learn that smooth changes in the geometry will lead to smooth changes in the charges.

The partial charge predictions are then constrained such that they should reproduce the molecular dipole. There is no specific constraint on the total molecular charge of the molecule, however, in practice the errors on the total charge are typically small. I believe this is due to a combination of the loss function on the molecular dipole and that every sample in our training data is a charge neutral system. To ensure charge neutrality, the error in the total charge is subtracted evenly from the atoms in the system. Observations reveal that this has a negligible affect on the predicted charges. It is also possible to add a loss function on the total charge of the molecule to ensure that the error is small.[68]

The partial charges are also used to calculate a Coulomb energy for the system. Because the Coulomb energy decays as r^{-1} , then the interactions decay very slowly with distance. It is possible to include all long-range interactions for a system through Ewald-type electrostatics when using periodic boundary conditions.[110, 111] While these methods introduce fewer artifacts in the simulation than electrostatic cutoff schemes, their computational cost can be high.

There is still work into designing pairwise r^{-1} summation methods which offer greater stability over longer simulation times. These cutoff schemes are more computationally efficient than lattice summation electrostatics and scale linearly, but the cutoff neglects some long-range interactions from the system. These small errors accumulate over the course of the simulation to give larger artifacts in the simulation. Fennel and Gezelter introduced the damped-shifted force (DSF) method which gives results that compare well to interactions calculated from the smooth particle-mesh Ewald method.[112] The DSF potential is given by,

$$V^{DSF}(r) = q_i q_j \left[\frac{\text{erfc}(ar)}{r} - \frac{\text{erfc}(aR_c)}{R_c} + \left(\frac{\text{erfc}(aR_c)}{R_c^2} + \frac{2a}{\pi^{1/2}} \frac{\exp(-a^2 R_c^2)}{R_c} \right) (r - R_c) \right], \text{ for } r \leq R_c, \quad (4.6)$$

where R_c is the cutoff distance beyond which no electrostatic interactions are included, and a is a parameter which controls the damping of the Coulomb pair potential in Equation 4.6. Damping is used to improve the convergence of the Madelung energy calculated with the Coulomb pair potential.[113]

At distances approaching zero the DSF exhibits a singularity which causes instability when evaluating the gradients of the NNs for training. For this reason, the DSF kernel used in this work is modified by the ELU function such that the Coulomb potential converges to a constant and the Coulomb forces are zero at short-range. The Coulomb energy in our model is calculated using a modified DSF kernel given by,

$$V^{Coul.}(r, q_i, q_j) = \begin{cases} q_i q_j (\alpha \times \exp[r - R_{SR}] + \beta) & r < R_{SR} \\ V^{DSF}(r, q_i, q_j) & r > R_{SR} \end{cases}, \quad (4.7)$$

where R_{SR} is a distance at which the DSF kernel switches from the the ELU function to the original DSF at long-range. In this region the energy network is able to learn

the rest of the contribution to the energy which will be shown in Section 4.3.1.

The energy network predicts an embedded atomic energy for each atom in the system, ε_i . The energies are summed to give,

$$V^{NN} = \sum_i \varepsilon_i. \quad (4.8)$$

Typically, V^{NN} would be taken as the total energy of the system. In our model the total potential energy is instead given by,

$$V^{total}(\vec{R}) = V^{NN}(\vec{R}) + \sum_{i \neq j} V^{Coul.}(r_{ij}, q_i, q_j) + \sum_{i \neq j} V^{vdW}(r_{ij}) \quad (4.9)$$

where $V^{Coul.}$ and V^{NN} are given by Equations 4.7 and 4.8 respectively, and V^{vdW} is given by

$$V^{vdW}(r_{ij}) = -s_6 \frac{C_6^{ij}}{r^6} f_{damp}(r) \quad (4.10)$$

which is the van der Waals energy following Grimme's C6 scheme.[114] This is a dispersion correction scheme where s_6 is a global scaling factor. C_6^{ij} is the dispersion coefficient for the atom pair i and j which is calculated by

$$C_6^{ij} = \sqrt{C_6^i C_6^j} \quad (4.11)$$

where C_6^i and C_6^j semiempirical parameters determined for the scheme. $f_{damp}(r)$ is a damping function used to avoid singularities which is given by

$$f_{damp}(r) = \frac{1}{1 + e^{-d(r/R_r - 1)}} \quad (4.12)$$

where R_r is the sum of the van der Waals radii of the atoms, and d is a parameter set to 20.

The atomic forces can be obtained by differentiating backwards through the

NNs. TensorMol is implemented in TensorFlow,[115] including the calculation of the ACSFs. Because TensorFlow is a python package which includes reverse accumulation automatic differentiation, calculating the atomic forces is implemented with a single line of code. The gradient can be propagated back to the input coordinates without the need to implement any of these gradients by hand.

The loss function for the energy network is given by

$$L_{\text{energy}} = \sum_A \left(\frac{V_A^{DFT} - V_A^{total}}{N_{atom}} \right)^2 + \gamma \sum_A \left(\frac{F_A^{DFT} - F_A^{NN}}{N_{atom}} \right)^2, \quad (4.13)$$

where V^{DFT} and F^{DFT} are calculated at a reference model chemistry for the training data, and A again indexes over a mini-batch of training data during the training procedure. F^{NN} are the forces obtained by differentiating the total energy of the network with respect to the atomic coordinates. Including the forces in the loss functions trains the model to learn the correct shape of the potential surface. Energies provide information about the potential at a certain point, but the gradients contain information about the local shape of the potential. γ is a scalar to control the weight of the loss on the atomic forces compared to the total energy. Each sample provides $3N$ forces but only one total energy for an N atom system. Setting γ to a value between zero and one allows us to weight the loss of the total energy higher to counteract the large amount of force loss components.

In training the TensorMol model, the charge network is trained first while the energy network weights are frozen. After the charge network has converged, the energy network is trained while the charge network weights are frozen. The reason for this is because initially the predicted charges are very poor until some training of the weight parameters has been performed. This causes the Coulomb energy to be unstable during these early iterations. The energy network tries to learn a short-range component of the total energy, but uses a loss function which includes the total

energy. Because of this the training process is more difficult if the Coulomb energy is changing from iteration to iteration as the energy network will be trying to optimize predictions to a moving target. Training the charge network first makes training the energy network easier.

4.2.2 Trained models

Two versions of the TensorMol model were trained. The first version is trained on clusters of water molecules. Each sample contains between one to 21 waters. The second network is trained on a set of 15,000 unique organic molecules containing only C, N, O, and H atoms with the largest molecule containing 35 atoms. These organic molecules were sampled randomly from the ChempSpider database.[116] These are referred to as the water model and ChempSpider model respectively.

Samples for each network were collected following a metadynamics procedure as outlined in Chapter 3 and in a previous publication.[70] For the water model, 370,000 geometries were collected in all. For the ChempSpider model, 3,000,000 geometries were collected from the 15,000 molecules. Energies, atomic forces, and molecular dipoles were collected from calculations with an ω B97X-D/6-311G** model chemistry using the Q-Chem software package.[93, 107] The data were split 80:20 for a training and testing set in both models. Because this is the model chemistry used to calculate our training data, comparisons in Section 4.3 will be made relative to the same model chemistry. For brevity, I will refer to this model chemistry as DFT in the rest of the analysis for this work.

Each model contains three hidden layers for each element in both the charge and energy networks. The water model has 500 neurons per hidden layer and the ChempSpider model has 2000 neurons per hidden layer. The softplus activation was used at each hidden layer in the form $\ln(1.0 + \exp\{\alpha x\})/\alpha$ with α set to 100.

4.3 Results

4.3.1 Water model

After training the water model, the root mean square error (RMSE) over the test set was $0.054 \text{ kcal mol}^{-1} \text{ atom}^{-1}$ and $0.49 \text{ kcal mol}^{-1} \text{ \AA}^{-1}$ for the energies and forces respectively. The top left panel of Figure 4.2 shows a binding energy curve for a water molecule trimer as one water is moved away from the other two. The network reproduces the binding energy curve accurately and changes smoothly throughout the stretch.

The bottom panel of Figure 4.2 shows the same binding energy curve, but with the contributions from the energy network, the calculated Coulomb energy, and the van der Waals energy in Equation 4.9 broken down as a percentage of the binding energy. At a short range the total energy is dominated by E^{NN} . The van der Waals contribution drops off quickly, and the energy network gradually decreases before dropping off smoothly as the Coulomb energy begins to dominate at long-range. Once all atoms in the translating water have moved beyond a certain point, they have moved past the sensory range of the ACSFs and so the contribution from E^{NN} drops to zero.

The charge network learns high quality charges able to accurately reproduce the dipole moment. A water dimer system with a hydrogen bond between the two water molecules is used to display this by rotating the proton-donating water about the axis between the oxygen and the hydrogen opposite the one donated to the hydrogen bond. The energy is shown throughout this rotation as well as the three components of the dipole vector from TensorMol and as calculated the by the same model chemistry used for the training data in Figure 4.3. The dipole components and energy all match the DFT results well and the dipole vector changes smoothly.

Given the higher dimension of the Hessian matrix, its accurate reproduction is

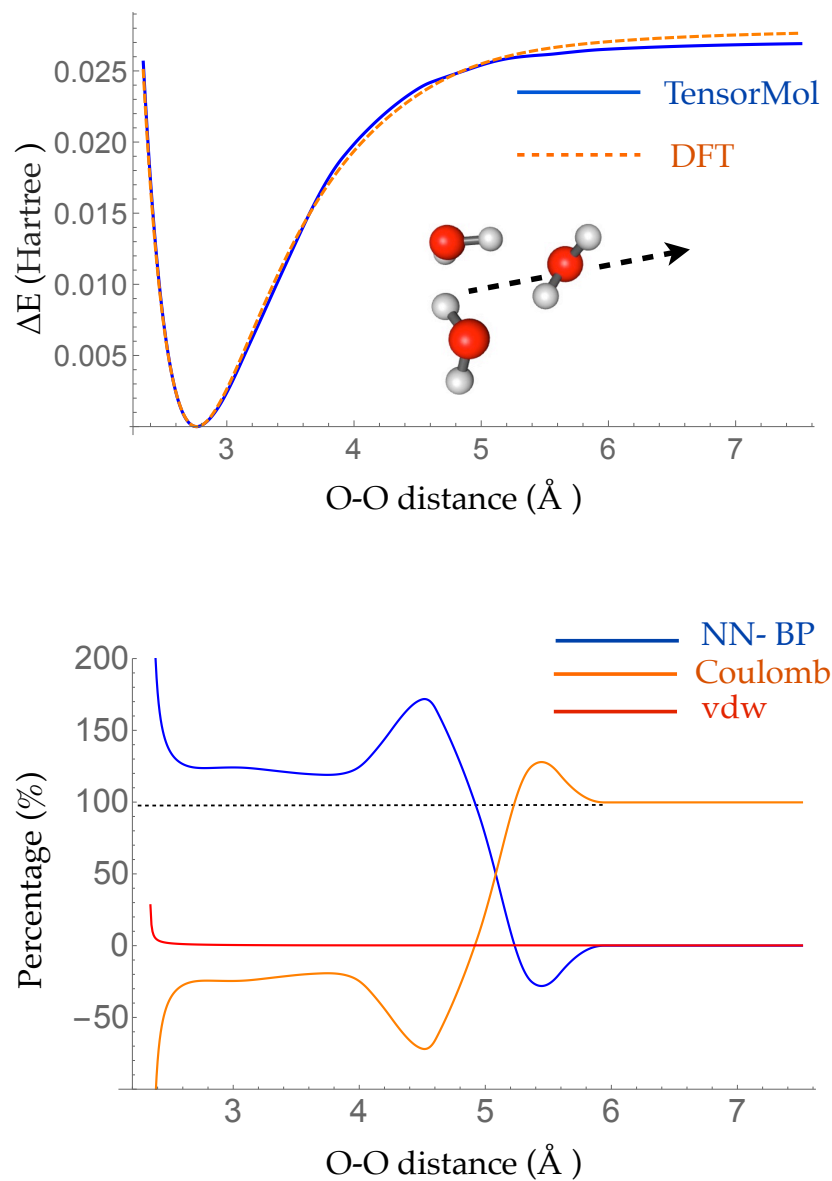


Figure 4.2. Top panel: PES of water trimer when one water is pulled away from the other two. Bottom panel: Percent contribution of binding energy between the water that is pulled away and the other two water molecules from the Behler-Parrinello atom-wise energy, electrostatic energy and van der Waals energy. Behler-Parrinello atom-wise energy contributes most of the binding energy at the short range, and electrostatic energy is the dominant contribution at long range.

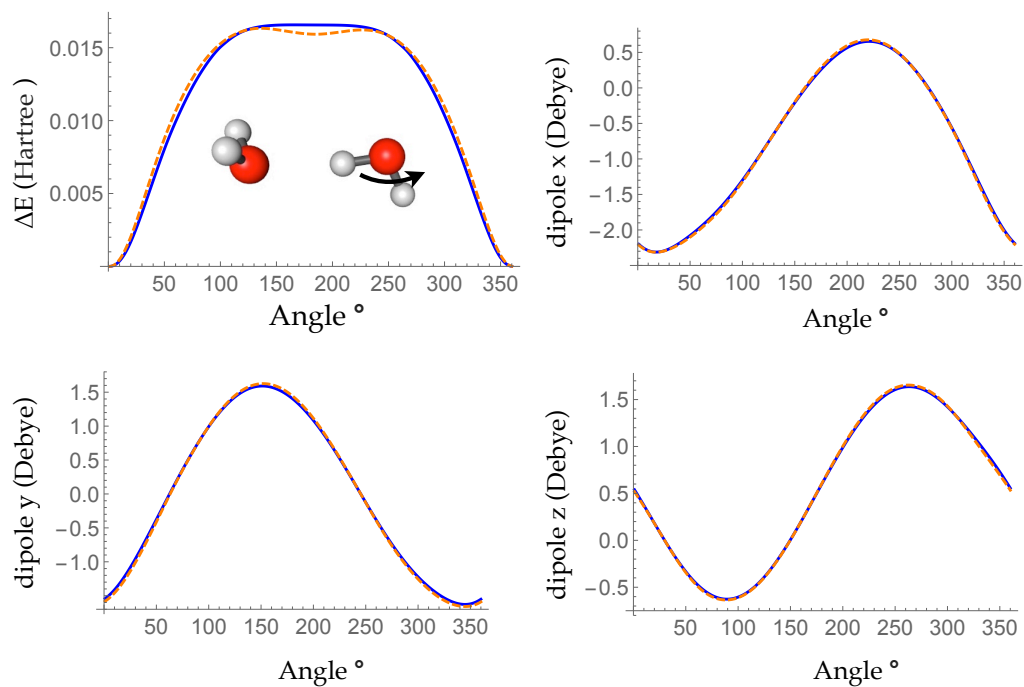


Figure 4.3. Top left panel: PES of breaking a hydrogen bond between two water molecules by rotating one water around the O-H bond. Top right, bottom left and bottom right panels: change of x, y, z, dipole component during the rotation, respectively

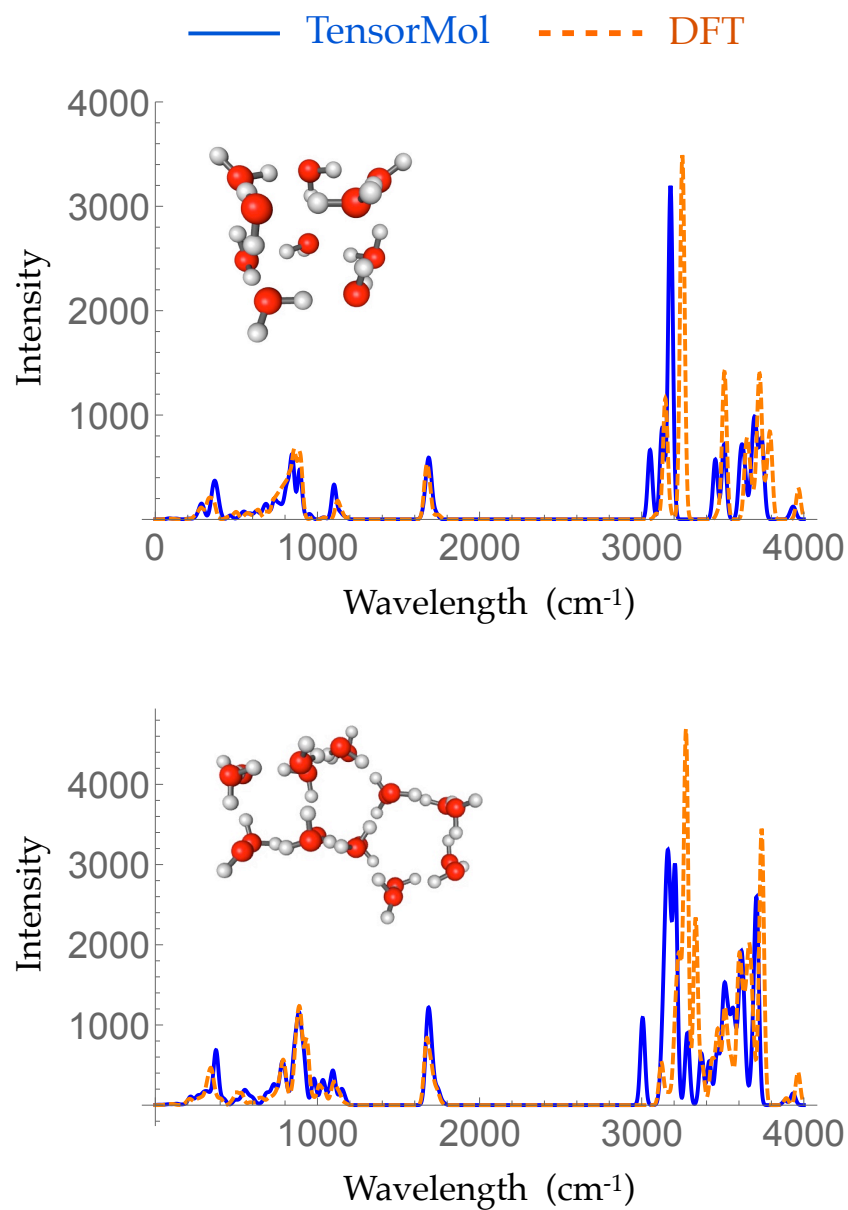


Figure 4.4. Simulated harmonic IR spectrum of 10 water cluster (top panel) and 20 water cluster (bottom panel) generated by ω B97X-D/6-311G** (dashed orange line) and TensorMol force field (solid blue line).

a more stringent test than energies. I used a harmonic approximation to calculate the frequencies and intensities of an IR spectrum for a 10 and 20 water cluster. The result is shown in Figure 4.4, along with the spectra calculated with DFT. The frequency and intensity calculations are performed at their respective minimum energy geometry from the TensorMol model and DFT, so the IR spectra further evidences the ability of TensorMol to be able to accurately produce the correct geometry for non-covalent systems.

4.3.2 Chemspider model

Next I will discuss the Chemspider model. The diversity of molecules to which the Chemspider model is fit compared to the water model greatly increases the complexity of the model. The RMSE of the energy and forces for the testing data were $0.24 \text{ kcal mol}^{-1} \text{ atom}^{-1}$ and $2.4 \text{ kcal mol}^{-1} \text{ \AA}^{-1}$, respectively. More importantly, the network reproduces properties of molecules not included in the training or testing set with accuracy.

We calculated the IR spectra for morphine, which is a molecule not included in our training or testing data. The results are shown in Figure 4.5. In the top right the 3D geometric minimum structure of morphine as optimized by the Chemspider model is given. RMSE of bond-lengths and angles for the minimum geometry is 0.0067 \AA and 1.04° respectively relative to DFT.

In the top left panel of Figure 4.5 the harmonic IR spectrum calculated is shown compared to the same calculation for DFT. The spectra are in good agreement. The MAE of frequencies calculated with the Chemspider model and DFT is 13.7 cm^{-1} . Figure 4.6 shows calculated harmonic IR spectra for four other molecules not included in our training or testing data. Results again compare favorably with DFT for each molecule. The MAE of calculated frequencies relative to DFT is less than 20 cm^{-1} for all four of these molecules.

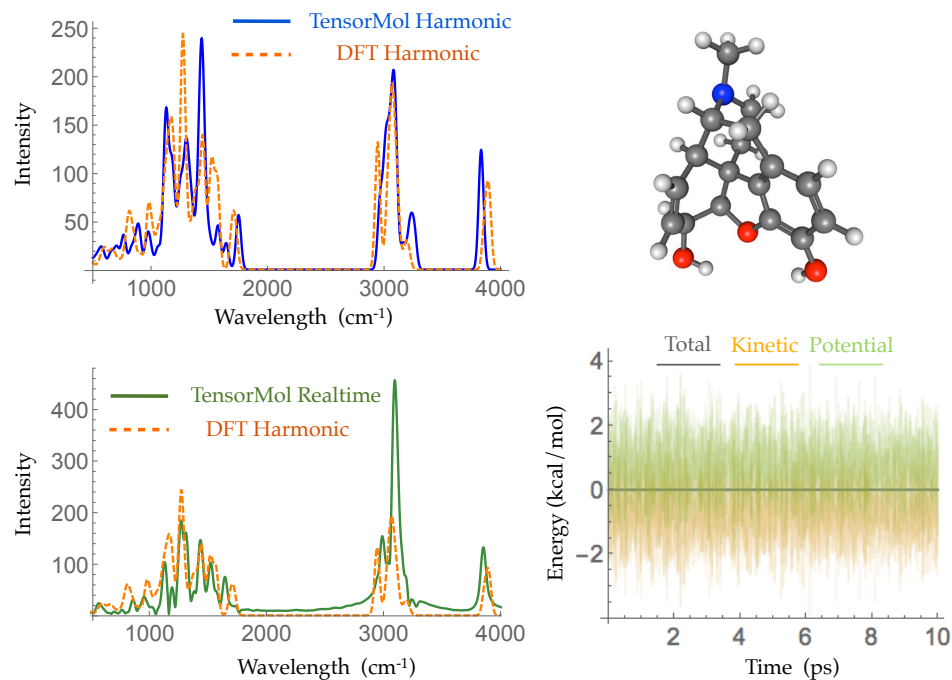


Figure 4.5. Morphine geometry optimized by TensorMol-0.1 (upper right panel) and its harmonic IR spectrum simulated by ω B97X-D/6-311G** (dashed orange line) and TensorMol force field (solid blue line) (upper left panel). Lower panels show TensorMol's real-time IR spectrum vs. DFT (left) and the conservation of energy maintained by the smoothness of the energy (right).

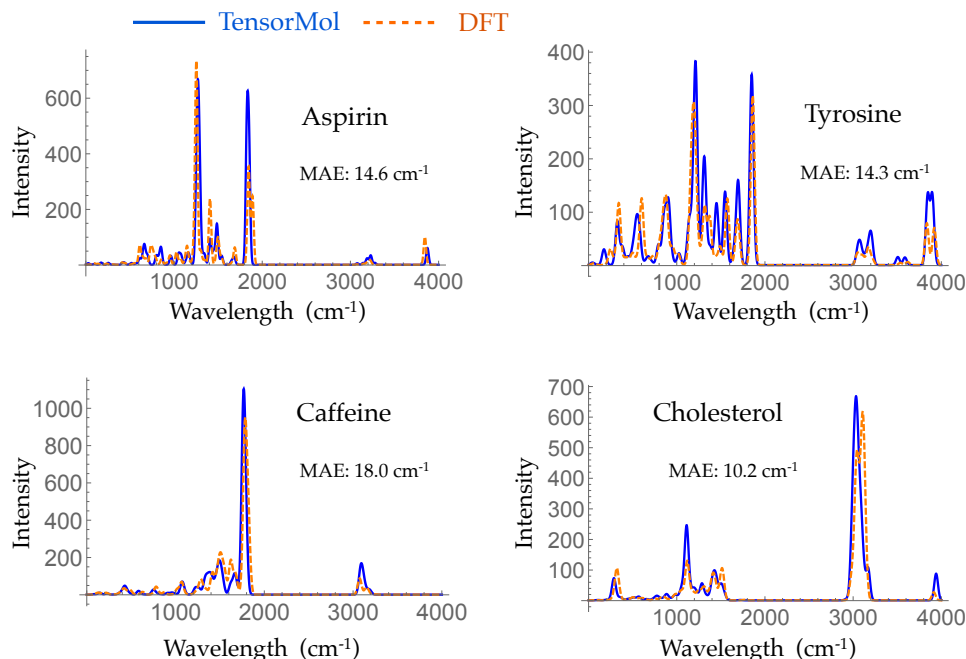


Figure 4.6. Harmonic IR spectrum of four different molecules simulated by $\omega\text{B97X-D/6-311G}^{**}$ (dashed orange line) and TensorMol-0.1. All the molecules are not included in the training set.

Calculating harmonic IR spectra requires quadratically scaling effort. For large molecules it becomes more computationally favorable to simulate the dynamics of the molecule and Fourier transform the dipole-dipole correlation function. I performed this simulation to calculate a real-time IR spectra for morphine which is shown in the bottom left of Figure 4.5. I also show the contribution from the potential and kinetic energy along with the total energy throughout the morphine simulation in the bottom right panel to demonstrate that TensorMol exhibits conservation of energy as well. A sample experimental IR spectrum for morphine is provided in the Supplementary Information for comparison.

TensorMol also exhibits good reproduction of non-covalent interactions. The Chemspider model was used to calculate binding energies for two sets of DNA base pairs: thymine-adenine and guanine-cytosine. Geometries and binding energies from

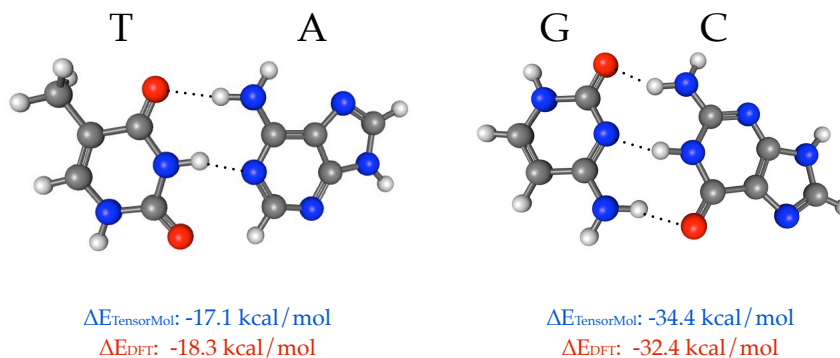


Figure 4.7. Binding energy between the DNA base pairs vs. ω B97x-D with methods at their optimized geometries. The difference between DFT and TensorMol binding energy is ± 2 kcal/mol.

TensorMol and from DFT are shown in Figure 4.7. The binding energy errors by the Chempidder model are $1.2 \text{ kcal mol}^{-1}$ for thymine-adenine and $-2.0 \text{ kcal mol}^{-1}$ for guanine-cytosine.

4.4 Conclusions

I have presented the TensorMol model, a transferable neural network model chemistry. TensorMol is functionally similar to previous highly transferable neural network potentials such as ANI-1[58], but includes long-range physical interactions to augment the exclusively short-range nature of the models based on their feature representations. Incorporating these physical interactions explicitly improves the accuracy and interpretability of TensorMol relative to previous works. I have described two models; one trained exclusively on clusters of water molecules, and one trained with a diverse set of random small organic molecules limited to C, N, O, and H atoms. The accuracy and transferability of these models is shown such that further developments should be pursued to bring the advantages of these models into practice.

One of the major advantages of TensorMol is that the charges are inductive but do not require solving a costly self-consistent polarization equation. Furthermore, the feature representations are implemented with a neighbor list to keep the cost scaling near-linear with system size. This makes TensorMol competitive in terms of computational cost with classical polarizable force fields.

There will be several improvements for TensorMol in future versions. First, splitting the ACSFs into channels for each unique element, or element pair, quickly becomes costly if more than a few elements are considered. The Chemspider model uses ten channels for the angular symmetry function component to encode the atomic numbers of its neighbor pairs when just four elements are included. A better method of encoding atomic numbers which does not grow the size of the feature representation would be a preferable alternative to keep the computational efficiency high. Improvements to the long-range physical interactions, such as a many-body dispersion scheme, would also improve TensorMol further.

CHAPTER 5

FULLY TRANSFERABLE HIGH-DIMENSIONAL NEURAL NETWORK POTENTIALS

5.1 Introduction

In Chapter 4, I presented TensorMol,[57] a neural network model of the potential energy surface that is augmented with long-range physics. TensorMol, along with other similar robust models[58, 117], offers near *ab initio* accuracy at computational costs competitive with classical force fields on a wide range of systems, nevertheless there are still limitations of these models which must be overcome to achieve widespread adoption by practitioners.

TensorMol, along with other robust NN models based on the HDNNP scheme of Behler and Parrinello[5] rely on splitting the geometric features of the ACSFs into channels to encode the atomic identity.[57, 58] The earliest attempts towards this goal used one-hot encodings of atomic number to split features into channels. An analogy may be drawn to digital representations of color images, where red, green, and blue colors are split into channels which collectively make up the image. In this analogy elements correspond to a discrete color channel (i.e. red, green, or blue).

Another limitation of models based off of HDNNPs is that they are parameterized differently for each element; that is, another NN is added to the model for each unique element in the training data. While this allows flexibility in the model such that it can parameterize predictions for each element, it also prohibits the model from sharing learned patterns in data between elements. If each element’s NN shared

parameters, then the model would be able to use transfer learning to improve predictions. Furthermore, the balance of elements in small organic molecules heavily favors C and H and to lesser extents N and O. Incorporating new elements into the training data means that there will be severe imbalances in the number of samples containing less common elements. Using a single NN for each element will help to compensate for this imbalance because the model can use patterns learned about other elements to improve its predictions.

Early models such as TensorMol and ANI-1 have been limited to relatively few unique elements for these reasons. Adding new elements to the training data means the number of channels in the feature representation will grow, and thus the entire model must be retrained from scratch. In Section 4.2.1 I mentioned that the number of channels in the angular symmetry functions from Equation 4.2 grows as $N(N + 1)/2$. Using the same set of ACSF parameters as TensorMol, for just four unique elements (C, N, O, and H), this means there are ten channels with 64 angular functions per channel for every atom in the system. Adding just two more unique elements already doubles the number of channels for the angular functions. Given the RAM constraints of modern GPUs, the scaling of channels in the ACSFs must be addressed.

Furthermore, if the model can be defined to be of constant size with respect to unique elements, then the need to retrain these models from scratch is eliminated. Indeed, the field of natural language processing has recently seen renewed interest after robust models such as BERT[118] and GPT-2[119] which are pretrained on massive datasets before fine-tuning to downstream learning tasks. Similarly, models such as TensorMol could be robustly trained on large datasets and then used to fine-tune a model to a particular system. This would allow practitioners to be able to rapidly develop bespoke potential models with near ab initio accuracy.

In this work, I continue the development of TensorMol by presenting a new ver-

sion which overcomes these issues. The new version of TensorMol is trained over a dataset containing a total of eleven unique elements (C, N, O, H, F, P, S, Cl, Se, Br, and I). The modifications made to TensorMol ensure that the feature representation is a constant size regardless of the number of unique elements in the training data. Furthermore, TensorMol now uses a single NN for each element in both the charge and energy networks. Regardless of the atomic number of an atom, its feature representation is fed through the same network.

5.2 Methods

5.2.1 Deriving the elemental modes

Designing an atomic identity fingerprint requires that each fingerprint be unique for each element. Beyond that, the fingerprint is free to be designed in whatever manner best suites the given model, dataset, and learning target. The most straightforward approach is to use the atomic number as a weighting factor.[120] A similar approach uses the group and period number instead, which parameterizes the model in a way that maps similar elements nearby in the space of the atomic identity fingerprint.[121, 122] De, Bartók, and coworkers used one or two physical properties (e.g. electronegativity) as an encoding which is more physically meaningful.[123, 124] Recently, Zhou et al. published results where they used a large dataset of chemical environments in materials to learn an encoded representation based on that dataset.[125] Finally, several other groups use random initialization and learn an encoding for each element as part of the training process.[56, 117, 126]

I wanted to incorporate the advantages of many of these approaches. Using the period and group number of an element encodes the atomic identity into dimensions which make intuitive sense to chemists. Furthermore, many chemists remember learning about periodic trends of ionization energies or electronegativities as an un-

dergraduate to reason about how different elements might interact. One might reason that a chemist learns an atomic identity fingerprint in their mind which guides their own intuition.

Modeling this concept of a chemists intuition, I collected a set of ten properties for each element which chemists use to reason about what distinguishes one element from another. These properties were used as the input for an autoencoder NN to learn a compressed representation. This compressed representation is referred to as the elemental modes. I take the elemental modes as an atomic identity fingerprint to be used in downstream learning tasks.

An autoencoder is a type of NN which uses the input to the network as the learning target as well. Autoencoder models consist of two NNs; an encoder which transforms the input into a latent representation, and a decoder which decodes that representation back to the same target inputs. By creating an information bottleneck at the latent space, the model is forced to learn a compressed representation of the inputs. Using an autoencoder NN has the benefit of a non-linear compression compared to methods like principal component analysis (PCA).

The ten properties of each element were atomic number, atomic mass, electron occupancy of the valence s-, p-, and d-orbitals, electronegativity, atomic radius, ionization energy, electron affinity, and atomic polarizability. The properties are collected for each element up to Bi. Using these properties, I trained an autoencoder with a reconstruction loss on the properties for each element. A diagram of the input properties and the learned latent space is shown in Figure 5.1.

The expectation is that the elemental modes will contain information regarding how similar two elements might be. For visualization purposes I have performed PCA on the elemental modes and plotted the first and second principal components in Figure 5.2. There is a clear trend which resembles the periodic table with some clustering of different groups of elements despite the fact that period and group

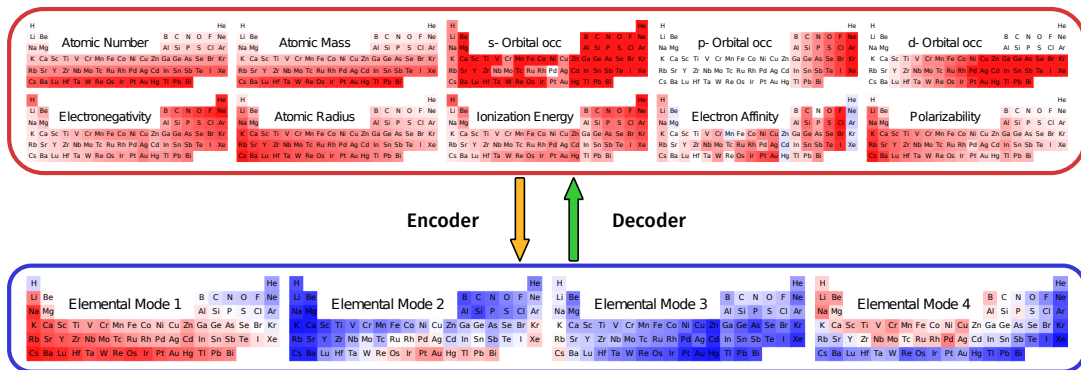


Figure 5.1. Heat maps of ten physical properties used as the feature vectors for each atomic species' input to an auto-encoder neural network shown at the top. The heat map of the latent space output from the encoder shown on the bottom are taken as the elemental modes, a compression of the physical properties used as a representation of atomic species for further machine learning tasks. The decoder network reconstructs the physical properties from the latent space, ensuring as much information is retained in the compression as possible.

were not among the properties used to train the NN. This provides some qualitative evidence that the autoencoder has indeed encoded relevant physical phenomena which is know intuitively correlates with the similarity of the elements.

5.2.2 Learning formation energies of elpasolites

The elemental modes derived in Section 5.2.1 are not exclusively useful for HDNNP models. In general, an atomic identity fingerprint is useful for many ML models of chemical systems. Recently, Faber et al.[127] presented a kernel ridge regression (KRR) model which they trained to predict formation energies of elpasolite crystals by calculating the formation energy with different main group elements at each crystal lattice position. Elpasolite is one of the most abundant crystal prototypes in the Inorganic Crystal Structure Database.[128] The crystal formula for elpasolites is ABC_2D_6 . There are $\sim 2 \times 10^6$ possible elpasolites considering each combination of main group elements up to Bi, many more than can be screened using *ab initio* cal-

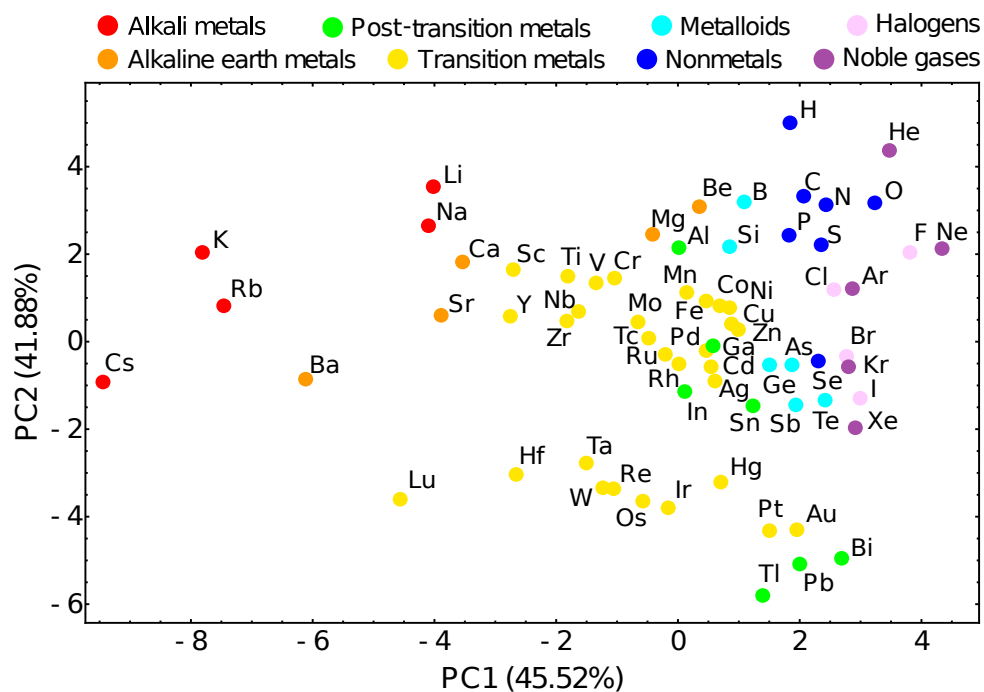


Figure 5.2. The first (PC1) and second (PC2) principal components of the elemental modes color coded by grouping into alkali metals (red), alkaline earth metals (orange), transition metals (yellow), post-transition metals (green), metalloids (teal), nonmetals (blue), halogens (pink), and noble gases (purple).

culations. Faber et al. calculated the formation energy of $\sim 10,000$ elpasolites chosen randomly from the set of $\sim 2 \times 10^6$ possible compositions using the PBE functional[129] within the Vienna *ab initio* simulations package (VASP).

Their feature representation was the group and period number of the atomic species at each lattice position ordered by the lattice position the species occupies. Because each sample has the same crystal structure, the geometry of each sample is constant up to the crystal lattice dimensions. Training a model on a common crystalline geometry allows one to ignore features representing the geometry as an input to the network. The network will learn to bias its predictions to this crystal geometry since it will see no other samples. Because of this, I propose that this learning task makes a good test case for the elemental modes. Using the elemental modes in this learning task allows us to examine how well these features may perform independently of being combined with geometric features.

I trained a NN model to predict the formation energies of elpasolites using the elemental modes as the feature representations. The feature vector for each sample elpasolite is the ordered set of elemental modes for each atom at the crystal lattice sites, analogously to the work of Faber et al. The NN model consisted of two hidden layers with 128 neurons in each hidden layer. The softplus activation function was used to introduce non-linearities.

5.2.3 Changes to HDNNP model

Previous works using HDNNP models have relied on separately parameterized NNs for each element. The advantage of this approach is that the model is given greater flexibility in making predictions which can lead to more accurate predictions. On the other hand this scheme prevents sharing of information between the NNs, which may improve the convergence or accuracy of the overall model as well.

Furthermore, previous works have typically been limited to a few unique elements.

Robust models such as ANI-1[58] and the previous TensorMol work[57] used training sets of small organic molecules limited to C, N, O, and H atoms. Including samples with more unique elements can lead to drastic imbalances in the dataset. The diversity of small organic molecules is such that there will be many samples with the aforementioned elements and relatively fewer samples containing other elements. Because training NNs requires stochastic optimization methods, a mini-batch of random samples from the training data has a high probability of containing few or no samples with certain elements if the imbalance is significant enough. Stochastic optimization is meant to approximate the gradients of the parameters of the full training set by using the gradients from random mini-batches of the training data. Mini-batches with too few samples containing certain elements then will have no gradient to these parameters or will have a poor approximate gradient. For this reason, it becomes impractical to train a HDNNP model if any elements are sparse in the training data because the updates to its parameters will be poor.

To remedy these issues I propose to use a single NN for every element. This can rectify issues with data imbalances since the elements will share all parameters. In a model where each element has a separately parameterized NN, it is reasonable to assume that each NN will learn many of the same trends for different elements. I suggest that using a single NN for all elements may even improve the training of the model to be more accurate and more robust. The network can learn to make reasonable predictions for elements which there is little training data by what it learns for elements with significantly more data.

One issue that arises with using a single NN for each element is that there is no way for the model to infer the identity of the central atom. The model should be able to share information between the weights for each element, but the predictions for one element in an identical local environment compared to another should differ to a degree. Previously, this was incorporated by a distinctly parameterized NN

for each element. In this work, I will incorporate this information into the feature vector to overcome this issue. Alternative methods could be explored, such as sharing the earliest layers of parameters for each element but parameterizing the last layers separately.

5.2.4 Adjustments to ACSFs

I begin with the formalism of the ACSFs given by Gastegger et al.[120]. The radial functions are given by

$$G_m^R = \sum_{j \neq i} g(Z_j) \exp[-\eta(R_{ij} - R_s)^2] f_c(R_{ij}) \quad (5.1)$$

and the angular functions are given by

$$G_m^A = 2^{1-\zeta} \sum_{j,k \neq i} h(Z_j, Z_k) (1 + \cos(\theta_{ijk} - \theta_s))^\zeta \\ \times \exp \left[-\eta \left(\frac{R_{ij} + R_{ik}}{2} - R_s \right)^2 \right] f_c(R_{ij}) f_c(R_{ik}), \quad (5.2)$$

where these equations are identical to equations 4.1 and 4.2 in Section 4.2.1 but the functions $g(Z_j)$ and $h(Z_j, Z_k)$ are included in Equations 5.1 and 5.2, respectively. Note that Gastegger and coworkers use the form of the ACSFs as originally described by Behler and Parinello,[47] where as I have retained the ANI-1 variant of the ACSFs as was done in the original TensorMol work.[57] The functions $g(Z_j)$ and $h(Z_j, Z_k)$ are weighting functions which modify the contribution of each symmetry function based on the atomic identity of the neighboring atoms.

In the original TensorMol and ANI-1 models, $g(Z_j)$ and $h(Z_j, Z_k)$ are functions which calculate the outer product of the geometric features vector with a one-hot encoding vector. Gastegger et al. suggested functions as simple as weighting each feature by the atomic number may be satisfactory, though the scope of their work

was limited to equilibrium structures from the QM9 dataset.[72] Indeed, the authors point out that $g(Z_j)$ and $h(Z_j, Z_k)$ may take many forms.

Here I implemented $g(Z_j)$ and $h(Z_j, Z_k)$ by using the elemental mode vectors derived above. $g(Z_j)$ takes the outer product of the elemental modes vector for atom j in the local environment. The elemental modes vectors will scale the Gaussian peaks in each channel by a constant factor. The network can then infer the identity of atom j by how identical radial features scale across the channels of the network.

Channeling features for the angular functions is more difficult because of the three-body nature of the functions. The features must remain invariant to the ordering of neighbor atoms in the local environment. Therefore $h(Z_j, Z_k)$ must result in the same set of features as $h(Z_k, Z_j)$. One of the most straightforward ways to achieve this is to first take the element-wise product of the two elemental modes vectors. The resulting vector is then used similarly to $g(Z_j)$ to take an outer product with the angular features.

In Section 5.2.3, I discussed incorporating the identity of the central atom into the feature representation. One could imagine a tensor of feature vectors of dimension $N_E \times N_{ACSFs} \times N_{channels}$ where N_E is the number of unique elements, N_{ACSFs} is the number of ACSFs, and $N_{channels}$ is the number of channels created by the elemental mode vectors. The submatrices along the first dimension of this tensor correspond to the feature vector of different elements in an identical local environment.

Calculating and storing a tensor of this size for each atom in a mini-batch is impractical for the memory constraints of current GPU architectures. The same relationship can be learned as a tensor decomposition. Indeed, there is previous work in the field of natural language processing where a tensor decomposition is used in a character-level recurrent NN (RNN).[130] Their work imagines characters as embedded in words. A letter has some meaning on its own, but also has a meaning which depends on the context of the surrounding letters. They use a tensor decomposition

to learn a transformation from the current context vector into features that represents the next letter in the word within that context.

Similarly we can think of learning a transformation from the context of the local environment to a feature representation which composes the local environment with the identity of the central atom. To achieve this, I introduce two learnable matrices P and Q . These matrices will be initialized randomly and trained with the rest of the parameters in the model. The tensor decomposition is incorporated as

$$\tilde{G}_i = P[G_i(\beta_i Q)] \quad (5.3)$$

where β_i and G_i are the elemental mode vector and feature vector for atom i . First, I calculate the vector which results from the product of β_i and Q . This is a linear transformation which maps the elemental mode vectors into a new vector of the same dimension to represent the identity of the central atom. The new vector then scales each channel in the original feature vector G_i . Finally, another linear transformation is applied by P to encode this identity into the feature vector. \tilde{G}_i becomes the new feature vector for atom i which now incorporates the local environment of the atom with its own identity vector. This should allow the model to learn patterns in how the feature vector changes based on the identity of the central atom, and to adjust its predictions accordingly.

5.2.5 Data generation

As mentioned in Section 5.2.3, the imbalances present in the elements occurring in small organic molecules can be an issue. Changes to the TensorMol model are intended to alleviate this concern, but it is still desirable to incorporate as much diversity in the dataset as possible. The updates of the parameters in the model during training are derived from the average gradients of the error in the predictions

for each sample. This averaged gradient can be thought of as an equally weighted contribution from each atom in all samples in the mini-batch. If a mini-batch contains few or no samples with certain elements, the gradients from those elements will be drowned out by elements of which there are many samples. Therefore, it is important to be cautious about drastic imbalances in the training data.

Similarly to the previous TensorMol implementation,[57] our data is generated by first collecting a set of unique molecules from ChempSpider.[116] I set several constraints on which molecules are allowed in the dataset. First, each molecule must contain only C, N, O, F, P, S, Cl, Se, Br, I, or H. This significantly expands the allowed set of molecules beyond previous works to many new elements. Even now most other models have been limited to at most eight unique elements.[131, 132]

Sampling small organic molecules results in drastically more samples with C, N, O, and H atoms than other elements. For this reason, I wanted to ensure the data was not severely imbalanced towards these four elements by adding a constraint on the dataset requiring that each molecule must contain at least one atom of F, P, S, Cl, Se, Br, or I. This ensures that each mini-batch will contain a greater diversity of elements in each sample and provides better gradients with stronger signals from the less frequent elements to update the parameters of the model.

Finally, the size of the molecules must not exceed 40 atoms in total, including hydrogen. This constraint is simply to keep the cost of generating reference data to train the model to a minimum. Previous works have shown that similar models are able to generalize predictions for larger molecules based on training on datasets of small molecules.[58, 102] I have increased the maximum number of atoms per sample as compared to previous datasets for similar tasks[71] because there are many fewer known organic molecules with the necessary diversity of elements for building this dataset. Increasing the maximum number of atoms allows us to collect more diverse molecules for our dataset to avoid drastic data imbalances. The largest molecules

in our dataset contain up to 40 heavy atoms, where a heavy atom is anything other than hydrogen.

Given these constraints, I collected a set of ~45,000 unique molecules from the ChempSpider database. Each molecule was first geometry optimized with Q-Chem and ω B97X-D/6-311G**. Next, the minimized geometries were used as initial geometries for a metadynamics simulation to collect sample geometries for each molecule following our previously outlined procedure.[70] In all, the final dataset contained ~1.35 million geometries from these molecules. A breakdown of the composition of the atoms and molecules in the dataset is given in Table 5.1.

This dataset is split into an 80:20 ratio for training and testing data used for developing the HDNNP model described in Section 5.2.3. The model consists of two NNs; a charge network and an energy network. Each network contains three hidden layers with 512 neurons per layer. I again used the softplus activation. The choice of activation function is particularly important for training models from which it is desired to produce smooth atomic force gradients. The forces predicted by the model are calculated by differentiating the predicted energy back through the parameters of the model. To ensure smooth predictions of forces, the activation function must have a continuous first-order derivative. In Chapter 4, I discussed adding atomic force gradients into the loss function of the network. Training the model by including forces requires the calculation of second-order derivatives of the model parameters. The softplus activation function is continuously differentiable up to arbitrary order, so it makes a suitable activation function for these types of NN potential models.

In Chapter 4 the TensorMol model was parameterized to fit atomic charges based on the molecular dipole[57] because it is a true quantum mechanical observable and changes in the dipole are smooth with respect to the geometry of the system. This ensures that charges predicted by the model are similarly smooth. On the other hand, partial charges from a charge partitioning scheme represent a much larger wealth of

TABLE 5.1

INFORMATION ABOUT THE ABUNDANCE OF EACH ATOMIC
SPECIES IN THE TRAINING DATA SET FOR THE HD-NNP.

Element	Atom count	Molecule count
H	1.79×10^7	1.35×10^6
C	1.57×10^7	1.35×10^6
N	3.15×10^6	1.15×10^6
O	2.47×10^6	9.61×10^5
F	1.27×10^6	4.78×10^5
P	7.60×10^4	5.32×10^4
S	1.11×10^6	6.01×10^5
Cl	5.81×10^5	2.96×10^5
Se	1.55×10^4	1.44×10^4
Br	3.10×10^4	2.88×10^4
I	2.97×10^4	2.60×10^4

Atom counts enumerates the number of atoms of each type in the whole data set. Molecule counts enumerate how many unique geometries contain each atomic species.

data than molecular dipoles, since each atom in the molecule will have an assigned partial charge. This can be beneficial when the size of the dataset is small and to help with generalization of the model to new data. In this work the charge network is trained using Mulliken charges as a learning target for each atom.[133]

5.2.6 Simulating alchemical intermediates

Free energy perturbation methods are a valuable tool in the drug discovery community.[134–138] Alchemical perturbations (i.e. alchemically changing one atom into a different element) can be used to calculate differences in, for example, binding free energies of different molecules in a protein binding pocket. These binding energies often may differ by less than a kcal mol^{-1} . [139] The accuracy of classical force fields may not be sufficient to measure these differences. NN potential models offer accuracy comparable to their *ab initio* training data with significantly reduced computational effort. Developing a model able to perform alchemical perturbations may offer a way to calculate these small differences in free energies more accurately.

Classical force fields are parameterized based on bonding, which makes it difficult to run simulations with reactive chemical processes. ReaxFF offers reactive simulations with a classical force field,[140] however, using ReaxFF is often limited due to poor energy conservation.[141] NN potentials typically use an ansatz based on local atomic environments, which makes simulating bond changes easier. Using a NN potential model then may enable researchers to perform alchemical perturbation simulations where bonding changes may be important as well.

I propose that a simple way to perform an alchemical perturbation is to perform a linear interpolation of the feature vectors for the atoms in the system. The initial state and final state of the system share coordinates but have different atomic numbers which correspond to the same coordinates. I calculate the feature vectors for each

atom in both systems and interpolate the corresponding feature vectors by

$$\tilde{G}_i^{\text{alc}} = (1 - \lambda)\tilde{G}_i^1 + \lambda\tilde{G}_i^2 \quad (5.4)$$

where \tilde{G}_i^1 and \tilde{G}_i^2 are the feature vectors for atom i in the first and second states. λ is a switching parameter used to interpolate between the two states. \tilde{G}_i^{alc} is used as the input to the model for the alchemically intermediate state.

5.3 Results

5.3.1 Elpasolite model

The model trained to predict formation energies of elpasolites showed an MAE of 67 meV/atom on the independent test set. Faber et al. achieved an accuracy of about 100 meV/atom on the same dataset.[127] Zhou et al. also used their learned atomic identity fingerprint on this task with a NN and achieved an accuracy of about 150 meV/atom.[125] Our atomic identity fingerprint has shown an improved performance over previous works of the same dataset. The distribution of errors is shown in Figure 5.3. The errors approximately follow a normal distribution.

While the model in the work has shown improvements over previous works, the main results is that the elemental modes are shown to be an effective feature representation for atomic identities. These types of features are commonly needed in many ML models for chemical applications.[30, 142]

5.3.2 HDNNP model

I trained the HDNNP model on the dataset as described in Sections 5.2.3 - 5.2.5. The root mean square error (RMSE) on the independent test set for the energy and atomic forces is 0.0976 kcal/mol per atom and 3.71 kcal/mol/Å respectively. The accuracy is similar to previous works trained on datasets with many fewer el-

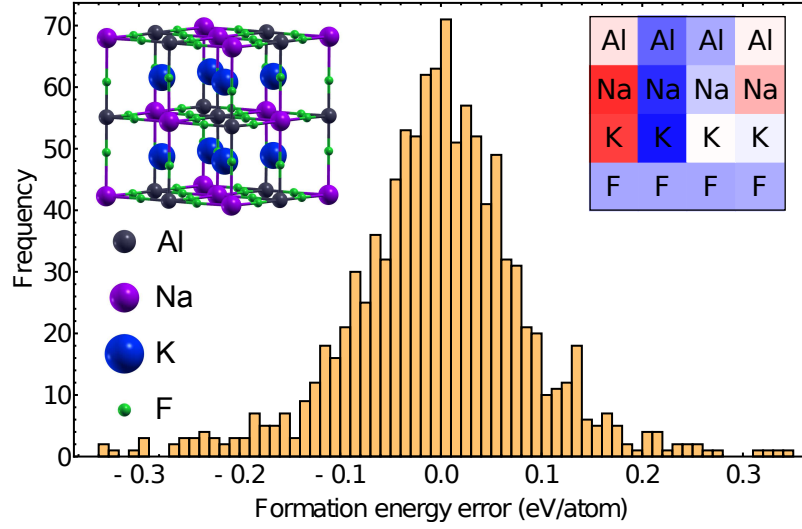


Figure 5.3. Formation energy error prediction for the independent test set of elpasolite crystals with chemical formula ABC_2D_6 . Inset in the top left is the crystal structure for the prototypical elpasolite ($AlNaK_2F_6$) with atom labels shown in the bottom left. Inset in the top right is a matrix plot of the elemental modes vectors for $AlNaK_2F_6$, which is flattened to a 1D vector before input to the model.

ements.[57, 58, 117] Incorporating active learning methods and a larger dataset are likely to further improve the model accuracy. Recent works have begun to explore the feature representations used for HDNNP models and other similar data-driven potential models to discover where issues may arise with the geometric features.[76] Improvements in this regard are likely to similarly improve the accuracy.

Aside from higher accuracy based on metrics of the model errors, the model presented here has several qualitative differences from previous works which should be examined for their effectiveness. One of the major changes was to use a single NN for each element in the training data. In Section 5.2.3, I suggested that using a single NN for each element would likely lead to improvements of the model based on information sharing and dataset imbalances. To examine to what degree this may occur, I trained another identical model on the same dataset but removed any molecules containing Cl atoms. I took the subset of molecules with Cl atoms from the independent test set used for the model trained on the full dataset as a new test set to compare between the networks trained with and without Cl atoms. The average number of Cl atoms per molecule in this test set is 1.96.

The embedded atomic energies are the predictions which come directly from the energy network. They are not a learning target, and the model may make slightly different predictions depending on random weight initialization, ordering of the training data during the model fitting, or other causes of non-determinism in the final parameterized model. Nonetheless, for two models trained identically these differences in predictions should be minimal, especially when the amount of training data is large (i.e. the model is less likely to overfit). Thus a qualitative comparison can be made of the embedded atomic energies predicted by each model.

In Figure 5.4, I have shown the distributions of embedded atomic energies predicted for the N and Cl atoms from the test set with both models. The top panel shows the distributions of atomic energies predicted by both networks for N. The

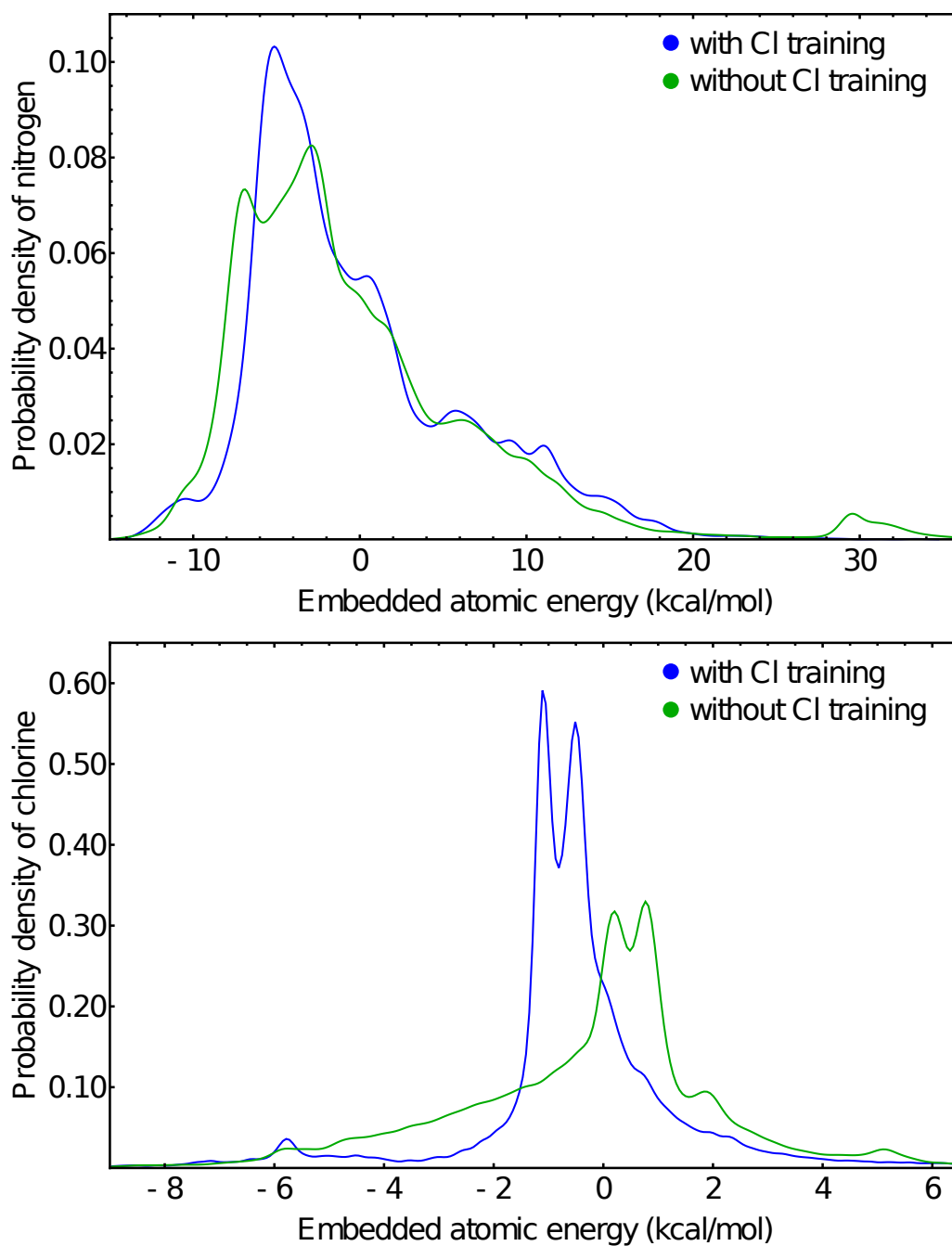


Figure 5.4. Distribution of embedded atomic energies predicted by two neural networks trained with (blue) and without (green) Cl-containing molecules. Top and bottom panels are for nitrogen and chlorine embedded atomic energy predictions respectively.

distributions are very similar, showing that the models indeed may differ slightly but their predictions will not differ significantly. This test set contains only molecules with at least one Cl atom, thus some of the predictions for N are based on local environments which have a Cl atom within the cutoff distance of the ACSFs. The network trained without Cl has also not seen any samples containing Cl within the local environment of a N atom. Thus some differences should also be attributed to this.

Comparing both models’ predictions for Cl atoms again shows a strikingly similar distribution of embedded atomic energies, however, the distributions have a shifted mean relative to each other. Both models exhibit distributions which show a similar separation of two approximately normal distributions. From this, it can be ascertained that the model has indeed learned to make reasonable predictions for Cl atoms without any training data containing Cl.

Comparing the errors for total molecular energy of each model allows us to quantitatively analyze the differences between each model. In the Cl test set the MAE and RMSE were 1.31 kcal/mol and 1.63 kcal/mol, respectively, for the network trained with the full data set and 9.20 kcal/mol and 12.74 kcal/mol, respectively, for the model trained without Cl data. While the errors are only about an order of magnitude larger when trained without Cl, this is evidence that the model successfully transfers knowledge learned about different elements to improve its predictions. Indeed it can be beneficial to use a single NN in the model.

A further analysis should compare the model presented with one trained in the earlier style with individually parameterized NNs for each element. I discussed one issue that might cause difficulty when training a model like this with a dataset of small molecules and a very diverse set of elements in Section 5.2.3. Due to the stochastic nature of the training process, severe imbalances in the number of samples with less common elements could result in issues with the gradients used to update the

parameters of the NNs. There are techniques used to overcome these issues, but in the case of training an HDNNP model, it may not be straightforward because the issue arises due to parameters not being shared by elements but the loss function depending on the output for multiple elements. The parameters of the model must be trained in unison so it becomes more difficult to separate the training of the subnetworks to accommodate an algorithm which might improve the data imbalance.

5.3.3 Alchemical intermediate simulation

To examine if using the model described in this Chapter to simulate alchemical intermediates has the potential for further development, I ran free energy simulations where I made a gradual alchemical transition over the course of 3 ps of simulation time. The purpose is to examine how smoothly the potential changes with respect to the linear interpolation of the feature vectors. Because free energy is a state function, the accuracy of the alchemical intermediate states is irrelevant so long as the potential energy surface changes smoothly between the two states of the system.

For a test case, I chose to examine an alchemical transition between an ethanol dimer and a water hexamer. This makes for a good first test because the carbon atoms can be interpolated into oxygen atoms without the need to destroy or create any atoms. This makes the simulation easier to set up because destroying or creating atoms requires more care to ensure that the alchemical intermediate potential surfaces do not exhibit any regions where the gradients become unstable. For example, with a classical force field it is typically recommended to smoothly turn off electrostatic interactions first to prevent attractive forces from causing atoms to collide before smoothly decaying other interactions in the force field.

The potential energy during a 10 ps simulation is shown in Figure 5.5. The initial geometry is an ethanol dimer for which I propagate dynamics with using the HDNNP model discussed in Section 5.3.2 for energies and forces and a 0.5 fs time step. After

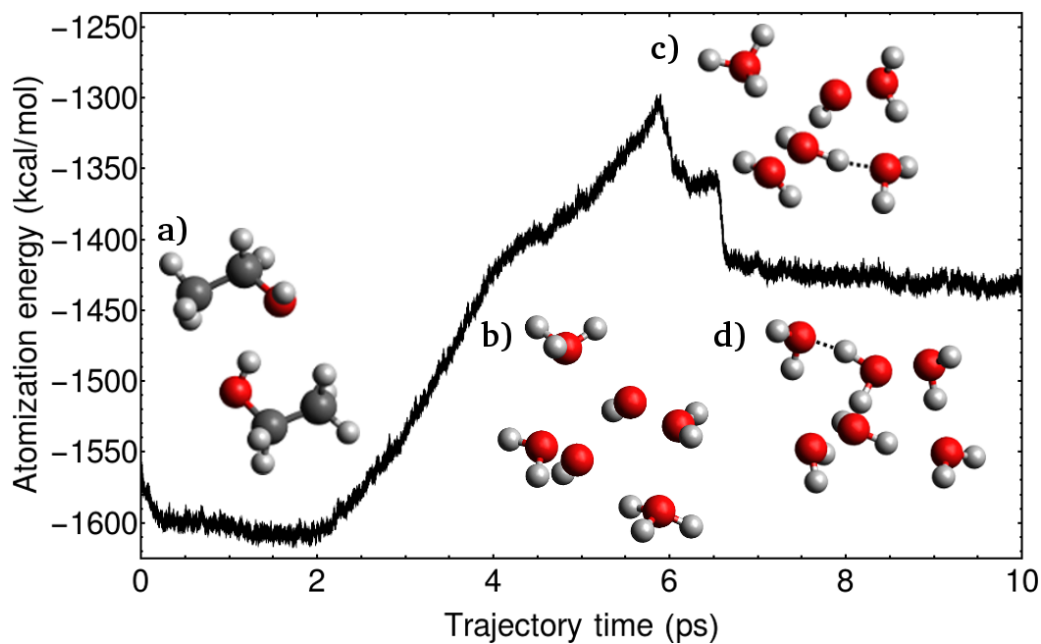


Figure 5.5. Atomization energy during an MD trajectory with an alchemical transformation of an ethanol dimer into a water hexamer over a 10 ps simulation time. Insets: a) Initial ethanol dimer geometry. b) Ethanol dimer geometry after 2 ps of simulation time. c) Water hexamer after the alchemical transformation completes at 5 ps. d) The first and second proton transfers at about 6 and 6.5 ps. Proton transfer is denoted with a dashed line.

two ps the alchemical transition is initiated by slowly incrementing λ from Equation 5.4 to interpolate the feature vectors of the dimer and water systems. Over the course of three ps (6,000 time steps), λ is incremented at each step to interpolate between the two systems. Observing this region of the potential surface in Figure 5.5, it is seen that the potential changes smoothly during the transition. After the transition finishes the water hexamer reequilibrates itself through two proton transfers occurring around 6-7 ps of simulation time. Direct observation of the simulation shows no obvious signs of pathological behaviour which would be problematic for free energy perturbations.

5.4 Conclusions

I presented an improved TensorMol model which is able to calculate energies and forces for up to 11 elements. The major modifications made to TensorMol include incorporating the elemental modes into the ACSFs to make the dimension of the NN constant with the number of elements in the training data and using a single NN for each element in the charge and energy networks. The accuracy of the new version of TensorMol is similar to that of earlier models, but it is able to calculate energies, forces, and partial charges for more elements, enabling the ability to simulate many more systems.

These changes may be necessary to develop models able to treat many different elements. I have discussed the challenges resulting from imbalances in datasets of small organic molecules which would likely lead to issues with the gradients of the parameters of the model if the elements continued to be parameterized separately. This issue should be studied further to determine how severe this problem is, and if it can be overcome by using an algorithm for filling mini-batches of data strategically. Training a model of the former type with as many elements would require a significantly larger training set than this work.

I have shown that these modifications may indeed even make the model more robust. By sharing parameters, the network is effectively trained on a much larger dataset. Distinctly parameterizing NNs for each element means each subnetwork will be trained on all atoms of the corresponding element. The modified TensorMol uses a single NN which is thus trained on every atom regardless of atomic number. These improvements were shown to make TensorMol able to extrapolate a reasonable prediction to a new element with which it had never been trained. This demonstrates that sharing parameters indeed encourages improvements in training of the model, in particular if some elements may be sparse in the training data.

I have also shown that TensorMol may be a suitable model for the calculation of alchemical free energy perturbations. This is due to the incorporation of the atomic identity fingerprint developed in Section 5.2.1. The elemental modes, as I have referred to them, are a compression of physical measurements and calculations that chemist intuitively understand as describing fundamental similarities and differences between the elements. The way in which I have incorporated them into the ACSFs uses the elemental modes as a continuous dimensional space. This parameterization enables the model to interpolate between elements to improve predictions and to enable the simulation of alchemical intermediates.

The first test of the elemental modes on a dataset of formation energies for elpasolites showed that the elemental modes are able to achieve good accuracy on a learning task, which is mostly dependent on the identity of the atoms present at each crystal lattice position. This simple example shows that the derived representation may be suitable for many other types of chemical ML models. The dimension of the elemental modes is small to keep models computationally efficient in this work, but if more robust or improved fingerprints are needed by incorporating more physical properties, or even adding another dimension to the fingerprint (e.g. fingerprints could represent element and another characteristic, such as orbital hybridization based on the local

geometry) the size of the fingerprint can be trivially increased to accommodate the greater complexity.

CHAPTER 6

GRAPH NETWORKS FOR REACTION OUTCOME PREDICTION

6.1 Introduction

Another area where ML model may have a significant impact on the field of chemistry is the prediction of organic reaction products. Accurate methods to predict reactions which lead to desired products would accelerate the development of many novel compounds which may be desirable drug candidates or many other purposes. Recently, ML models have exhibited the potential to make significant contributions towards improved screening methods.

This field has been active for many decades. Early models were based on expert coded heuristics to define mechanistic reactions.[143–146] The goal of these projects was to develop new chemistry of previously unknown reactions. Unfortunately, these methods have not been adopted into mainstream use by chemists due to their limitations. These systems were developed by hand-coded rules which required considerable effort and the expertise of chemists. This meant that the scopes of these systems were limited.

With the current surge of interest the ML field is experiencing, and the ever increasing availability of large datasets, many of these problems for which it has been historically difficult to develop algorithms for are receiving renewed interest by researchers employing data-driven methods. Some early examples of incorporating data-driven approaches include using ML models with traditional reaction templates to rank the most likely template or rank predicted products from the templates.[35,

147, 148] Reaction templates were introduced by Corey and Jorgensen along with their synthesis planning software Logic and Heuristics Applied to Synthetic Analysis (LHASA).[149] Ever since, reaction templates have continued to find use for chemical synthesis planning.

Reaction templates, however, provide little flexibility and thus are not ideal for discovering novel chemical reactions. Data-driven approaches which depart from reaction templates show potential to provide more flexibility in their predictions. Some works have predicted mechanistic steps with ML models,[150–152] but these methods require human-annotated data along with published experimental results. Another approach models the problem as a machine translation task analogously to natural language processing models for translations of text between foreign languages.[27, 28]

More recently, graph convolution networks have received a lot of attention in the molecular sciences. Traditional convolutional neural networks (CNNs) take advantage of symmetries in data by convolving filters over regularly spaced data, such as the pixels in an image. The advent of CNNs was a crucial aspect of the recent leaps in accuracy achieved by the image recognition community since Krizhevsky and coworkers introduced their deep CNN model trained on the Imagenet dataset.[1] Graph convolutions are the abstraction of CNNs to graph structured data.

There is a long history of chemical graph theory.[153, 154] Representing molecules and reactions as graphs is a natural fit. Atoms are represented as nodes which are connected by bonds, represented as edges in the graph. Information passes through the edges to make updates to the nodes based on their connections to other nodes. Graph convolutional network models have been the subject of much research within the ML community over the past several years.[155–158] Molecular graph networks have similarly received a lot of interest.[159–161]

Organic reaction prediction has seen an influx of efforts using graph network models.[29–31] In particular, the work by Coley et al.[30] achieved considerably higher

accuracy on the US Patent and Trademark Office dataset than a Seq-to-Seq model trained on the same dataset.[28, 162] The number of parameters for the graph network of Coley et al. also has approximately an order of magnitude fewer parameters than the Seq-to-Seq model, evidencing the advantages of using graph representations of molecules.

The work of Coley et al.[30] and many others[27, 28, 31] has a goal of searching for new chemistry; i.e. their work is intended to identify new reactions through data-driven methods based on what reactions are already well known in the literature. The application of known reactions is still somewhat of an art form for synthetic chemists. One must not only worry about whether or not a reaction will proceed, but also will the reaction yield be sufficient to obtain enough product or what the side products will be. Often, each reaction will be only a single step in a chain of multiple reactions needed to synthesize the final product. High yields of compounds at each step are necessary to keep the overall yield reasonably high through a long sequence.

Further complicating matters is stereochemistry. Stereoisomers have drastically different effects in medicine, but obtaining enantioenriched products can be challenging because $\Delta\Delta G^\ddagger$, the difference of the change in free-energy for the two transition states leading to the stereoisomers of the product, is often on the order of a few kcal mol⁻¹. [163] Predicting these subtle differences would not be captured by a model trained on a dataset as diverse or noisy as the USPTO. Transformer models have been shown to handle stereochemical information,[164] but with graph networks most proposed models to date do not accommodate stereochemistry.[165]

To achieve quantitative accuracy, researchers must use domain knowledge to provide the model with sufficient information to determine the patterns necessary to predict yields and stereochemical preferences. This requires a loss of generality in favor of models which are efficient and accurate, using information which may be

critical for a particular subset of reactions, but not for others. For example, the Doyle group recently published work using ML models to predict yields from deoxyfluorination reactions with sulfonyl fluorides.[166] Related work from the Sigman group details regressive models of enantioselectivity by similarly instituting domain knowledge of subsets of related reactions.

To achieve next generation ML models of reaction yield prediction and over-the-arrow reaction optimization, one logical path is to combine the success of data-driven graph models with the expert knowledge chemists have been able to gain about correlations between measured or calculated properties and reaction yields and stereoselectivities. The first step toward this goal is the prediction of reaction yields. This work is the early stages of building a graph model based on the work of Coley et al.[30] and incorporating select properties similar to the work from the Doyle group[166] to make quantitative predictions of reaction yields. Combining expert knowledge with the power of the graph representation incorporates multiple types of domain knowledge to improve the model predictions.

The first step in this work is to reproduce the result of Coley et al. While training on a dataset such as the USPTO will not provide enough information for quantitative predictions, transfer learning has shown considerable benefit in the ML community. In particular BERT[118] and GPT-2[119] received considerable recognition for their success using unsupervised pre-training on massive corpora of human generated text which resulted in improved models when the parameters are fine-tuned on smaller datasets for supervised learning tasks. Furthermore, reaction yields require high-throughput experimentation to achieve considerable amounts of data, but the analysis of the results can still be a limiting factor. The magnitude of datasets with reaction yields available is considerably smaller than is typical for NN models. It may be beneficial to pre-train a graph model on the USPTO datasets to alleviate issues with the size of reaction yield datasets.

6.2 Methods

6.2.1 Graph model

The first part of the model builds representations of atoms or other graph objects by convolving filters over the neighbors of atoms, typically defined as those which are bonded to the atom, to update its own features. The process is iterative, so at each successive step the filters incorporate information from more distant atoms through the updates to the features of an atom’s neighbor. Typically, three to four iterations are included for molecular graph models.

Atoms and bonds each have a set of initial features that represent the nodes and edges in the graph, respectively. Initial features are either encoded into one-hot vectors (e.g. atomic number) or take binary values (e.g. atom is in a ring or not). Atom features include atomic number, atom degree, valence, and whether the atom is part of an aromatic system or not. Degree is defined as the number of explicit neighbors in the graph and valence as the actual number of neighbors of the atom. In graph models, hydrogens are often treated implicitly to reduce computational effort. Using degree and valence as features includes information about the number of implicit hydrogen bonded to an atom. Initial bond features include bond type, whether the bond is conjugated, and whether the bond is in a ring. The bond type is encoded by a one-hot vector with classes for single, double, triple, and aromatic bonds.

Algorithm 1: graph node update procedure

```
for  $\nu_i \in \mathcal{G}$  do  
    Get features  $\{h_{\nu_j}\}$  of neighboring nodes  $\{\nu_j\}$  ;  
    Update node features  $\{h_{\nu_i}\} \leftarrow f\left(\sum_j h_{\nu_j}\right)$  ;  
end
```

For a sample reaction, the initial features for each atom and bond in all reactants and reagents are collected using the RDKit cheminformatics package.[167] The model from Coley et al. uses a Weisfeiler-Lehman Network for the graph updates.[168] The Weisfeiler-Lehman algorithm follows Algorithm 1 where ν_i is the node for atom i , h_{ν_i} is the set of features for atom i at the current step, \mathcal{G} is the molecular graph which contains the set of nodes (atoms) and edges (bonds) in the molecule. $f(x)$ here is an ideally injective function. The process is repeated for k steps to iteratively update the atom node features. In a graph-convolution model, $f(x)$ is typically a NN to combine the neighbor features with the atoms features in an invariant way with respect to ordering of the neighbors.

Following Coley et al., the graph model takes the initial features for the neighboring atoms and the bonds which connect them. For each neighbor, the atom and bond features are concatenated into a single vector which is fed into a NN with a single hidden layer. The features of this hidden layer are then summed across neighbors of each atom to preserve invariance. Then, the sum is concatenated with the atom’s own features from the previous iteration in the graph, or from the initial features for the first iteration. Finally, this vector is input into another hidden layer which produces the updated atom features. The process is repeated for two iterations.

6.2.2 Predicting bond changes as graph operations

At this point, the model derives a set of "local" features for each atom. The features of each atom and its neighbor atoms and bonds are again collected, but each one is fed through a separate hidden layer. The element-wise product of those hidden features is taken for each neighbor atom and bond. Again, the neighbors are summed to preserve invariance at this point. An atom’s own hidden features are then multiplied element-wise with the neighbor features. This set of local features is used to represent what the graph has collected about atoms based on their local

environment.

After the local features have been collected, the model then uses these features to predict the probability of two atoms to change their current (non-)bond to any other bond order. Each reaction in the dataset contains labeled bond changes of two atoms and their new bond order in the product. The first portion of the model uses these labels to assign probabilities to perceived likely bond changes based on the reactants and reagents. The most probable bond changes are then used in another model to build candidate products which the model will rank as the most likely products.

For the model to predict whether two atoms are likely to change their bond, purely local information from the graph model is insufficient. Reactions depend on the reaction conditions including the solvent, catalysts, or other reagents involved in the reaction. The graph update process only allows for information to spread through bonded connections over a few atoms. Distant atoms or atoms in another molecule carry important information for predicting the probability of two other atoms to form a particular bond.

To remedy this, Coley et al. implement an attention NN to collect a set of "global" features for each atom. The global features are calculated by taking the local features for one atom and summing with the local features for every other atom across all atoms in the reactants and reagents. The pairwise sums are then fed through a single hidden layer followed by a sigmoid layer which produces a scalar for each atom in the system. The sigmoid layer ensures an output between zero and one. The scalar is used to weight the pairwise sums which were fed into the first layer of the attention NN, and then the result is summed across all atoms. This is taken as the global features of the atom.

The intent behind the attention mechanism is to allow the model to collect another set of features by deciding which other atoms in the reaction system it should consider when deciding the probability of two atoms to form a particular bond order. Because

the model is based on the net bond changes in a reaction, without the attention mechanism there would be no way for the model to consider information from the atoms on a catalyst in deciding which atoms are most likely to react.

Next, the local features and global features are separately summed for each pair of atoms in the molecule to represent features that will be used to predict the probability of that pair of atoms forming each bond order including a non-bond and aromatic bond. Each local atom pair and global atom pair feature vector are fed into separate single hidden layers and then summed to create a feature representation of the atom pair which contains the local and global information about both atoms. This feature vector is then fed through another layer which maps the features to a vector of five scores. The five scores correspond to probabilities of the atom pair to form a non-bond (i.e. to break an existing bond), single, double, triple, and aromatic bond. This score is then used to rank the most likely bond changes over the reaction system. Larger positive numbers have a high probability of forming the bond change and lower negative numbers have a low probability.

The loss function to train the model is a cross entropy loss over a sigmoid function for each of the five bond types. This type of loss function is used in multi-class classification tasks with classes which are not mutually exclusive. The learning target for each atom pair for each bond type is a binary number corresponding to whether or not that atom pair newly forms the bond type in the products of the reaction. If the bond exists in both the reactants and products, or the bond does not exist in both the reactants and products, the label will be zero. If the atom pair has a change in the bond type between them, the label will be one for the bond type the atom pair forms in the products. For all other bond types between that atom pair the label will be zero. The rationale is that the model should predict high probabilities for bonds in the products that are different than in the reactants.

6.2.3 Future implementations

The future work in this project consists of completing the graph model from Coley et al. After the model predicts the most probable bond changes, candidate products are formed by selecting the top k bond changes, and enumerating all possible products using up to five bond changes for each candidate. Each candidate is screened for invalid stoichiometry and only valid molecules are retained. A reaction representation is calculated by finding the difference in atom features between the reactants and product candidates using a similar graph network which are used to rank the most likely products.

Coley et al. use a softmax cross entropy loss over the candidates to rank the most probable candidate. This amounts to a classification task, as the few most probable candidates are typically considered, but there is no information about the yield of the reaction. If the model is tailored to a specific reaction and we wish to predict yields, then often the product is known so instead this portion of the model can be turned into a regression model of reaction yield.

Including calculated and measured properties into the model should improve performance. One consideration is that if properties are required to run the model, then they should be fast to evaluate. Otherwise, properties may be implemented by having the model learn the properties as a pre-training method to build features, and then fine-tuning the model for prediction of yields by incorporating those learned features.

6.3 Results

The model is trained on the same dataset used by many previous works.[27, 28, 30, 31] The work is in very early stages and as such the early focus is in replicating the work from Coley et al.[30] The progress so far has reached a point where I have replicated the results from the portion of the model which has been discussed here. In

TABLE 6.1

THE ACCURACY OF THE MOST PROBABLE BOND CHANGE
PREDICTIONS FROM COLEY ET AL.[30] AND THIS WORK.

Model	Top-12 [%]	Top-16 [%]	Top-20 [%]	Top-40 [%]	Top-80 [%]
Coley et al.	87.0	90.5	92.0	95.3	97.1
This work	87.0	89.5	91.0	94.8	97.0

Accuracy here is defined as all of the correct bond changes to form the products occurring in the top n most probable bond changes predicted by the model where n is any integer. If any of the correct bonds is missing from the set of the n most probable changes the model is considered to be wrong for that sample.

the work from Coley et al. after their model ranks the top bond change candidates, they form product candidates by enumerating combinations of the highest probability bond changes and applying the changes to the reactants. Unstable structures with violated valencies in the structure are removed from the candidates. The future work regarding this project is to finish replicating the later portion of the model published by Coley et al.

Because reproducing the full model from Coley et al. is incomplete, the current progress cannot be compared to much of the analysis in their manuscript.[30] However, a comparison of the accuracy can be made for the portion of the model which predicts the most likely bond changes. From the graph model implementation provided by the authors, they supply metrics for the accuracy of this portion of the model.[169] These metrics are compared to the results obtained in this work so far in Table 6.3. The accuracy of the models are well within the expected variance based on the non-determinism of training NN models.

6.4 Conclusions

In the immediate future, work on this project will consist of finishing to replicate the model from Coley et al.[30] Their graph network model was shown to achieve superior accuracy over other state-of-the-art models with an order of magnitude fewer parameters. The strength of graph representation is clear from this result. With the goal to make a regressive model of reaction yields, changes will need to be made to accommodate the incorporation of properties which experts have found correlate with reaction yields in other models. A model which combines the power of these two types of domain knowledge may be a powerful method for predicting reaction yields.

Since predicting yield is a regressive task instead of classification, the model needs to be more sensitive to the reaction conditions. This may require innovative ways of including more granular details about the reaction conditions. For example the temperature of the reaction can affect yield, but it is not clear how this information would be included in graph models. Similarly for including measured or calculated properties of the reactions, the way this is incorporated into the model will require some experimenting with the model parameters. Furthermore, significant efforts are required to generate a dataset with yields. High-throughput experimentation enables the generation of modest datasets, but analysis is still a rate-limiting step in the pipeline.

One of the biggest challenges likely to be faced during this work is a lack of significant amounts of data needed for NN models. The small size of datasets with reaction yield data will require including pre-training methods to encourage the model to learn what it can from datasets for related tasks. Aside from pre-training with the USPTO data, it may be possible to include pre-training from other standard datasets with calculated properties that may correlate with reaction yields.

CHAPTER 7

STOKES SHIFTS IN LEAD HALIDE PEROVSKITES

7.1 Introduction

Brennan and coworkers published a manuscript detailing an apparent Stokes shift in CsPbBr₃ perovskite nanocrystals.[170] These perovskite nanomaterials are promising candidates for next-generation photovoltaic and light-emitting applications due to their high photoluminescence yields, narrow emission line widths, and their size-dependent band gaps which are tunable over the visible range of the electromagnetic spectrum. Understanding the underlying electronic structure and dynamics is necessary for their implementation into modern devices.

The work by Bennnan et al. showed a nanocrystal size-dependence of the Stokes shift exhibited by these materials. The ensembles of nanocrystals studied ranged in size from $l \approx 4$ nm to 12 nm. Photoluminescence experiments showed a red-shift in the emission relative to the absorption band edge. The Stokes shifts measured ranged from ~ 30 meV for nanocrystals with $l \approx 12$ nm to ~ 100 meV for $l \approx 4$ nm. These results are shown in Figure 7.1.

To understand the origin of the Stokes shift, I collaborated with Brennan and coworkers to develop a theoretical model to rationalize their experimental findings. Historically, CdSe quantum dots were shown to exhibit a similar size-dependent Stokes shift, which was shown to originate from band edge excitonic fine structure.[172, 173] These works increased the understanding of the CdSe model system. Similar efforts to understand the Stokes shifts in CsPbBr₃ nanocrystals should provide insight into the design for modern devices.

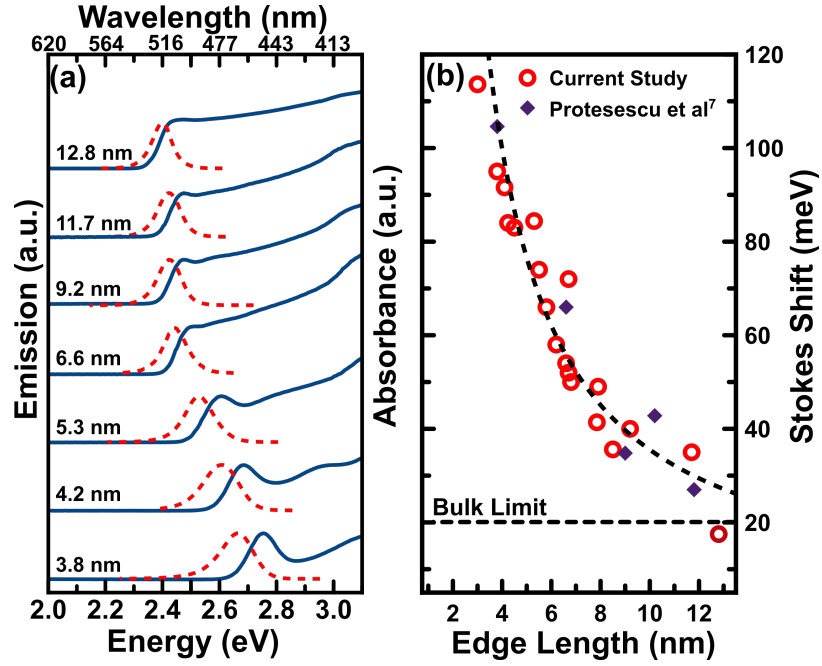


Figure 7.1. (a) Ensemble absorption (solid blue lines) and emission (dashed red lines) spectra from a small size series of CsPbBr₃ NCs dispersed in toluene. Above gap excitation ($E_{exc} = 3.543$ eV, $\lambda_{exc} = 350$ nm) used to acquire PL spectra. All absorption/emission spectral pairs offset for clarity. (b) Corresponding size-dependent Stokes shifts and those extracted from existing literature.[171]

To establish that the Stokes shift is intrinsic to the perovskite nanocrystal, Brennan et al. ran experiments to rule out extrinsic causes. Time-correlated single photon counting time-resolved emission spectroscopy (TCSPC-TRES) ruled out solvatochromism and other effects with slow degrees of freedom. Absorption and emission spectra were recorded in chloroform, hexane, and toluene which showed that Stokes shifts deviate by no more than 10 meV based on the solvent which means the effect is not due to the dielectric medium. Contributions from the residual size distribution of the nanocrystal ensembles was accounted for by measuring absorption/emission spectra when the nanocrystal ensembles were excited at their respective band edge and comparing to those obtained when exciting further to the blue in the spectrum. Residual size distribution was shown to contribute to the Stokes shift, though the effect only accounted for a small portion of the exhibited Stokes shift. Corrections brought the range of apparent intrinsic Stokes shifts from an upper-limit of ~ 100 meV to ~ 80 meV.

In this work, I collaborated with an experimental group to develop the theoretical model of the Stokes shifts for these perovskite nanomaterials. First-principles calculations were run for model systems to elucidate the electronic structure and determine if we could develop an understanding of the Stokes shifts based on the experimental results in combination with my theoretical calculations. The model developed in this work matches observed trends in the experimental data. Our model is based on first-principles calculations of multiple size supercell models of the perovskite. I show that the electronic structure is consistent across many different surface defect models.

7.2 Methods

7.2.1 Bulk material characterization

To determine if our model chemistry is appropriate for the material being studied, it is necessary to first ensure that the bulk system is treated well within our approximations. Simulating bulk materials requires the inclusion of k-space vectors sufficient for covering the first Brillouin zone up to the symmetries of the material. Our calculations for the nanocrystal model systems uses CP2K due to its efficiency. However, sampling k-space vectors was not implemented in CP2K as of the time this research was done. For the bulk materials, we used the plane-wave DFT code PWSCF implemented in the Quantum Espresso software suite.[174] Scalar relativistic and spin-orbit coupling interactions are included with a plane-wave cutoff of 500 eV. A $6 \times 6 \times 6$ Monkhorst-Pack grid was used for sampling the Brillouin zone.

The calculated optimized lattice parameters and band gaps for the three crystal morphologies of this material are reported in Table 7.2.1 along with experimental values from the literature. Calculated lattice parameters and band gaps agree well with experimental values. The band structure is plotted in Figure 7.3(a) for the cubic crystal morphology. The calculated band gap occurs at the R point [$k = (1/2, 1/2, 1/2)$] and has an estimated gap of 2.16 eV, which agrees well with experimental values.[177, 178]

7.2.2 Generation of defect models

Experimentally observed nanocrystals of CsPbBr_3 exhibit a cuboidal shape as shown by TEM results from my experimental collaborators in Figure 7.2, so we limited the models to this morphology. CsPbBr_3 has been shown to exist in three main crystalline lattice systems; an orthorhombic, cubic, and tetragonal structure. Experiments to describe the crystal lattice present in the nanocrystals of this ma-

TABLE 7.1
CALCULATED CRYSTALLOGRAPHIC PARAMETERS FOR BULK
CSPBBR₃.

Space group	Lattice parameters	Band gap (eV)
Cubic	$a = b = c = 5.87$ [5.84] (Å)	2.16 [2.39]
Pm-3m	$\alpha = \beta = \gamma = 90^\circ$	
Pseudocubic	$a = b = c = 5.85$ (Å)	2.39
P4mm	$\alpha = \beta = \gamma = 90^\circ$	
Orthorhombic	$a = 8.24$ [8.26] (Å)	
Pnma	$b = 11.74$ [11.78] (Å)	2.41 [2.25]
	$c = 8.20$ [8.25] (Å)	
	$\alpha = \beta = \gamma = 90^\circ$	

Experimental values are reported in square brackets.[175–179]

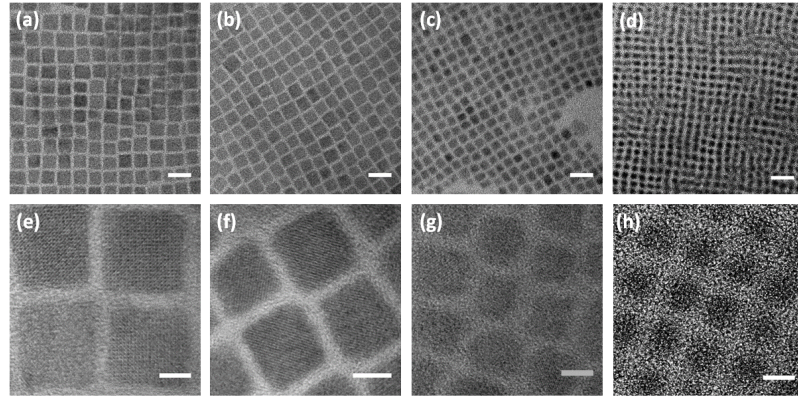


Figure 7.2. Representative low- [(a) $l = 12.8$ nm, (b) $l = 8.5$ nm, (c) $l = 6.8$ nm, (d) $l = 4.2$ nm] and high- [(e) $l = 12.8$ nm, (f) $l = 8.5$ nm, (g) $l = 6.8$ nm, (h) $l = 4.1$ nm] magnification TEM images of CsPbBr₃ NCs. Scale bars on low- and high-magnification images are 20 and 5 nm, respectively.

terial suggest that the apparent lattice reflects the orthorhombic or cubic structure more.[80, 180] We developed models of each crystal lattice type that show no apparent differences in the resulting electronic structure of the optimized structures.

Starting from the cubic (Pm-3m) lattice, the [100], [010], and [001] directions of the lattice are identical. Along any of the directions, the lattice exhibits two unique alternating layers; a first layer containing only Pb and Br atoms and a second layer containing Cs and Br atoms. These shall be referred to as Pb-Br and Cs-Br layers in this thesis. To build a model nanocrystal, we begin by building an $n \times n \times n$ supercell for $3 \leq n \leq 7$ by building a supercell one size larger than n , then cleaving one layer from either the three faces which terminate with a Pb-Br layer or those with a Cs-Br layer termination. This leaves a symmetric crystal structure but the overall charge of the model is still non-neutral.

Orthorhombic supercells are cut to identical stoichiometry and shape such that the only difference in the supercell model systems is the tilting of the octahedral geometry surrounding Pb in the crystal structure. By making models of both crystal morphologies, I show that the model systems exhibit a structure which becomes hard to distinguish between cubic or orthorhombic, but that the electronic structure is qualitatively similar for models derived from the cubic or orthorhombic morphologies.

Important properties of these materials include a negligible dipole moment and an overall neutral charge nanocrystal. One method to achieve this is to passivate the surface with a model ligands similar to those observed experimentally. However, CsPbBr₃ possesses a defect-tolerant electronic structure similar to other lead-based semiconductors.[181–183] The top of the valence band arises from the antibonding interaction of the Pb(6s)-Br(4p) orbitals while the bottom of the conduction band corresponds to the Pb(6p)-Br(4p) antibonding interaction. Thus cation vacancies (Cs or Pb) would cause valence-band states, but anion vacancies (Br) would only appear as resonances inside the conduction band if the conduction band minimum

falls well below the energy of the Pb(6p) orbitals. For Pb, as opposed to Ge and Sn materials, there is an increased spin-orbit interaction that increases the width of the Pb(6p) conduction band.[183] This analysis is supported by the high photoluminescence quantum yields observed in these materials, which indicates a trap-free band gap. As such, the models are based on surface defects to remedy the non-neutral charge.

To keep a negligible dipole moment, defects must be made to lattice positions symmetrically to retain the inversion center about the Cartesian center of the nanocrystal. Due to this restriction, and the stoichiometry of the perovskite, it is only possible to terminate supercells with even n unit cells with Pb-Br layers and odd n with Cs-Br layers. The models developed in this work all follow this restriction.

Additional vacancies beyond what is necessary to neutralize the charge of the model system are possible. A thorough study of the free energy differences of surface defects is needed to fully determine the composition of the surfaces, but in this work I rely on properties which are robust with respect to the number of positions of surface defects. If the model systems exhibit certain features common to their electronic structure regardless of the surface defects, then it is likely that those features are independent of surface defects. With this in mind, I studied multiple models of each size by varying the number and position of surface defects.

7.2.3 Determination of electronic structure

All calculations of nanocrystal model systems performed in this work were done with CP2K 4.1[184–192] using a Gaussian and Augmented Plane Wave method (GAPW). Each model system is given 15 Å of vacuum surrounding to prevent effects from periodic images of the system. The density functionals used include PBE and HSE06.[129, 193] PBE is used for an initial geometry optimization of the model systems to reduce the computational effort needed with a hybrid functional such

as HSE06. The hybrid functional, however, is crucial to obtaining an accurate description of the electronic structure of these materials. After initial model systems have been relaxed by PBE, the geometry is reoptimized with HSE06. Calculations employed a double- ζ quality basis and pseudopotentials including scalar relativistic effects.[194–197]

Once model system geometries have reached convergence, a density of states (DOS) and partial density of states (PDOS) analysis is performed to determine the composition of the molecular orbitals formed in the model and the density of electronic states at and near the band edges. I analyzed the PDOS for expected character of atomic orbitals contributing to each molecular orbital and comparing with previous studies. Furthermore, each model should exhibit a band gap which roughly matches what is expected based on experimentally measured trends.

Finally, I observed the calculated molecular orbitals from the electronic structure to determine what characteristics of the orbitals could guide our intuition to explain the cause of a size-dependent Stokes shift. The bulk material exhibits a Stokes shift of ~ 20 meV, which is attributed to lattice-induced carrier stabilization and remains distinct from the size-dependent shifts observed in nanocrystals of this material. I also invoke an adiabatic assumption that electron-hole polaronic lattice relaxation is size-independent.[198, 199]

To observe trends in the electronic structure of these materials with respect to nanocrystal size, I constructed models with edge lengths of $l = 2.05, 2.64, 3.23, 3.82$, and 4.40 nm. Smaller models begin to exhibit less crystalline character as the lattice positions deviate more drastically in smaller models. Larger models quickly become computationally intractable. I was only able to complete a single model system with an edge length of 4.40 nm due to the amount of computational effort needed. Larger model systems would be desirable so that we can match the size distributions experimentalists are able to synthesize, however, our largest model is approaching

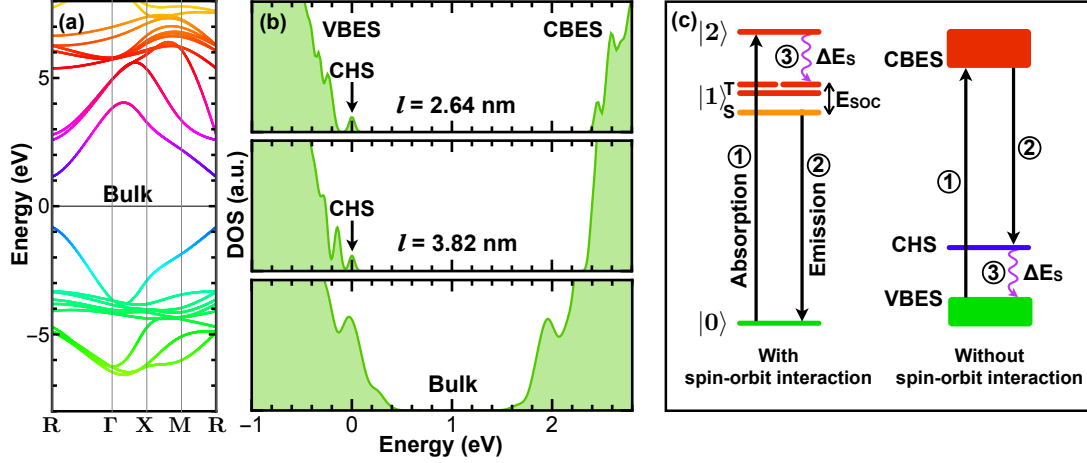


Figure 7.3. (a) Electronic band structure of bulk, cubic CsPbBr₃ calculated with the spin-orbit coupling interaction. (b) DOS for two NC sizes and the bulk material. The CHS shifts towards the VB edge as the NC size increases, eventually becoming indistinguishable from the bulk VB edge. (c) Model Jablonski diagrams show the fine structure resulting from SOC.

the smallest size distributions made by Brennan et al..

7.3 Results

The DOS for two sizes of the nanocrystal models are plotted in Figure 7.3(b) narrowed in on the band gap region, along with the DOS for the bulk material. The trend observed shows a single electronic state which becomes separated from the valence band edge (VBE) in the nanocrystal materials. Smaller nanocrystal model systems typically exhibit larger separation of this state from the VBE. Figure 7.4 shows the molecular orbitals from the optimized structures for four model system sizes. The two orbitals shown for each size are the two highest occupied valence states, the valence band edge state (VBES) along with the low-lying gap state. The gap state is a nodeless, cuboidal shaped orbital which delocalizes over the entire particle. This state, referred to as the confined hole state (CHS), exists in both cubic and orthorhombic models.

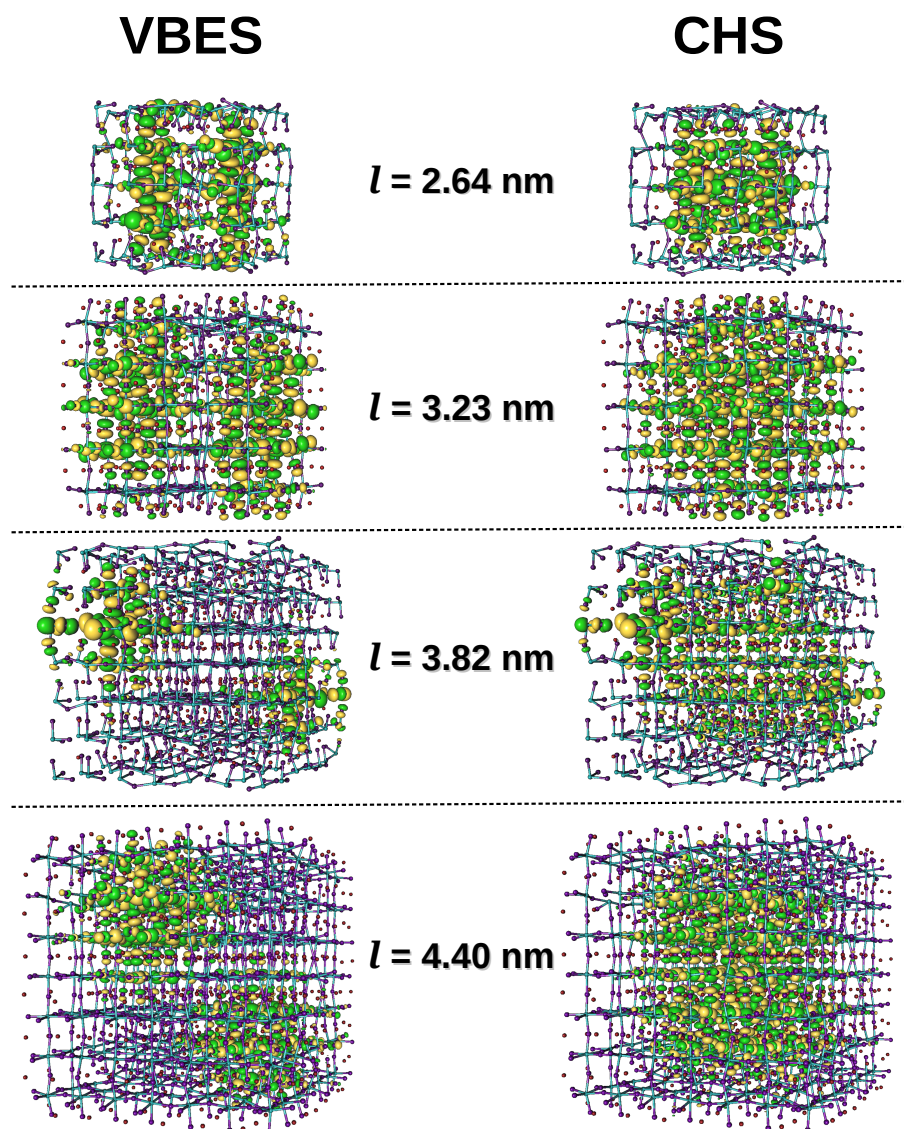


Figure 7.4. Molecular orbitals for models with $l = 2.64$, 3.23 , 3.82 , and 4.40 nm. The VBES always shows a nodal plane cutting through the NC and remains near the surface. The CHS is always highly delocalized across the NC core and gradually becomes more diffuse as l increases.

In our computed model systems, the gap between the CHS and the VBES is on the order of tens to hundreds of meV. Because of the apparent size trend of the energy difference between the CHS and VBES, and because the order of magnitude of this difference is the same as the experimentally observed Stokes shifts, we hypothesized that this electronic state could account for the Stokes shifts. This would indicate that the absorbing and emitting states are distinct from one another. Furthermore, this state does not exist in the bulk material. Our results also show that the observed state is robust with respect to size and surface defects of the nanocrystals.

Figure 7.5(a) shows the energy differences between the CHS, VBES, and the conduction band edge state (CBES) for the five sizes of model systems. Energies have been shifted so that the CHS is always at zero. The energy difference between CHS and VBES decreases rapidly with larger size models, but shows an apparent sloping off of the rate at which this gap closes (i.e. the second order derivative of the energy gap between CHS and VBES with respect to model size is negative). Qualitatively, this tracks with what might be expected if this state indeed is responsible for the Stokes shift. Experimentally measured Stokes shifts in Figure 7.1 show a similar behavior, where the rate at which the Stokes shift decreases with size of the ensemble distributions decays for larger samples.

These model systems are smaller than our experimental collaborators are able to synthesize. The largest model system I made has an edge length of 4.40 nm. The smallest average edge length from my experimental collaborators ensembles was $l = 3.0$ nm, with most ensembles in the range of 3.8 nm or larger. Nonetheless, my experimental collaborators and myself feel there is enough overlap between the experimental ensembles and theoretical model systems to provide insights into the cause of the Stokes shift. Experimental values for the ensembles are reported in Table 7.2. Theoretical values for the model systems are provided in Table 7.3. The Stokes shift values from both experimental and theoretical work are plotted in Figure 7.5(b) along

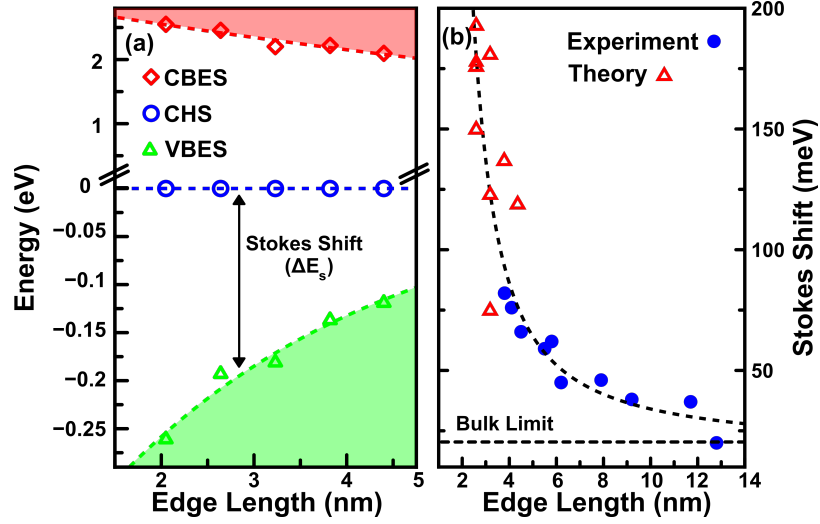


Figure 7.5. (a) Size-dependent NC CBES and VBES energies plotted relative to CHS energies, which are set to zero. State energies calculated at the Γ -point. Dashed lines are corresponding interpolation fits. (b) Experimental band edge excitation-derived ΔE_s values (closed blue circles) plotted against theoretical Stokes shifts (open red triangles).

with a line of best fit to the combined sets. The values from the theoretical models appear to exhibit a reasonable trend with size for Stokes shifts to experimentally observed values.

The size of our theoretical models makes including spin-orbit coupling interactions computationally too expensive. Spin-orbit interactions can contribute to exciton fine-structure, which can cause a Stokes shift in emission. To account for this, I modeled the spin-orbit interaction between $\text{CHS} \rightarrow \text{CBES}$ spin states; the singlet spin $m_s=0$ ($|S_{m_s=0}\rangle$), triplet $m_s=0$ ($|T_{m_s=0}\rangle$), and triplets $m_s=\pm 1$ ($|T_{m_s=\pm 1}\rangle$) were calculated using a Breit-Pauli Hamiltonian[200–202] to approximate the spin-orbit coupling between relevant spin states. The form of the Hamiltonian is

$$\hat{H}_{SO} = -\frac{\alpha_0^2}{2} \sum_{i,A} \frac{Z_A}{r_{iA}^3} (\mathbf{r}_{iA} \times \mathbf{p}_i) \cdot \mathbf{s}_i \quad (7.1)$$

TABLE 7.2

ABSORPTION, EMISSION, AND STOKES SHIFT DATA FOR
CSPBBR₃ NANOCRYSTAL ENSEMBLES.

Edge Length (nm)	Stokes Shift (meV)	Band Edge Absorption (eV)	Emission Max (eV)
3.0	114	2.853	2.739
3.8	95 (82)	2.753	2.658
4.1	93 (76)	2.711	2.618
4.2	85	2.703	2.618
4.5	83 (66)	2.670	2.587
5.3	84	2.610	2.526
5.5	74 (59)	2.599	2.525
5.8	66 (62)	2.586	2.520
6.2	58 (45)	2.554	2.496
6.6	54	2.507	2.453
6.7	52	2.501	2.449
6.8	50	2.530	2.480
7.9	49 (46)	2.476	2.427
7.9	41	2.481	2.440
8.5	35	2.471	2.436
9.2	40 (38)	2.462	2.422
11.7	35 (37)	2.455	2.420
12.8	17 (20)	2.415	2.398

Stokes shifts in parentheses are taken from emission spectra with band edge E_{exc} . All other data are for emission spectra with $E_{exc} = 3.543$ eV.

TABLE 7.3
STOKES SHIFT AND BAND GAP DATA FOR THEORETICAL
MODEL SYSTEMS.

Label	Edge Length (nm)	Stokes Shift (meV)	Band Gap (eV)	Surface Defect (%)
NC-3-1	2.05	261	2.665	7.8
NC-4-1	2.64	176	2.517	58.6
NC-4-2	2.64	150	2.371	7.6
NC-4-3	2.64	193	2.503	5.5
NC-4-4	2.64	176	2.213	58.6
NC-5-1	3.23	123	2.044	37.1
NC-5-2	3.23	181	2.237	5.3
NC-5-3	3.23	75	1.914	31.8
NC-6-1	3.82	137	2.382	55.7
NC-7-1	4.40	119	2.070	3.7

Naming convention follows NC- n - m where n correspond to the number of unit cells used to construct the supercell model as discussed in Section 7.2.2 and m distinguishes models of the same size. Surface defect refers to the percentage of atom vacancies relative to a supercell model with no surface vacancies.

where i indexes over electrons, A indexes over nuclei, $\alpha_0 = 137.037^{-1}$ is the fine structure constant. Z_A is the bare positive charge on nucleus A . s_i represents the spin of electron i . The term $\mathbf{r}_{iA} \times \mathbf{p}_i$ is the angular momentum operator of electron i calculated with respect to nucleus A at position R_A , with r_{iA} being the distance between them. The resulting Hamiltonian is diagonalized to obtain spin-orbit coupled fine structure states. The four resulting fine-structure states are separated by as much as 10.3 meV for a model system with an edge length of 2.64 nm, and as little as 0.13 meV for an edge length of 3.82 nm, so we did not consider this interaction significant enough to result in the size of Stokes shifts observed.

Summarizing this with the observations of the CHS thus far discussed, the model Jablonski diagram of the spin-orbit interaction and the ordering of the fine-structure splitting is shown in Figure 7.3(c). Furthermore, we show the model of our rationalization for the Stokes shift. Our model suggests that the CHS may be dark in absorption relative to the states at the top of the valence band. Excitation occurs primarily from the VBE to the conduction band edge (CBE) with a non-radiative relaxation of the hole from the CHS to the VBES, followed by relaxation of the electron from CBES to CHS.

The absorption strength of the CHS→CBES transition was measured as the transition dipole moment between these two states. This is compared to the absorption strength of the states for all transitions from VBES→CBES which fall within the range of energies for the laser used in the photoluminescence experiments. No trend with size is observed, but the absorption strength of the VBES→CBES ranges from a factor of 10 to 50 higher than the strength of the CHS→CBES transition. Thus we suggest that absorption of the CHS is relatively dark compared to the VBES.

7.4 Conclusions

This work establishes a model for the Stokes shift observed in CsPbBr_3 . Experimental photoluminescence spectra showed a Stokes shift for nanocrystal ensembles which exhibited Stokes shifts in the range of 82 to 20 meV. Experimental results eliminated the possibility of extrinsic factors causing the Stokes shift. Our theoretical model systems supported evidence of the CHS, a nodeless cuboidal electronic state which is a low lying gap state that only exists in nanocrystals. The state is shown to be relatively dark in absorption compared to the VBES. We further show results which suggest that the Stokes shift cannot be explained by exciton fine-structure splitting. Furthermore, because Cs does not significantly contribute to the electronic states near the band gap, these results may be applicable for similar perovskites which use alternative cations. Other halides can be substituted as well, which may show similar electronic structure in these materials.

APPENDIX A

TERMINOLOGY

TABLE A.1

COMMONLY USED ABBREVIATIONS

Abbreviation	Definition
ML	machine learning
NN	neural network
HDNNP	high-dimensional neural network potential
ACSF	atom-centered symmetry functions
GAN	generative adversarial network
RNN	recurrent neural network
CNN	convolutional neural network
ELU	exponential linear unit (activation function)
KRR	kernel ridge regression
MAE	mean absolute error
MSE	mean signed error
RMSE	root mean square error
MBE	many-body expansion
MD	molecular dynamics
AIMD	ab initio molecular dynamics

TABLE A.1 (CONTINUED)

Abbreviation	Definition
MetaMD	metadynamics
NMS	normal mode sampling
DFT	density functional theory
MP2	Møller-Plesset perturbation theory method at second order
BSSE	basis set superposition error
CM	Coulomb matrix
PCA	principal component analysis
IR	infrared
VBE	valence band edge
VBES	valence band edge state
CBE	conduction band edge
CBES	conduction band edge state
CHS	confined hole state
DOS	density of states
PDOS	projected density of states
fs	femtosecond
ps	picosecond
ns	nanosecond

BIBLIOGRAPHY

- (1) Krizhevsky, A.; Sutskever, I.; Hinton, G. E. In *Advances in Neural Information Processing Systems*, 2012, pp 1097–1105.
- (2) Large Scale Visual Recognition Challenge 2012. <http://image-net.org/challenges/LSVRC/2012/results.html>.
- (3) Csáji, B. C. et al. *Faculty of Sciences, Eötvös Loránd University, Hungary* **2001**, 24, 7.
- (4) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. In *Advances in Neural Information Processing Systems*, 2014, pp 2672–2680.
- (5) Behler, J.; Parrinello, M. *Physical Review Letters* **2007**, 98, 146401.
- (6) Rupp, M.; Tkatchenko, A.; Müller, K.-R.; Von Lilienfeld, O. A. *Physical Review Letters* **2012**, 108, 058301.
- (7) Hansen, K.; Biegler, F.; Ramakrishnan, R.; Pronobis, W.; Von Lilienfeld, O. A.; Müller, K.-R.; Tkatchenko, A. *Journal of Physical Chemistry Letters* **2015**, 6, 2326–2331.
- (8) Montavon, G.; Rupp, M.; Gobre, V.; Vazquez-Mayagoitia, A.; Hansen, K.; Tkatchenko, A.; Müller, K.-R.; Von Lilienfeld, O. A. *New Journal of Physics* **2013**, 15, 095003.
- (9) Hansen, K.; Montavon, G.; Biegler, F.; Fazli, S.; Rupp, M.; Scheffler, M.; Von Lilienfeld, O. A.; Tkatchenko, A.; Müller, K.-R. *Journal of Chemical Theory and Computation* **2013**, 9, 3404–3419.
- (10) Rupp, M.; Ramakrishnan, R.; Von Lilienfeld, O. A. *The Journal of Physical Chemistry Letters* **2015**, 6, 3309–3313.
- (11) Yang, K.; Swanson, K.; Jin, W.; Coley, C.; Eiden, P.; Gao, H.; Guzman-Perez, A.; Hopper, T.; Kelley, B.; Mathea, M.; Palmer, A.; Settels, V.; Jaakkola, T.; Jensen, K.; Barzilay, R. *Journal of Chemical Information and Modeling* **2019**, 59, 3370–3388.
- (12) Isayev, O.; Fourches, D.; Muratov, E. N.; Oses, C.; Rasch, K.; Tropsha, A.; Curtarolo, S. *Chemistry of Materials* **2015**, 27, 735–743.

- (13) Isayev, O.; Oses, C.; Toher, C.; Gossett, E.; Curtarolo, S.; Tropsha, A. *Nature Communications* **2017**, *8*, 15679.
- (14) Schütt, K.; Glawe, H.; Brockherde, F.; Sanna, A.; Müller, K.; Gross, E. *Physical Review B* **2014**, *89*, 205118.
- (15) Pilania, G.; Wang, C.; Jiang, X.; Rajasekaran, S.; Ramprasad, R. *Scientific Reports* **2013**, *3*.
- (16) Ouyang, R.; Curtarolo, S.; Ahmetcik, E.; Scheffler, M.; Ghiringhelli, L. M. *arXiv preprint arXiv:1710.03319* **2017**.
- (17) Ghiringhelli, L. M.; Vybiral, J.; Ahmetcik, E.; Ouyang, R.; Levchenko, S. V.; Draxl, C.; Scheffler, M. *New Journal of Physics* **2017**, *19*, 023017.
- (18) Jinnouchi, R.; Asahi, R. *Journal of Physical Chemistry Letters* **2017**, *8*, 4279–4283.
- (19) Yao, K.; Parkhill, J. *Journal of Chemical Theory and Computation* **2016**, *12*, 1139–1147.
- (20) Snyder, J. C.; Rupp, M.; Hansen, K.; Blooston, L.; Müller, K.-R.; Burke, K. *The Journal of Chemical Physics* **2013**, *139*, 224104.
- (21) Brockherde, F.; Vogt, L.; Li, L.; Tuckerman, M. E.; Burke, K.; Müller, K.-R. *Nature Communications* **2017**, *8*, 872.
- (22) Snyder, J. C.; Rupp, M.; Hansen, K.; Müller, K.-R.; Burke, K. *Physical Review Letters* **2012**, *108*, 253002.
- (23) Li, L.; Snyder, J. C.; Pelaschier, I. M.; Huang, J.; Niranjana, U.-N.; Duncan, P.; Rupp, M.; Müller, K.-R.; Burke, K. *International Journal of Quantum Chemistry* **2016**, *116*, 819–833.
- (24) Li, L.; Baker, T. E.; White, S. R.; Burke, K., et al. *Physical Review B* **2016**, *94*, 245129.
- (25) Fracchia, F.; Del Frate, G.; Mancini, G.; Rocchia, W.; Barone, V. *Journal of Chemical Theory and Computation* **2018**, *14*, 255–273.
- (26) Li, Y.; Li, H.; Pickard IV, F. C.; Narayanan, B.; Sen, F. G.; Chan, M. K.; Sankaranarayanan, S. K.; Brooks, B. R.; Roux, B. *Journal of Chemical Theory and Computation* **2017**, *13*, 4492–4503.
- (27) Nam, J.; Kim, J. *arXiv preprint arXiv:1612.09529* **2016**.
- (28) Schwaller, P.; Gaudin, T.; Lanyi, D.; Bekas, C.; Laino, T. *Chemical Science* **2018**, *9*, 6091–6098.

- (29) Schwaller, P.; Petraglia, R.; Zullo, V.; Nair, V. H.; Haeuselmann, R. A.; Pisoni, R.; Bekas, C.; Iuliano, A.; Laino, T. *arXiv preprint arXiv:1910.08036* **2019**.
- (30) Coley, C. W.; Jin, W.; Rogers, L.; Jamison, T. F.; Jaakkola, T. S.; Green, W. H.; Barzilay, R.; Jensen, K. F. *Chemical Science* **2019**, *10*, 370–377.
- (31) Jin, W.; Coley, C.; Barzilay, R.; Jaakkola, T. In *Advances in Neural Information Processing Systems*, 2017, pp 2607–2616.
- (32) Feng, F.; Lai, L.; Pei, J. *Frontiers in Chemistry* **2018**, *6*, 199.
- (33) Liu, B.; Ramsundar, B.; Kawthekar, P.; Shi, J.; Gomes, J.; Nguyen, Q. L.; Ho, S.; Sloane, J.; Wender, P.; Pande, V. *ACS Central Science* **2017**, *3*, 1103–1113.
- (34) Coley, C. W.; Green, W. H.; Jensen, K. F. *Accounts of Chemical Research* **2018**, *51*, 1281–1289.
- (35) Segler, M. H.; Waller, M. P. *Chemistry—A European Journal* **2017**, *23*, 5966–5971.
- (36) Segler, M. H. S.; Preuss, M.; Waller, M. P. *Nature* **2017**, *555*, 604–610.
- (37) Granda, J. M.; Donina, L.; Dragone, V.; Long, D.-L.; Cronin, L. *Nature* **2018**, *559*, 377–381.
- (38) Zhavoronkov, A. et al. *Nature Biotechnology* **2019**, *37*, 1038–1040.
- (39) Stokes, J. M. et al. *Cell* **2020**, *180*, 688–702.e13.
- (40) Popova, M.; Isayev, O.; Tropsha, A. *Science Advances* **2017**, *4*, eaap7885.
- (41) Popova, M.; Shvets, M.; Oliva, J.; Isayev, O. *arXiv preprint arXiv:1905.13372* **2019**.
- (42) Noé, F.; Olsson, S.; Köhler, J.; Wu, H. *Science (New York, N.Y.)* **2019**, *365*, eaaw1147.
- (43) Ribeiro, J. M. L.; Bravo, P.; Wang, Y.; Tiwary, P. *The Journal of Chemical Physics* **2018**, *149*, 072301.
- (44) Senior, A. W. et al. *Nature* **2020**, *577*, 706–710.
- (45) Noé, F.; Fabritiis, G. D.; Clementi, C. *Current Opinion in Structural Biology* **2020**, *60*, 77–84.
- (46) Behler, J. *Physical Chemistry Chemical Physics* **2011**, *13*, 17930–17955.
- (47) Behler, J. *The Journal of Chemical Physics* **2011**, *134*, 074106.

- (48) Behler, J. *The Journal of Chemical Physics* **2016**, *145*, 170901.
- (49) Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K. R.; Tkatchenko, A. *Nature Communications* **2016**, *8*, 13890.
- (50) Chmiela, S.; Tkatchenko, A.; Sauceda, H. E.; Poltavsky, I.; Schütt, K. T.; Müller, K.-R. *Science Advances* **2017**, *3*, e1603015.
- (51) Chmiela, S.; Sauceda, H. E.; Müller, K.-R.; Tkatchenko, A. *Nature Communications* **2018**, *9*, 3887.
- (52) Sauceda, H. E.; Chmiela, S.; Poltavsky, I.; Müller, K.-R.; Tkatchenko, A. *The Journal of Chemical Physics* **2019**, *150*, 114102.
- (53) Chmiela, S.; Sauceda, H. E.; Poltavsky, I.; Müller, K.-R.; Tkatchenko, A. *Computer Physics Communications* **2018**, *240*, 38–45.
- (54) Unke, O. T.; Meuwly, M. *Journal of Chemical Theory and Computation* **2019**, *15*, 3678–3693.
- (55) Schütt, K.; Kindermans, P.-J.; Felix, H. E. S.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. In *Advances in Neural Information Processing Systems*, 2017, pp 991–1001.
- (56) Schütt, K. T.; Sauceda, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. *The Journal of Chemical Physics* **2018**, *148*, 241722.
- (57) Yao, K.; Herr, J. E.; Toth, D. W.; Mckintyre, R.; Parkhill, J. *Chemical Science* **2018**, *9*, 2261–2269.
- (58) Smith, J. S.; Isayev, O.; Roitberg, A. E. *Chemical Science* **2017**, *8*, 3192–3203.
- (59) Herr, J. E.; Koh, K.; Yao, K.; Parkhill, J. *The Journal of Chemical Physics* **2019**, *151*, 084103.
- (60) Smith, J. S.; Nebgen, B.; Lubbers, N.; Isayev, O.; Roitberg, A. E. *The Journal of Chemical Physics* **2018**, *148*, 241733.
- (61) Smith, J. S.; Nebgen, B. T.; Zubatyuk, R.; Lubbers, N.; Devereux, C.; Barros, K.; Tretiak, S.; Isayev, O.; Roitberg, A. E. *Nature Communications* **2019**, *10*, 1–8.
- (62) Lubbers, N.; Smith, J. S.; Barros, K. *The Journal of Chemical Physics* **2017**, *148*, 241715.
- (63) Suwa, H.; Smith, J. S.; Lubbers, N.; Batista, C. D.; Chern, G.-W.; Barros, K. *Physical Review B* **2018**, *99*, 161107.

- (64) Smith, J. S.; Roitberg, A. E.; Isayev, O. *ACS Medicinal Chemistry Letters* **2018**, *9*, 1065–1069.
- (65) Behler, J. *Angewandte Chemie International Edition* **2017**, *56*, 12828–12840.
- (66) Behler, J.; Martoňák, R.; Donadio, D.; Parrinello, M. *Physical Review Letters* **2008**, *100*, 185501.
- (67) Gastegger, M.; Marquetand, P. *Journal of Chemical Theory and Computation* **2015**, *11*, 2187–2198.
- (68) Gastegger, M.; Behler, J.; Marquetand, P. *Chemical Science* **2017**, *8*, 6924–6935.
- (69) Bartók, A. P.; Kondor, R.; Csányi, G. *Physical Review B* **2012**, *87*, 184115.
- (70) Herr, J. E.; Yao, K.; McIntyre, R.; Toth, D. W.; Parkhill, J. *The Journal of Chemical Physics* **2018**, *148*, 241710.
- (71) Smith, J. S.; Isayev, O.; Roitberg, A. E. *Scientific Data* **2017**, *4*, 170193.
- (72) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; Von Lilienfeld, O. A. *Scientific Data* **2014**, *1*, 140022.
- (73) Vandermause, J.; Torrisi, S. B.; Batzner, S.; Kolpak, A. M.; Kozinsky, B. *arXiv preprint arXiv:1904.02042* **2019**.
- (74) Deringer, V. L.; Csányi, G. *Physical Review B* **2016**, *95*, 094203.
- (75) Deringer, V. L.; Pickard, C. J.; Csányi, G. *Physical Review Letters* **2017**, *120*, 156001.
- (76) Pozdnyakov, S. N.; Willatt, M. J.; Bartók, A. P.; Ortner, C.; Csányi, G.; Ceriotti, M. *arXiv preprint arXiv:2001.11696* **2020**.
- (77) Kondor, R. *arXiv preprint arXiv:1803.01588* **2018**.
- (78) Thomas, N.; Smidt, T.; Kearnes, S.; Yang, L.; Li, L.; Kohlhoff, K.; Riley, P. *arXiv preprint arXiv:1802.08219* **2018**.
- (79) Yao, K.; Herr, J. E.; Parkhill, J. *The Journal of Chemical Physics* **2016**, *146*, 014106.
- (80) Brennan, M. C.; Herr, J. E.; Nguyen-Beck, T. S.; Zinna, J.; Draguta, S.; Rouvimov, S.; Parkhill, J.; Kuno, M. *Journal of the American Chemical Society* **2017**, *139*, 12201–12208.
- (81) Rupp, M.; Tkatchenko, A.; Müller, K.-R.; Von Lilienfeld, O. A. *Physical Review Letters* **2012**, *108*, 058301.

- (82) Raghavachari, K.; Saha, A. *Chemical Reviews* **2015**, *115*, 5643–5677.
- (83) Kim, B.-s.; Xu, S.; Savarese, S. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.
- (84) Song, S.; Xiao, J. In *Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part VI*, Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, 2014, pp 634–651.
- (85) Xantheas, S. S. *The Journal of Chemical Physics* **1994**, *100*, 7523–7534.
- (86) Kestner, N. R.; Combariza, J. E. *Reviews in Computational Chemistry* **1999**, *13*, 99–126.
- (87) Boys, S. F.; Bernardi, F. d. *Molecular Physics* **1970**, *19*, 553–566.
- (88) Ouyang, J. F.; Cvitkovic, M. W.; Bettens, R. P. *Journal of Chemical Theory and Computation* **2014**, *10*, 3699–3707.
- (89) Hermann, A.; Krawczyk, R. P.; Lein, M.; Schwerdtfeger, P.; Hamilton, I. P.; Stewart, J. J. *Physical Review A* **2007**, *76*, 013202.
- (90) Góra, U.; Podeszwa, R.; Cencek, W.; Szalewicz, K. *The Journal of Chemical Physics* **2011**, *135*, 224102.
- (91) Cerutti, D.; Cheatham III, T.; Darden, T.; Duke, R.; Giese, T.; Gohlke, H.; Goetz, A.; Homeyer, N.; Izadi, S.; Janowski, P., et al. Amber 2016., 2016.
- (92) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. *Journal of Computational Chemistry* **2004**, *25*, 1157–1174.
- (93) Shao, Y.; Gan, Z.; Epifanovsky, E.; Gilbert, A. T.; Wormit, M.; Kussmann, J.; Lange, A. W.; Behn, A.; Deng, J.; Feng, X., et al. *Molecular Physics* **2015**, *113*, 184–215.
- (94) Kazachenko, S.; Bulusu, S.; Thakkar, A. J. *The Journal of Chemical Physics* **2013**, *138*, 224303.
- (95) Antipov, G.; Baccouche, M.; Dugelay, J.-L. In *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp 2089–2093.
- (96) Goodfellow, I. *arXiv preprint arXiv:1701.00160* **2016**.
- (97) Laio, A.; Parrinello, M. *Proceedings of the National Academy of Sciences* **2002**, *99*, 12562–12566.
- (98) Bernardi, R. C.; Melo, M. C.; Schulten, K. *Biochimica et Biophysica Acta (BBA)-General Subjects* **2015**, *1850*, 872–877.

- (99) Dickson, A.; Dinner, A. R. *Annual Review of Physical Chemistry* **2010**, *61*, 441–459.
- (100) Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K. R.; Tkatchenko, A. *Nature Communications* **2017**, *8*, 1–8.
- (101) Grisafi, A.; Wilkins, D. M.; Csányi, G.; Ceriotti, M. *Physical Review Letters* **2018**, *120*, 036002.
- (102) Gastegger, M.; Kauffmann, C.; Behler, J.; Marquetand, P. *The Journal of Chemical Physics* **2016**, *144*, 194110.
- (103) Han, J.; Zhang, L.; Car, R., et al. *arXiv preprint arXiv:1707.01478* **2017**.
- (104) Mitchell, B. E.; Jurs, P. C. *Journal of Chemical Information and Computer Sciences* **1997**, *37*, 538–547.
- (105) Andersen, H. C. *The Journal of Chemical Physics* **1980**, *72*, 2384–2393.
- (106) TensorMol. <https://github.com/jeherr/tensormol>.
- (107) Chai, J.-D.; Head-Gordon, M. *Physical Chemistry Chemical Physics* **2008**, *10*, 6615–6620.
- (108) Braun, E.; Gilmer, J.; Mayes, H. B.; Mobley, D. L.; Monroe, J. I.; Prasad, S.; Zuckerman, D. M. *Living Journal of Computational Molecular Science* **2019**, *1*.
- (109) Tersoff, J. *Physical Review B* **1989**, *39*, 5566.
- (110) Cisneros, G. A.; Karttunen, M.; Ren, P.; Sagui, C. *Chemical Reviews* **2014**, *114*, 779–814.
- (111) Sagui, C.; Darden, T. A. *Annual Review of Biophysics and Biomolecular Structure* **1999**, *28*, 155–179.
- (112) Fennell, C. J.; Gezelter, J. D. *The Journal of Chemical Physics* **2006**, *124*, 234104.
- (113) Wolf, D.; Keglinski, P.; Phillpot, S.; Eggebrecht, J. *The Journal of Chemical Physics* **1999**, *110*, 8254–8282.
- (114) Grimme, S. *Journal of Computational Chemistry* **2006**, *27*, 1787–1799.
- (115) Martín Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems., Software available from tensorflow.org, 2015.
- (116) Chempider. <https://www.chemspider.com/>.

- (117) Schütt, K. T.; Saucedo, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. *The Journal of Chemical Physics* **2017**, *148*, 241722.
- (118) Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. *arXiv preprint arXiv:1810.04805* **2018**.
- (119) Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. *OpenAI Blog* **2019**, *1*, 9.
- (120) Gastegger, M.; Schwiedrzik, L.; Bittermann, M.; Berzsényi, F.; Marquetand, P. *The Journal of Chemical Physics* **2018**, *148*, 241709.
- (121) Faber, F. A.; Hutchison, L.; Huang, B.; Gilmer, J.; Schoenholz, S. S.; Dahl, G. E.; Vinyals, O.; Kearnes, S.; Riley, P. F.; Von Lilienfeld, O. A. *Journal of Chemical Theory and Computation* **2017**, *13*, 5255–5264.
- (122) Faber, F. A.; Christensen, A. S.; Huang, B.; Von Lilienfeld, O. A. *The Journal of Chemical Physics* **2018**, *148*, 241717.
- (123) De, S.; Bartók, A. P.; Csányi, G.; Ceriotti, M. *Physical Chemistry Chemical Physics* **2016**, *18*, 13754–13769.
- (124) Bartók, A. P.; De, S.; Poelking, C.; Bernstein, N.; Kermode, J. R.; Csányi, G.; Ceriotti, M. *Science Advances* **2017**, *3*, e1701816.
- (125) Zhou, Q.; Tang, P.; Liu, S.; Pan, J.; Yan, Q.; Zhang, S.-C. *Proceedings of the National Academy of Sciences* **2018**, *115*, E6411–E6417.
- (126) Zubatyuk, R.; Smith, J. S.; Leszczynski, J.; Isayev, O. *Science Advances* **2019**, *5*, eaav6490.
- (127) Faber, F. A.; Lindmaa, A.; Von Lilienfeld, O. A.; Armiento, R. *Physical Review Letters* **2016**, *117*, 135502.
- (128) Belsky, A.; Hellenbrandt, M.; Karen, V. L.; Luksch, P. *Acta Crystallographica Section B: Structural Science* **2002**, *58*, 364–369.
- (129) Perdew, J. P.; Burke, K.; Ernzerhof, M. *Physical Review Letters* **1996**, *77*, 3865.
- (130) Sutskever, I.; Martens, J.; Hinton, G. E. In *Proceedings of the 28th International Conference of Machine Learning*, 2011, pp 1017–1024.
- (131) Stevenson, J. M.; Jacobson, L. D.; Zhao, Y.; Wu, C.; Maple, J.; Leswing, K.; Harder, E.; Abel, R. *arXiv preprint arXiv:1912.05079* **2019**.
- (132) Devereux, C.; Smith, J.; Davis, K.; Barros, K.; Zubatyuk, R.; Isayev, O.; Roitberg, A.; Isayev, O., et al. *ChemRxiv* **2020**.

- (133) Mulliken, R. S. *The Journal of Chemical Physics* **1955**, *23*, 1833–1840.
- (134) Hauser, K.; Negron, C.; Albanese, S. K.; Ray, S.; Steinbrecher, T.; Abel, R.; Chodera, J. D.; Wang, L. *Communications Biology* **2018**, *1*, 70.
- (135) Mey, A. S.; Jiménez, J. J.; Michel, J. *Journal of Computer-Aided Molecular Design* **2018**, *32*, 199–210.
- (136) Harger, M.; Li, D.; Wang, Z.; Dalby, K.; Lagardère, L.; Piquemal, J.-P.; Ponder, J.; Ren, P. *Journal of Computational Chemistry* **2017**, *38*, 2047–2055.
- (137) Williams-Noonan, B. J.; Yuriev, E.; Chalmers, D. K. *Journal of Medicinal Chemistry* **2017**, *61*, 638–649.
- (138) Matricon, P.; Ranganathan, A.; Warnick, E.; Gao, Z.-G.; Rudling, A.; Lambertucci, C.; Marucci, G.; Ezzati, A.; Jaiteh, M.; Dal Ben, D., et al. *Scientific Reports* **2017**, *7*, 6398.
- (139) Jiao, D.; Golubkov, P. A.; Darden, T. A.; Ren, P. *Proceedings of the National Academy of Sciences* **2008**, *105*, 6290–6295.
- (140) Senftle, T. P.; Hong, S.; Islam, M. M.; Kylasa, S. B.; Zheng, Y.; Shin, Y. K.; Junkermeier, C.; Engel-Herbert, R.; Janik, M. J.; Aktulga, H. M., et al. *npj Computational Materials* **2016**, *2*, 1–14.
- (141) Furman, D.; Wales, D. J. *The Journal of Physical Chemistry Letters* **2019**, *10*, 7215–7223.
- (142) Xie, T.; Grossman, J. C. *Physical Review Letters* **2018**, *120*, 145301.
- (143) Salatin, T. D.; Jorgensen, W. L. *The Journal of Organic Chemistry* **1980**, *45*, 2043–2051.
- (144) Gasteiger, J.; Hutchings, M. G.; Christoph, B.; Gann, L.; Hiller, C.; Löw, P.; Marsili, M.; Saller, H.; Yuki, K. In *Organic Synthesis, Reactions and Mechanisms*; Springer: 1987, pp 19–73.
- (145) Bauer, J.; Fontain, E.; Forstmeyer, D.; Ugi, I. *Tetrahedron Computer Methodology* **1988**, *1*, 129–132.
- (146) Satoh, H.; Funatsu, K. *Journal of Chemical Information and Computer Sciences* **1995**, *35*, 34–44.
- (147) Wei, J. N.; Duvenaud, D.; Aspuru-Guzik, A. *ACS Central Science* **2016**, *2*, 725–732.
- (148) Coley, C. W.; Barzilay, R.; Jaakkola, T. S.; Green, W. H.; Jensen, K. F. *ACS Central Science* **2017**, *3*, 434–443.

- (149) Corey, E.; Jorgensen, W. L. *Journal of the American Chemical Society* **1976**, *98*, 189–203.
- (150) Kayala, M. A.; Azencott, C.-A.; Chen, J. H.; Baldi, P. *Journal of Chemical Information and Modeling* **2011**, *51*, 2209–2222.
- (151) Kayala, M. A.; Baldi, P. *Journal of Chemical Information and Modeling* **2012**, *52*, 2526–2540.
- (152) Fooshee, D.; Mood, A.; Gutman, E.; Tavakoli, M.; Urban, G.; Liu, F.; Huynh, N.; Van Vranken, D.; Baldi, P. *Molecular Systems Design & Engineering* **2018**, *3*, 442–452.
- (153) Bonchev, D., *Chemical graph theory: introduction and fundamentals*; CRC Press: 1991; Vol. 1.
- (154) Trinajstić, N., *Chemical graph theory*; Routledge: 2018.
- (155) Kipf, T. N.; Welling, M. *arXiv preprint arXiv:1609.02907* **2016**.
- (156) Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. *arXiv preprint arXiv:1710.10903* **2017**.
- (157) Bronstein, M. M.; Bruna, J.; LeCun, Y.; Szlam, A.; Vandergheynst, P. *IEEE Signal Processing Magazine* **2017**, *34*, 18–42.
- (158) Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P. S. *arXiv preprint arXiv:1901.00596* **2019**.
- (159) Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. In *Advances in Neural Information Processing Systems*, 2015, pp 2224–2232.
- (160) Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T. *Drug discovery today* **2018**, *23*, 1241–1250.
- (161) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp 1263–1272.
- (162) Patent reaction extraction: downloads. <https://bitbucket.org/dan2097/patent-reaction-extraction/downloads>.
- (163) Xu, L.; Hilton, M. J.; Zhang, X.; Norrby, P.-O.; Wu, Y.-D.; Sigman, M. S.; Wiest, O. *Journal of the American Chemical Society* **2014**, *136*, 1960–1967.
- (164) Schwaller, P.; Laino, T.; Gaudin, T.; Bolgar, P.; Hunter, C. A.; Bekas, C.; Lee, A. A. *ACS Central Science* **2018**, *5*, 1572–1583.

- (165) Andersen, J. L.; Flamm, C.; Merkle, D.; Stadler, P. F. In *International Conference on Graph Transformation*, 2017, pp 54–69.
- (166) Nielsen, M. K.; Ahneman, D. T.; Riera, O.; Doyle, A. G. *Journal of the American Chemical Society* **2018**, *140*, 5004–5008.
- (167) RDKit: Open-source cheminformatics. <http://rdkit.org>.
- (168) Douglas, B. L. *arXiv preprint arXiv:1101.5211* **2011**.
- (169) rexgen direct: Template-free prediction of organic reaction outcomes using graph convolutional neural networks. https://github.com/connorcoley/rexgen_direct/commits/master.
- (170) Brennan, M. C.; Zinna, J.; Kuno, M. *ACS Energy Letters* **2017**, *2*, 1487–1488.
- (171) Protesescu, L.; Yakunin, S.; Bodnarchuk, M. I.; Krieg, F.; Caputo, R.; Hendon, C. H.; Yang, R. X.; Walsh, A.; Kovalenko, M. V. *Nano Letters* **2015**, *15*, 3692–3696.
- (172) Kuno, M.; Lee, J.-K.; Dabbousi, B. O.; Mikulec, F. V.; Bawendi, M. G. *The Journal of Chemical Physics* **1997**, *106*, 9869–9882.
- (173) Efros, A. L.; Rosen, M.; Kuno, M.; Nirmal, M.; Norris, D. J.; Bawendi, M. *Physical Review B* **1996**, *54*, 4843.
- (174) Giannozzi, P.; Baroni, S.; Bonini, N.; Calandra, M.; Car, R.; Cavazzoni, C.; Ceresoli, D.; Chiarotti, G. L.; Cococcioni, M.; Dabo, I., et al. *Journal of physics: Condensed matter* **2009**, *21*, 395502.
- (175) Cottingham, P.; Brutchey, R. *Chemical Communications* **2016**, *52*, 5246–5249.
- (176) Heidrich, K.; Schäfer, W.; Schreiber, M.; Söchtig, J.; Trendel, G.; Treusch, J.; Grandke, T.; Stolz, H. *Physical Review B* **1981**, *24*, 5642.
- (177) Begum, R.; Parida, M. R.; Abdelhady, A. L.; Murali, B.; Alyami, N.; Ahmed, G. H.; Hedhili, M. N.; Bakr, O. M.; Mohammed, O. F. *Journal of the American Chemical Society* **2017**, *139*, 731–737.
- (178) Zhang, X.; Lin, H.; Huang, H.; Reckmeier, C.; Zhang, Y.; Choy, W. C.; Rogach, A. L. *Nano Letters* **2016**, *16*, 1415–1420.
- (179) Stoumpos, C. C.; Malliakas, C. D.; Peters, J. A.; Liu, Z.; Sebastian, M.; Im, J.; Chasapis, T. C.; Wibowo, A. C.; Chung, D. Y.; Freeman, A. J.; Wessels, B. W.; Kanatzidis, M. G. *Crystal Growth & Design* **2013**, *13*, 2722–2727.

- (180) Protesescu, L.; Yakunin, S.; Bodnarchuk, M. I.; Krieg, F.; Caputo, R.; Hendon, C. H.; Yang, R. X.; Walsh, A.; Kovalenko, M. V. *Nano Letters* **2015**, *15*, 3692–3696.
- (181) Allan, G.; Delerue, C. *Physical Review B* **2004**, *70*, 245321.
- (182) Ten Brinck, S.; Infante, I. *ACS Energy Letters* **2016**, *1*, 1266–1272.
- (183) Brandt, R. E.; Stevanović, V.; Ginley, D. S.; Buonassisi, T. *MRS Communications* **2015**, *5*, 265–275.
- (184) Borštnik, U.; VandeVondele, J.; Weber, V.; Hutter, J. *Parallel Computing* **2014**, *40*, 47–58.
- (185) Hutter, J.; Iannuzzi, M.; Schiffmann, F.; VandeVondele, J. *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2014**, *4*, 15–25.
- (186) Guidon, M.; Hutter, J.; VandeVondele, J. *Journal of Chemical Theory and Computation* **2010**, *6*, 2348–2364.
- (187) Guidon, M.; Hutter, J.; VandeVondele, J. *Journal of Chemical Theory and Computation* **2009**, *5*, 3010–3021.
- (188) Genovese, L.; Deutsch, T.; Goedecker, S. *The Journal of Chemical Physics* **2007**, *127*, 054704.
- (189) Genovese, L.; Deutsch, T.; Neelov, A.; Goedecker, S.; Beylkin, G. *The Journal of Chemical Physics* **2006**, *125*, 074105.
- (190) VandeVondele, J.; Krack, M.; Mohamed, F.; Parrinello, M.; Chassaing, T.; Hutter, J. *Computer Physics Communications* **2005**, *167*, 103–128.
- (191) Frigo, M.; Johnson, S. G. *Proceedings of the IEEE* **2005**, *93*, 216–231.
- (192) VandeVondele, J.; Hutter, J. *The Journal of Chemical Physics* **2003**, *118*, 4365–4369.
- (193) Heyd, J.; Scuseria, G. E. *The Journal of Chemical Physics* **2004**, *120*, 7274–7280.
- (194) VandeVondele, J.; Hutter, J. *The Journal of Chemical Physics* **2007**, *127*, 114105.
- (195) Krack, M. *Theoretical Chemistry Accounts* **2005**, *114*, 145–152.
- (196) Hartwigsen, C.; Goedecker, S.; Hutter, J. *Physical Review B* **1998**, *58*, 3641–3662.
- (197) Goedecker, S.; Teter, M.; Hutter, J. *Physical Review B* **1996**, *54*, 1703–1710.

- (198) Dou, L.; Wong, A. B.; Yu, Y.; Lai, M.; Kornienko, N.; Eaton, S. W.; Fu, A.; Bischak, C. G.; Ma, J.; Ding, T., et al. *Science* **2015**, *349*, 1518–1521.
- (199) Emin, D. *Physical Review B* **1971**, *4*, 3639.
- (200) Bethe, H. A.; Salpeter, E. E., *Quantum Mechanics of One- and Two-Electron Atoms*; Springer: 1957.
- (201) Sayfutyarova, E. R.; Chan, G. K.-L. *The Journal of Chemical Physics* **2016**, *144*, 234301.
- (202) Shao, Y. et al. *Molecular Physics* **2015**, *113*, 184–215.