Machine Learning-Accelerated Computing of Molecular and Catalytic Properties of Metallic Alloys

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

Department of Chemical Engineering

Yilin Yang

B.S., Chemical Engineering, Zhejiang University
M.S., Machine Learning, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, Pennsylvania

April, 2022

Acknowledgments

There are so many people who helped me in my PhD journey. Without them, I would not be able to write this dissertation. I would like to thank my advisor, Prof. John Kitchin for his instruction, guidance, and support throughout my PhD. He always expresses enthusiasm in our discussions and encourages me to explore new directions when I feel lost in my research. I would also like to thank him for building the scimax toolkit for writing and coding, with which I have tried and written up many interesting things. Also, his passion for new techniques will keep inspiring me to learn new things in the future.

I am sincerely grateful to my thesis committee, Prof. Andrew Gellman, Prof. Chrysanthos Gounaris, and Prof. Alan McGaughey. I would like to thank them for their constructive comments and feedback from my proposal to this dissertation. I would especially thank Prof. Andrew Gellman. His surface science and kinetics classes laid the foundation of my research works. His valuable insights and suggestions also helped me a lot during my collaboration with his group.

I would like to thank Prof. Zachary Ulissi for his helpful research advice and computing resources. I am also grateful to Justin Dawber for maintaining computing services. I still remember Justin and my advisor working together to solve issues of the Gilgamesh server over several weeks. Also thanks to the administrative staff for their efforts to create a stable working environment even under special conditions.

I would like to gratefully acknowledge the funding sources that supported my research: National Science Foundation DMREF Award CBET-1921946 and the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, Catalysis program under Award No. DE-SC0018187.

I would also like to thank my friends and roommates Zicheng Cai and Zhitao Guo, their help and support are important to my PhD life. Also great thanks to Mingjie Liu, Ni Zhan, Maya Bhat, Siddarth K. Achar, Omar A. Jiménez-Negrón and Muhammed Shuaibi for helpful discussion and collaboration. Especially to Mingjie Liu, we worked together on two research projects, which constitute two chapters in this dissertation.

Finally, and most importantly, I would like to express my deep sense of gratitude to my mom for her meticulous and consistent care, to my father, who always supports my decisions, and to God, who is all, and in all.

Abstract

Metallic alloys are important materials in engineering for their versatile properties. With the development of computational power, molecular simulation plays an increasingly important role to study the properties of metallic alloys. Molecular simulation is used to calculate the properties that are hard or expensive to be measured experimentally. Traditional molecular simulation relies on density functional theory (DFT) or empirical potentials to calculate the energies and forces during the simulations. DFT provides higher accuracy, but it is more time-consuming. Empirical potentials are much faster but they are not as accurate as DFT. Recently, machine learning (ML) potentials have become attractive because they are potentially both accurate and fast. When trained with sufficient data, machine learning potentials could be as accurate as DFT. Also, because machine learned potentials are essentially regression models, their inferences could be as fast as empirical potentials. With the help of machine learning models, simulations for large molecular systems can be accurately finished with a reasonable computational cost.

In this dissertation, we combined ML methods, DFT, and Monte Carlo (MC) simulations to study the surface segregation of the CuPdAu alloy under vacuum conditions. We built a neural network (NN) model to accurately approximate the DFT potential energies during the MC simulations which were used to estimate the surface compositions of the CuPdAu alloy with various bulk compositions. A range of factors that might contribute to surface segregation was investigated, such as surface relaxation, vibrational contribution, and orientation dependence. Next, we developed a NN ensemble-based active learning method to accelerate the geometry optimization process, which enabled us to obtain the ground-state structures in a faster way. We used a NN-ensemble approach to provide the uncertainty estimation of the NN prediction during the molecular simulation, such that we were able to replace the DFT calculation using the NN model appropriately during the geometry optimization process. After that, the surface segregation and aggregation phenomenon was investigated using semi-grand canonical Monte Carlo simulation with the help of ML surrogate models for the bulk, slab potential energies, and the pseudo-adsorption energies. Then, we illustrated using automatic differentiation to evaluate the degree of rate control (DRC). Automatic differentiation provided higher accuracy and faster computation of the DRC. Lastly, we developed an efficient method to search for similar molecular structures in a large database. The search method was based on the approximate nearest neighbor search and machine learning embedding.

Contents

Abstract vi

Contents ix

List of Tables x

List of Figures xvi

1 Introduction 1

2 Machine Learning Potentials 4
2.1 Introduction 4
2.2 Neural Network-based ML Potentials 5
2.2.1 Atomic Centered Symmetry Functions 6
2.2.2 Feed-forward Neural Networks 8
2.2.3 Graphical Neural Networks for ML Potential 11
2.2.4 Implementation and Softwares 11
2.3 Kernel Methods-based ML Potentials 12

3 Simulating segregation in a ternary Cu-Pd-Au alloy 14
3.1 Introduction 14
3.2 Methods 16
3.2.1 Density Functional Theory 16
3.2.2 Experimental Data 16
3.2.3 Neural Network 16
3.2.4 Training Samples across the G^2 Fingerprints Space . . 17
3.2.5 Monte Carlo Simulation 19
3.3 Results 20
3.3.1 Performance of Neural Network 20
3.3.2 Monte Carlo Simulation Results 22
3.3.3 Phase Separation in Experiments 25
3.3.4 The Role of Surface Relaxation 26
3.3.5 Vibrational Contribution to the Surface Segregation . . 27
3.3.6 Orientation dependence of CuPd surface segregation mode 29
3.3.7 DFT errors in alloy formation energies 32
3.4 Conclusions 32

4 Machine-learning accelerated geometry optimization in molecular simulation 34
4.1 Introduction 34
4.2 Methods 37
4.2.1 ML Surrogate Model for Potential Energy Surface . . 37
4.2.2 Relaxation with Active Learning 41
4.2.3 Application Dataset 43

4.3 Results 45
4.3.1 Active learning for geometry optimization of single configuration 45
4.3.2 Further acceleration by information sharing among configurations and utilizing prior data 45
4.3.3 Performance of the active learning on more complex systems and nudged elastic band calculations 50
4.3.4 Limiting the training data to recent configurations for training efficiency 53
4.4 Conclusion 56

5 Simulating Surface Segregation and Aggregation of AgPd Alloy in the Presence of Acrolein 57
5.1 Introduction 57
5.2 Methods 61
5.2.1 DFT Calculations for Potential Energies and Adsorption Energies 61
5.2.2 DFT investigation on the Pd Segregation with Acrolein 61
5.2.3 ML surrogate models for Potential and Adsorption Energies 63
5.2.4 Semi-grand Canonical Monte Carlo Simulation 66
5.3 Results 69
5.3.1 DFT Calculation for the Pd Segregation with Acrolein 69
5.3.2 Performance of the ML Surrogate Models 70
5.3.3 Semi-grand Canonical Monte Carlo Simulation Results 73
5.4 Conclusion 78

6 Searching Similar Molecular Structures with Neural Network Embeddings 81
6.1 Introduction 81
6.2 Methods 84
6.2.1 Searching Similar Molecules via Neural Network Embeddings 84
6.2.2 GemNet to Generate Atom Embeddings 84
6.2.3 Approximate Nearest Neighbor (ANN) Search 86
6.2.4 Datasets 88
6.3 Results 89
6.3.1 ANN Search for Organic Molecules 89
6.3.2 ANN Search for Bulk Local Environments 93
6.3.3 ANN Search for Surfaces 95
6.4 Conclusion 100

7 Evaluation of the Degree of Rate Control via Automatic Differentiation 104
7.1 Introduction 104

7.2 Methods 108
7.2.1 Automatic Differentiation 108
7.2.2 Distributed Evaluation of Local Sensitivity Analysis . . 112
7.3 Results 113
7.3.1 Case I: Hypothetical Two-Step Reaction 114
7.3.2 Case II: Redox Mechanism for Water-gas Shift 116
7.3.3 Case III: Propylene Partial Oxidation 119
7.3.4 Degree of Rate Control for Uncertain Parameters . . . 122
7.4 Conclusion 123

8 Conclusions 125
8.1 Machine Learning for Prediction, Exploration and Searching . 125
8.2 Automatic Differentiation for Fast and Accurate Derivatives . 127
8.3 Potential Directions for Future Works 127

References 151

Appendix A Density of the Training Data Sampling 152

Appendix B ACSF Parameters 152

Appendix C AgPd Slab Configurations with Acrolein for DFT Calculation 153

Appendix D Gaussian Process Regression 153

Appendix E Hyperparamters for GemNet model, Faiss searching and Flare GP model 154

Appendix F Supplementary Examples for ANN Search 156
F.1 ANN Search Examples for Molecular Substructures 156
F.2 ANN Search Example for Similar Bulk Environment 156
F.3 Supporting Configurations for the OC20 Case 159

List of Tables

3.1 Potential energy and Helmholtz free energy change for segregation of Pd in Cu and Cu in Pd. 28

3.2 CuPd segregation energy on FCC(111) and FCC(110). 30

3.3 Experimental and DFT formation energies for CuAu alloys . . 32

4.1 EMT or DFT calls queried by NN ensemble with active learning, EMT with MDMin and VASP with built-in quasi-Newton optimizer for Pt heptamer rearrangement and acetylene hydrogenation. 52

6.1 Euclidean distances between the GemNet embeddings of the searched atoms (excluding the query atom itself) and the query atom. Configuration ID b, c and d correspond to the configurations b, c and d shown in Figure 6.6. 95

7.1 Forward AD for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Left side shows the forward function evaluations. Right side shows the derivative of $\frac{\partial y}{\partial x_1}$. Head dot means $\frac{\partial}{\partial x_1}$ 110

7.2 Reverse AD for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Left side shows the forward function evaluations. Right side shows the back-propagation of the derivative of $\frac{\partial y}{\partial x_1}$. 110

7.3 Hypothetical Two-Step Catalytic Reaction (case I).[162] 115

7.4 Redox mechanism for water-gas shift (case II).[172] 118

7.5 Elementary steps (step 1 to step 17 from top to bottom) for the propylene partial oxidation (case III).[173] E_a and E_a^{-r} are the activation energies for the forward and reverse directions. A and A^{-r} are the pre-exponential factor of the forward and reverse reactions. v represents the oxygen vacancy. 120

7.6 Average percentage difference between the solutions of the FD with different perturbations and the AD method over the time axis. 122

B.1 The ACSF parameters for Chapter 3. 152

B.2 The ACSF parameters for Chapter 5. 153

E.1 Hyperparameters for the GemNet model used in Chapter 6. More details about the hyperparameters can be found in the OC20 GitHub repository. 155

E.2 Hyperparameters for the Faiss IndexIVFPQ method. More details about the hyperparameters can be found in Faiss wiki. 155

E.3 Hyperparameters for the Flare GPR model. 155

List of Figures

2.1 Structure of the Behler-Parrinello neural network. $\mathbf{R_i}$ is a vector of the Cartesian coordinates of atom i. $\mathbf{G_i}$ represents the vector of symmetry function values describing the local environment of atom i. E_i is the atomic energy contributed by atom i. E is the total energy of the N atoms. 6

2.2 Radial symmetry functions with different η. The cutoff radius used in the plot is 6.0 Å. 8

2.3 Typical example of a FFNN which takes an input vector $\mathbf{x}$ and outputs a scalar y. 9

3.1 Fingerprints calculated by $\eta = 0.05$ for surface slabs with different structures ($1 \times 1 \times 7$, $2 \times 1 \times 7$, $\sqrt{3} \times \sqrt{3} \times 7$, $\sqrt{7} \times \sqrt{7} \times 5$) and lattice constants (3.61 Å, 3.73 Å, 3.84 Å, 3.96 Å and 4.08 Å). The x, y, z axes are the fingerprint values for different surrounding elements in the atomic local environment bounded by the cutoff radius. In the left figure, fingerprints of the same lattice constant are located in three planes, which represents the bulk, subsurface and surface environments from top to bottom. In the right figure, fingerprints in the same plane are distributed around a triangle, where the three angles represent the local environments purely consist of three different elements. 18

3.2 Fingerprints of bulk atoms in the training slabs with lattice constant of 3.64 Å. The fingerprints in the figure were calculated with the η of 0.05. 19

3.3 Distribution of the NN residual error on the training, validation and test set. Figure (a) shows the MAE and the residual error of the training and the validation set which share the same distribution of the slab configurations. Figure (b) shows the MAE and the residual error of the test set which contains slabs with larger unit cell than the training data. 21

3.4 MC trajectory of the top layer concentration on a $10 \times 10 \times 15$ FCC(111) slab with bulk concentrations of 20:23:47 for Pd:Cu:Au over 15000 successful steps. The colored dashed lines are the bulk concentrations while the solid lines are the surface concentrations. The last 6000 steps were used to calculated the average surface concentration and the standard deviation which represented by the grey dashed lines. 22

3.5 MC simulation results for 24 bulk concentrations at 600 K. These plots show the excess surface composition compared to the bulk composition for Cu, Pd and Au. The first row is for the simulation results, and the second row shows the experimental results.[60] 23

3.6 Segregation energies for Au-Cu, Au-Pd, Pd-Cu pairs derived from simulation and experimental data. Each dot corresponds to a different bulk composition. 25

3.7 Potential energy difference before and after atomic swaps calculated by NN without surface relaxation (left), DFT with surface relaxation (middle), as well as their pairwise difference (right). 27

3.8 Parity plot of the potential energy change before and after atomic swap calculated by NN without surface relaxation and DFT with surface relaxation. 28

3.9 Segregation profile of CuPd(110) at 600 K. Blue squares show the MC simulation results with surface relaxation. Orange triangles show the simulation results without surface relaxation. Red points show the experimental results.[60] 31

4.1 SingleNN structure for atomic energy prediction. The use of each variable is shown in Equations 4.1 - 4.3 and described in the text. 38

4.2 Surrogate machine learning models for the Lennard Jones potential. Left plot shows the GPR while the right plot shows the NN ensemble. Both models have low prediction variance in the region of training set and high variance for the data that is far from the training set. 41

4.3 Framework for relaxation with online active learning. The overall workflow starts with the initial configurations that needs to be relaxed. At first, the DFT energies and forces are calculated and the NN ensemble is trained with these initial information. Then the model is utilized with optimizers to reduce the energy of the configurations. The relaxation with model stops when encountering with uncertain configurations or reaching the relaxation criterion. The uncertain configurations are submitted for further DFT calculations. . . 43

4.4 Comparison of the number of DFT calls between active learning with NN ensemble and quasi-Newton built in VASP when each configuration is relaxed independently. 46

4.5 Number of DFT calls for three different active learning settings for the relaxation of acrolein/AgPd(111). The blue line represents the single configuration mode, the orange line is for the multiple configurations mode and the green line denotes the multiple configurations with warm up. The red line serves as a baseline which is the performance of GPR model implemented according to previous literatures.[81,82] For comparison, with no ML it takes about 193 DFT calls to converge. 48

4.6 Offline relaxation on 13 acrolein/AgPd configurations using NN trained on 243 existing relaxation trajectories. Blue points show the maximum DFT forces for the initial configurations. Orange scatters are the maximum DFT forces for the NN relaxed configurations while purple dots are the NN maximum forces. 49

4.7 Comparison of active learning (AL) and VASP quasi-Newton (QN) method on relaxing three different structures: bare AuPd slab, CO on AuPd icosahedron and acrolein on AgPd slab. . . 51

4.8 Climbing NEB curves generated by NN ensemble and (a) EMT for Pt heptamer rearrangement (b) DFT for acetylene hydrogenation over Pd FCC(111) surface. 52

4.9 Time spent on the training process using a single NN with 2 layers and 50 neurons at each layer over iterations. The blue line shows the time for the model trained on all queried configurations while the orange line shows the time for training on the training set with fixed size. The experiment is repeated 10 times and the shaded area is the standard deviation for the 10 experiments. Time measured on 4 CPU cores. 54

4.10 Scaled Pearson correlation coefficient between the intermediate configurations and the final relaxed configuration. The Pearson correlation is scaled by the base correlation between the initial configuration and the final configuration. 55

5.1 ML-SGCMC workflow used to simulate the surface surface segregation and aggregation phenomenon of AgPd in the presence of the acrolein. Three ML models serve as the surrogate models for the bulk potential, slab potential and the adsorption energies. SGCMC simulations are powered by these three ML models to study the Δ chemical potentials for different bulk compositions, as well as the surface segregation and aggregation phenomenon. 60

5.2 Pd segregation energies under vacuum (Pd ΔE_{seg}), acrolein adsorption energies (Acrolein ΔE_{ads}), and Pd segregation energies with acrolein ($\Delta E_{seg} + \Delta\Delta E_{ads}$) of different AgPd configurations. $\Delta\Delta E_{ads}$ means the difference of the acrolein adsorption energy on an adsorption site and on pure Ag. . . . 70

5.3 Parity plots for the surrogate model (SingleNN) of AgPd bulk potential energies. Left figure shows the performance on the training, validation, and test set. Right figure shows the prediction difference of the SingleNN models trained from different training setss (split by differnet random seeds). . . . 71

5.4 Parity plots for the surrogate model (SingleNN) of the AgPd slab potential energies. Left figure shows the performance on the training, validation, and test set. Right figure shows the prediction difference of the SingleNN models trained from different training set (split by differnet random seeds). 72

5.5 (a) Parity plot for the pseudo-adsorption energy surrogate model. Dashed lines indicate an error of 0.05 eV. (b) Histogram of the prediction error. (c) Typical configuration with large prediction error. Configuration with red pseudo atom is the test configuration while the grey pseudo atom indicates the label configuration. They have different number of Pd atoms in the sublayer. (d) Typical configuration with small prediction error. Red and grey pseudo atom indicate the test and label configuration respectively. They have the same number of Pd atoms in the sublayer. 73

5.6 Relationship between the AgPd bulk composition and $\Delta\mu_{Ag-Pd}$. Blue points show the results of the simulations without changing the lattice constant with the bulk compositions. Orange points show the simulations with changing the lattice constant according to the Vegard's law.[66] The error bars show the standard deviation of the simulations using 5 bulk SingleNN models trained from the datasets with different split. 75

5.7 SGCMC simulation results for the AgPd slab under vacuum condition. Pd concentrations in the first and second layer are colored as blue and orange. Error bars represent the standard deviations of simulations using 5 different SingleNN model trained from different dataset split. 76

5.8 AgPd segregation results in the presence of acrolein on the surface. Error bars represent the standard deviation of the simulations performed with different SingleNN models. 78

5.9 Pd adsorption site distribution in the presence of acrolein. Error bars represent the standard deviation of the simulations performed with different SingleNN models. 79

6.1 Framework for similar molecules search using neural network embeddings and ANN search methods. The red frame encloses the query stage and the blue frame denotes the preparation stage. 85

6.2 Retrieved similar molecules (top 5) for benzene. Figure a is benzene used as the query molecule. Figures b to f show the nearest molecules in the QM9 dataset. 90

6.3 Searched similar molecules (top 5) using Morgan fingerprint with Tanimoto coefficient as a distance measure. 91

6.4 Prediction performance of GPR models trained on molecules searched using GemNet embeddings and Morgan fingerprints, as well as a set of molecules randomly sampled from the QM9 dataset. Left figure shows the prediction error while the right figure shows the standard deviation (std) of the GPR prediction. Number of added configuration means the numebr of molecules added into the training set to build the GP regression model. The annotations in the figures are the mininum prediction error and standard deviation for the models trained on different number of configurations. 92
6.5 Joint search result for hydroxyl and amino groups. The query substructures are marked as crossed at the left of the arrow. The retrieved molecule is at the right side with matched atoms also marked as crossed. 93
6.6 Top 7 nearest atoms to the query oxygen atom in the Materials Project dataset. Atom 9 in figure a is the query atom. Atom 10, 13, 15, and 17 in figure b, atom 6, 8 in figure c, and atom 17 in figure d are the searched atoms. 95
6.7 Local configurations of the query and searched atoms (marked as crossed). Configuration a is the query oxygen and configurations b to d are the retrieved atoms. 97
6.8 Density of states projected onto the p-orbital of the query and searched oxygen atoms. Figures a to d correspond the configurations a to d in Figure 6.7. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data. . . 98
6.9 Density of states projected onto the p-orbital of four randomly selected oxygen atoms. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data. . . 99
6.10 Configurations of the query (config. a) and top 3 retrieved acetylene adsorption configurations (config. b to d). The query and matched carbon atoms are marked as crossed. 100
6.11 Density of states projected onto the p-orbital of the selected query and searched carbon atoms. Figures a to d correspond the configurations a to d in Figure 6.10. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data. . . 101

6.12 Density of states projected onto the p-orbital of the carbon atoms in four randomly selected acetylene adsorption configurations. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data. 102
7.1 Computational graph for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Each blue node contains the corresponding variable and the operation that applied on its parent node(s). For instance, w_3 is the result of the multiplication between a constant 3 and another node variable w_1. 109
7.2 Degree of rate control for the hypothetical reaction of case 1 evaluated by analytic solution and automatic differentiation. . 117
7.3 Degree of rate control for the water-gas shift reaction (case II).[172] 118
7.4 Degree of rate control for the propylene partial oxidation reaction (case III).[173] 120
7.5 Transient degree of rate control evaluated by the finite difference and automatic differentiation for the propylene partial oxidation reactions 3, 4, 14 and 17 (case III).[173] Different perturbation sizes are used in the FD method as indicated in the legend. . . 121
7.6 DELSA results for 3 important steps (3, 14 and 17) in the propylene oxidation reaction.[173] 123
A.1 Distribution of the distances to the nearest neighbors for the points in the fingerprint space. 152
C.1 Relaxed AgPd configurations with acrolein 153
F.1 Similar molecules (b to f) retrieved from querying toluene molecule (a). 156
F.2 Similar -N(H)- substructure (b to f) retrieved from querying -N(H)- substructure (a). 157
F.3 Similar -NH_2 substructure (b to f) retrieved from querying -NH_2 substructure (a). 157
F.4 Similar -OH substructure (b to f) retrieved from querying -OH substructure (a). 158
F.5 Top 10 nearest atoms to a palladium query atom in the Materials Project dataset. Atom 2 in figure a is the query atom. Atom 2 to 7 in figure b, atom 1, 2 in figure c, and atom 1, 3 in figure d are the searched atoms. 158
F.6 Four randomly selected oxygen atoms in the OC20 dataset. . . 159
F.7 Zoomed-in local configurations for the acetylene search example. 160
F.8 Four randomly selected acetylene adsorption systems in the OC20 dataset. 160

1 Introduction

Metallic alloys are important catalysts for various chemical reactions.[1,2] For example, CuPd alloys for the NO reduction reaction,[3] PtCu alloys for the glycerol hydrogenolysis reaction,[4] and AuPd alloys for the N_2O decomposition reaction.[5] Catalytic properties such as activity and selectivity highly depend on the surface properties of the metallic catalysts. However, it is difficult to control the surface properties of the alloy catalysts during production since surface composition usually differs from the bulk composition due to the surface segregation phenomenon. Therefore, investigations into surface segregation are necessary for the design of the alloy catalysts. With the dramatic improvement of the computational capacity, molecular simulations become much more significant to investigate and design alloy catalysts.[6–8] However, even with current computational capacity, it is still computationally expensive to calculate the properties of a large molecular system (e.g., thousands of atoms) using *ab initio* calculations such as DFT, which limits the scale of the materials that we can simulate. We must find a way to conduct the molecular simulations at a larger scale to get more detailed properties under current computational constraints. Greater computational power also comes with a requirement to retrieve data efficiently. Nowadays, there are many scientific databases storing a huge amount of molecular structures and properties.[9–11] An efficient method to search for and retrieve molecular structures of interest is necessary for us to utilize these databases effectively.

In this dissertation, we investigated approaches to address these issues. The dissertation is organized as follows. Chapter 2 introduces the machine learning potentials which is a potential solution to the trade-off between accuracy and speed. Different types of machine learning potentials were discussed. We also briefly introduced the steps to build a machine learning potential for a specific task. The packages and software to implement the machine learning potentials were also covered in this chapter. After that, the surface segregation phenomenon of a ternary CuPdAu alloy is investigated in Chapter 3. A NN model was built to approximate the potential energy surface (PES) of the CuPdAu slabs. In this way, potential energies with the DFT accuracy of the alloy slabs could be calculated with a low computational cost during the Monte Carlo (MC) simulations. These simulations provided the equilibrium surface compositions for different bulk compositions. We also compared the simulation results with the experimental data, which were partially consistent. Possible reasons for the partial discrepancy were substantially investigated. In Chapter 4, we developed an active learning framework to speed up the geometry optimization for molecules. NN ensemble served as a measure of uncertainty for the NN predictions. It was used as a metric to determine when the NN predictions could be trusted and when queries for new DFT calculations were required. By integrating the NN model into the geometry optimization process, we were able to obtain the DFT-verified relaxed structures with much fewer DFT calculations. This active learning framework was used in Chapter 5 to acquire the relaxed structures for hundreds of acrolein-adsorbed AgPd slabs with much less computational cost. These structures were utilized to train a machine learning model to predict the pseudo-adsorption energies of acrolein/AgPd slabs, which was combined with the surrogate models of the bulk and slab potential energies to provide the required energy data during the simulations. Semi-grand canonical Monte Carlo simulation (SGCMC) was utilized to study the surface aggregation and segregation phenomenon of the AgPd alloys with acrolein adsorbed on the surface. In

Chapter 6, we illustrated a way to efficiently search for similar molecular structures from a database. Machine learning embeddings were used to describe the atomic environments, which were applied with the approximate nearest neighbor (ANN) search methods to achieve fast searching for similar structures. In Chapter 7, we moved forward to the topic of the degree of rate control (DRC) of the catalysis. DRC achieves a quantitative measurement of the kinetic contribution of each elementary step to the overall reaction rate. We demonstrated the use of automatic differentiation (AD) to evaluate the DRCs of a reaction system, which led to higher accuracy and faster computational speed. Finally, we concluded this dissertation in Chapter 8 by discussing the works we achieved and potential directions for further research.

2 Machine Learning Potentials

2.1 Introduction

A fundamental trade-off in computational material science is between accuracy and computational cost. Simulations for large systems of atoms at a degree of accuracy of *ab initio* methods are desired, but it is computationally unaffordable with *ab initio* methods. Machine learning (ML) methods are promising to bring *ab initio* accuracy to large-scale molecular simulations at a reasonable computational cost. For example, it may takes several weeks to finish the energy evaluation of a 1000-atom alloy slab using DFT, but it only takes several seconds with a NN model, which has a comparable computational efficiency to classical force fields like the effective medium theory (EMT).[12]

Nowadays, ML techniques are applied in various fields of science and engineering in manifold ways. The basic idea of ML is that a computer program learns something from experience which can be used for future tasks according to the definition given by Tom M. Mitchell: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with the experience E".[13] Different ML methods are suitable for different application scenarios. Based on the form of the problems, classification or regression ML models can be adopted. ML models can also be chosen according to the form of the data that need to be processed. For example, the recurrent neural network is suitable for sequential data,[14] and the convolutional neural network is preferred for image data.[15] The basic demand in molecular simulation is an accurate representation of the atomic interactions which are usually indicated by energy and forces. In other words, the ML task is to accurately fit a complicated function that describes the atomic interactions of a molecular system. This complicated function is the potential energy surface (PES) which maps from the configurations to the energy and/or forces of an atomistic system.[16] Using the definition of ML above, the experience E is the data generated from *ab initio* calculations like DFT. The task T is to predict the energy and/or forces given an atomistic system. The performance measure P is the metric of interest, for example, the mean absolute error (MAE). The machine learning surrogate model for the potential energy surface is the machine learning potential.

2.2 Neural Network-based ML Potentials

Neural Network (NN) is a widely used group of regression methods to build the ML potential. The first application of NNs to fit the PES dates back to 1995 in the work of Doren and coworkers.[17] However, wide applications of NN for various molecular and material systems appear after Behler and Parrinello introduced the Behler-Parrinello neural network (BPNN) to represent high-dimensional PES in 2007,[18] where they proposed to use high-dimensional symmetry functions to describe the atomic configurations in a generalized and size-independent way.[18,19]

The core concepts underlying the BPNN are the atom-centered symmetry functions (ACSFs) and the feed-forward neural networks. Figure 2.1 shows the overall structure of the BPNN. Cartesian coordinates ($\mathbf{R}$) of the atoms are the inputs to the BPNN. They are processed by the symmetry functions and become high dimensional vectors ($\mathbf{G}$) describing the local environments of the atoms. These descriptors are invariant to transformation and rotation, which is required to model the potential energy surface. After that, these high dimensional vectors serve as the inputs of the feed-forward neural networks.

Each element type has its own NN weights and the output of each atomic NN is the atomic energy E_i representing the atomic contribution to the total energy E.

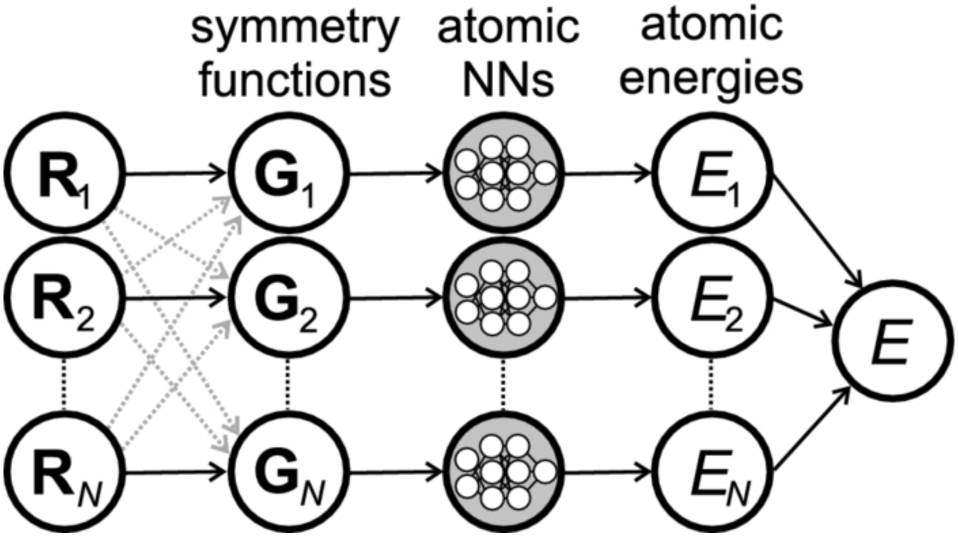

Figure 2.1: Structure of the Behler-Parrinello neural network. $\mathbf{R_i}$ is a vector of the Cartesian coordinates of atom i. $\mathbf{G_i}$ represents the vector of symmetry function values describing the local environment of atom i. E_i is the atomic energy contributed by atom i. E is the total energy of the N atoms.

2.2.1 Atomic Centered Symmetry Functions

Cartesian coordinates are not good to serve as the direct inputs of the regression models like a NN to construct the surrogate potential, because the total energy of an atomistic system does not change with translation or rotation. The invariance to translation or rotation should be kept in the ML potentials. Another requirement for the ML potential is that it needs to be continuous and differentiable since the forces should be obtainable from the derivatives of the potential energy surface. Symmetry functions are an option that satisfy these two requirements. Before defining the explicit form of the symmetry functions, Behler and Parrinello employed a cutoff function f_c to limit the atomic descriptors focusing more on the local environments.[18,19]

The cutoff function has the form of

$$f_c\left(R_{ij}\right)=\begin{cases}0.5\times\left[cos\left(\frac{\pi R_{ij}}{R_c}\right)+1\right] & for\ R_{ij}\leq R_c\\ 0 & for\ R_{ij}>R_c\end{cases}\tag{2.1}$$

where R_{ij} is the distance between atom i and atom j. The function value decays as the atomic distance R_{ij} increases from 0 to R_c, which means the interaction between a pair of atoms decreases as their distance increases. This cutoff function also eliminates the interaction between two atoms with a distance greater than R_c, which is the cutoff radius.

There are at least two types of symmetry functions to describe the atomic configurations. Radial symmetry functions are based on the radial distance between each pair of atoms. Angular symmetry functions consider the angular interactions in triples of atoms. Radial and angular symmetry functions are represented by G_i^2 and G_i^4 respectively.[18,19]

$$G_i^2=\sum_{j\neq i}e^{-\eta(R_{ij}-R_s)^2}f_c(R_{ij})\tag{2.2}$$

$$G_i^4=2^{1-\zeta}\sum_{j,k\neq i}(1+\lambda cos\theta_{ijk})^{\zeta}e^{-\eta(R_{ij}^2+R_{ik}^2+R_{jk}^2)}f_c(R_{ij})f_c(R_{ik})f_c(R_{jk})\tag{2.3}$$

Different parameters (η,λ,ζ,R_s) can be used to generate a set of values to describe a local environment in the configuration, and the choice of these parameters is dependent on different systems. In Figure 2.2, we show the radial symmetry functions with different η.

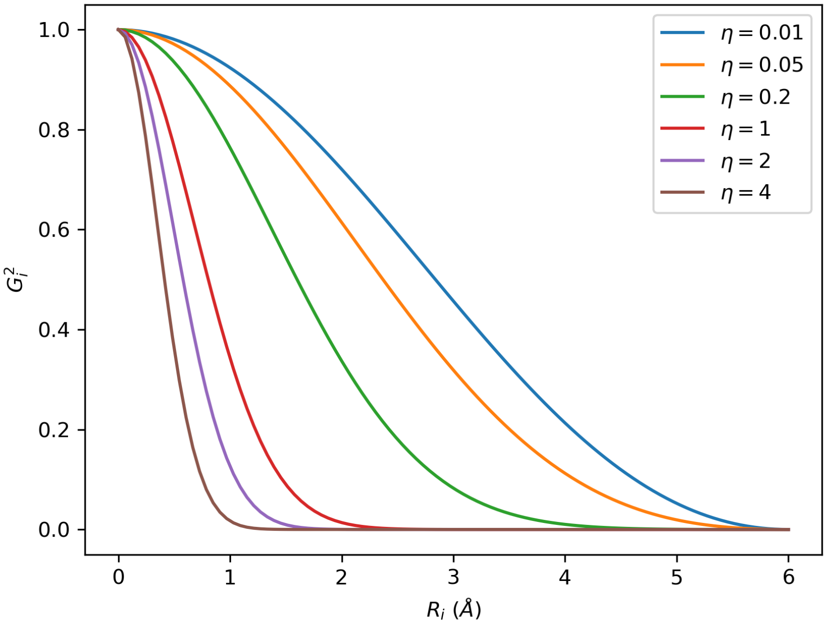

Figure 2.2: Radial symmetry functions with different η. The cutoff radius used in the plot is 6.0 Å.

With the BPNN, we have a mathematical model to describe the total energy of a configuration:

$$E = \sum_i E_i = \sum_M \sum_j f_M(G_j^M; \boldsymbol{w}^M) \tag{2.4}$$

in which $f_M(\cdot)$ is the regressed function by the NN of element M. $\boldsymbol{w}^M$ is the weight parameters in the NN of element M.

2.2.2 Feed-forward Neural Networks

A feed-forward neural network (FFNN) is responsible to map the ACSF vector to atomic energy in the BPNN framework. Since atoms with different identities usually have different contributions to the total energy even with the same surrounding environment, BPNN has separate FFNNs for the atoms with different elements. A typical example of a FFNN is shown in Figure 2.3.

In this example, it has two hidden layers and it takes a vector $\mathbf{x}$ and outputs a scalar y. Between two consecutive layers, there is a weight matrix $\mathbf{W}^l$ and a bias vector $\mathbf{b}^l$. Each hidden layer contains nodes of activation functions which are nonlinear functions like ReLU, sigmoid and hyperbolic tangent. The mathematical form of the function represented by the NN of Figure 2.3 is

$$\hat{y} = \mathbf{W}^{(2)} f_a^{(2)} \left(\mathbf{W}^{(1)} f_a^{(1)} \left(\mathbf{W}^{(0)} \mathbf{x} + \mathbf{b}^{(0)} \right) + \mathbf{b}^{(1)} \right) + \mathbf{b}^{(2)} \tag{2.5}$$

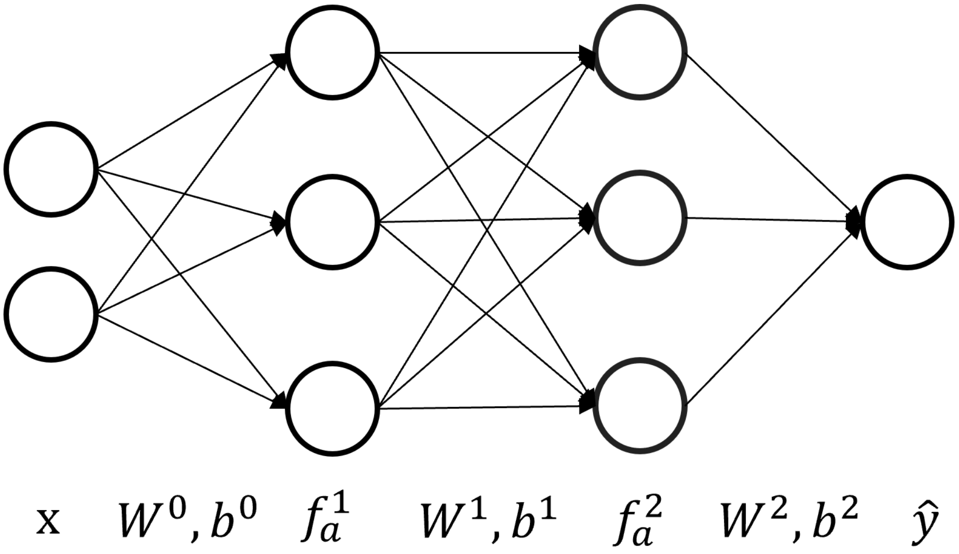

Figure 2.3: Typical example of a FFNN which takes an input vector $\mathbf{x}$ and outputs a scalar y.

The training of a NN is an optimization process to minimize an objective or loss function. The loss function usually represents the discrepancy between the NN predictions and the target labels. The mathematical form is

$$\mathbf{W}^*, \mathbf{b}^* = \underset{\mathbf{W},\mathbf{b}}{\arg\min}\, L\left(\mathbf{W}, \mathbf{b}; \mathbf{x}, y\right) \tag{2.6}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the trainable weights and biases, $\mathbf{x}$ is the feature of a data point, and y is the true label. L is the loss function that we want to minimize during the training process. A typical example for regression tasks is the mean square error (MSE):

$$L = \sum_i \frac{1}{N} \left(y_i - \hat{y_i}\right)^2 = \sum_i \frac{1}{N} \left[y_i - f\left(\mathbf{x}_i, \mathbf{W}, \mathbf{b}\right)\right]^2 \tag{2.7}$$

Here, N is the number of training points. f is the function represented by a NN. Many optimization algorithms can be used to conduct the minimization, a simple yet effective method is the gradient descent.[20] Basically, gradient descent updates the model parameters along with the directions of the negative derivatives of the loss function with respect to the parameters. Which can be formed as

$$\mathbf{W}^{t+1} = \mathbf{W}^t - \gamma \frac{\partial L}{\partial \mathbf{W}^t} \tag{2.8}$$

$$\mathbf{b}^{t+1} = \mathbf{b}^t - \gamma \frac{\partial L}{\partial \mathbf{b}^t} \tag{2.9}$$

Here, γ is the learning rate or step size that controls the update magnitude at step t.

The performance of the NN is highly dependent on the hyperparameters. For example, the number of hidden layers, the number of nodes at each layer, the activation functions, as well as the parameters in the optimization methods. Thus, choosing appropriate hyperparameters is crucial to building a reasonable ML model. One way to select the hyperparameters is through a validation set. The whole dataset can be divided into training, validation, and test set. The training set is combined with optimization methods to find the optimal weights and biases of a NN. The validation set is used to find the best hyperparameters. Then, we evaluate the performance of a NN via the test set.

2.2.3 Graphical Neural Networks for ML Potential

ACSF is a type of handcrafted descriptor to represent atomic configurations. One needs to choose the parameters in the symmetry functions before generating the features. Also, ACSF has a combinatorial scaling with the number of elements in an atomistic system, which leads to a super sparse high-dimensional vector when the number of elements becomes large. Recently, several models based on the graphical neural network (GNN) were proposed to address these issues. For example, the crystal graph convolutional neural networks (CGCNN) proposed by Xie et al. in 2018,[21] the graph neural network force field (GNNFF) proposed by Mailoa et al. in 2021,[22] and the geometric message passing neural network (GemNet) proposed by Klicpera et al. in 2021.[23] One advantage of these GNN based models is that they can automatically learn the embeddings for the atoms during the training process without any explicit functions as the descriptors. In addition, they have better scaling to the number of elements than ACSFs.

2.2.4 Implementation and Softwares

Wide applications of the neural network-based ML potentials require well-established and easy-to-use packages and software. With the development of many deep learning frameworks, such as Tensorflow and Pytorch,[24,25] we can implement a NN model without caring about differentiation, optimization, and other detailed operations. Several high-level NN potential packages have been released in the past few years. For example, PiNN is a library with the implementations of a high-performing graph convolutional neural network variant, PiNet.[26] SchNetPack includes the implementations of the (weighted) ACSFs and the deep tensor neural network SchNet.[27] AMP and RuNNer also contain the implementations of the BPNN.[28,29] With these packages, we can employ ML potentials into our simulations even without explicitly implementing a NN.

2.3 Kernel Methods-based ML Potentials

The kernel method is another kind of basic and popular machine learning technique.[30] The basic idea under the kernel method is to learn a better decision boundary by projecting data points into high-dimensional space, but without ever computing the explicit coordinates in that high-dimensional space. The operation of the data in the high-dimensional feature space is usually supported by a nonlinear kernel function and the inner product of pairs of data in the original feature space, which is much cheaper than computing the inner product directly in the high-dimensional space. An important application of the kernel method in machine learning potential is the Gaussian approximation potential (GAP),[31,32] which was developed based on the Gaussian process regression method.[33] It has a general form of[31,32,34]

$$E_i = \sum_{j=1}^{N_{train}} \alpha_j K\left(\mathbf{d}_i, \mathbf{d}_j\right) \tag{2.10}$$

where E_i is the atomic energy of atom i whose local environment is described by a descriptor $\mathbf{d}_i$. K is a kernel function that measures the similarity between two atomic structures. α represents the coefficients which are determined by the reference atomic energies in the training set and the covariance matrix of the training data. In other words, the energy prediction of an atomic environment is a weighted average of the atomic energies in a reference dataset, and the weights are determined by the similarities between the test atomic structure and the reference atomic configurations.

Several descriptors can be used to represent the atomic environments. For example, the bispectrum of the neighbor is used as the descriptor in the original GAP work.[31] Another descriptor that is widely used for inorganic materials is the smooth overlap of atomic positions (SOAP).[35] In GAP, a common choice for the kernel function is the squared exponential kernel[34]

$$K_{SE}\left(\mathbf{d},\mathbf{d'}\right) = exp\left(-\sum_{j}\frac{(d-d')}{l_j}\right) \tag{2.11}$$

where l_j is the length scale parameter for dimension j of the descriptor.

Compared to the ML potentials based on NN, GAP not only provides an accurate prediction but also estimates the uncertainty of the predictions. However, GAP has a high scaling $O\left(n^3\right)$ to the number of the data points in the reference dataset, which hinders its wide application on large datasets.

3 Simulating segregation in a ternary Cu-Pd-Au alloy

The main results in this chapter are adopted from our previously published paper Ref. 36.

3.1 Introduction

Our first application of the ML potentials is to investigate the surface segregation phenomenon in a ternary alloy CuPdAu. Here, we briefly introduce the conventional experimental and theoretical methods to study alloy surface segregation. Then, we demonstrate how ML potentials help to conduct more detailed and efficient simulations on CuPdAu alloy.

Several experimental and theoretical methods have been developed to study surface segregation. Most experimental approaches measure the surface concentration of alloy membranes using low-energy ion scattering spectroscopy (LEIS) and X-ray photoelectron spectroscopy (XPS).[37,38] One shortcoming is that at each measurement, only one bulk concentration is investigated. Although high throughput methods have been used in some surface segregation experiments, the usage is still limited.[39–42]

With the limitations of experimental methods and the development of the computational capacity, the demand for the theoretical study of surface segregation is growing constantly. Some simulation methods such as molecular dynamics[43] and Monte Carlo[44] simulation have been adopted in the study of surface segregation. For higher accuracy, density functional theory (DFT) may also be applied to evaluate the potential energies of the alloy configurations during the simulation process.[45] In general, simulations based on (semi)empirical methods such as embedded atom method (EAM) can simulate the surface segregation on large atomic systems but typically with lower accuracy, while simulations using DFT calculations are usually limited to small slabs although it has higher accuracy. Furthermore, most of these theoretical methods deal with bimetallic alloys, only a few simulations involve ternary alloys, and those use empirical potentials[46] or small slabs.[47]

As mentioned in Chapter 2, ML potentials[34] have become a hot topic in the atomic simulation field. They can be trained with a set of DFT potential energies and be used to predict the potential energies of new atomistic configurations. The appropriately trained ML potentials are much more computationally efficient than DFT while having similar accuracy with DFT under certain conditions.[48] The effectiveness of the ML models (especially the NN) in the construction of the potential energy surface for the alloys has been demonstrated in systems such as PdAu alloys[49] and SiLi amorphous alloys.[50] Despite many examples of NN used for bimetallic alloys, its applications to ternary alloys are relatively rare.[51] There is an example of using a NN to study the surface properties of AuPd nanoalloy in aqueous solvents, which involves four different elements,[52] where the NN also performed well.

In this chapter, we specifically utilized BPNN (introduced in Chapter 2) to develop a NN potential to simulate segregation in a ternary Cu-Pd-Au alloy across composition space. We generated a dataset including 5278 DFT calculations to train and validate the NN. The training samples were shown to cover the G^2 fingerprint space for the CuPdAu ternary slab. The trained NN was then used with Monte Carlo simulation on a $10 \times 10 \times 15$ FCC(111) slab to model the surface segregation of this ternary alloy. The predicted surface concentrations were compared with the experimental results and the discrepancy between them was substantially discussed.

3.2 Methods

3.2.1 Density Functional Theory

The Vienna Ab initio Simulation Package (VASP)[53,54] was used to conduct the DFT calculations in which the wavefunctions were represented by projector augmented wave (PAW) method.[55,56] The Perdew-Burke-Ernzerhof generalized gradient approximation (GGA-PBE)[57,58] was chosen as the exchange-correlation functional. The density of k-points in the Monkhorst-Pack mesh[59] was approximately 5 per reciprocal Angstrom and the plane-wave energy cutoff was 400 eV. The convergence error from the chosen parameters mentioned above was around 2 meV/atom.

3.2.2 Experimental Data

The compared experimental data was taken from Ref. 60, in which composition spread alloy films (CSAFs) were used as a high-throughput method to accelerate the experimental process. Bulk composition and surface composition were characterized by energy-dispersive X-ray (EDX) and low-energy He$^+$ ion scattering (LEIS) respectively. More details of the experimental conditions can be found in the original paper.[60]

3.2.3 Neural Network

The BPNN framework[18] was used to predict the total potential energy of the surface slabs, of which the configurations were represented by the fingerprints of radial G^2 symmetry functions[19] with η s of 0.05, 4 and 20. The cutoff radius was set as 6 Å to bound the range of the local environment around every atom. This value was reported with good results in alloys involving Pd, Cu, and Au.[49,61] The NN framework in this chapter contains three independent NNs one for each element: Pd, Au and Cu. Each NN has the same structure of two hidden layers with ten neurons per layer. Thus, there were in total 221 parameters in each NN. The training and evaluation process of the NNs on the CuPdAu surface slabs were conducted using the atomistic machine learning package (AMP)[28] in combination with the atomic simulation environment (ASE) package.[62,63]

3.2.4 Training Samples across the G^2 Fingerprints Space

In this work, 5278 FCC(111) slabs were generated to train, validate and evaluate the neural network. Six lattice constants ranging from 3.637 Å (of pure Cu) to 4.174 Å (of pure Au) and FCC(111) slabs of $1 \times 1 \times 7$, $2 \times 1 \times 7$, $\sqrt{3} \times \sqrt{3} \times 7$, $\sqrt{7} \times \sqrt{7} \times 5$ and $3 \times 3 \times 5$ were used to generate these slabs. In total, 5100 slabs were generated randomly as the training and validation dataset, with a split ratio of 9:1. Furthermore, to evaluate the generalization ability of the NN, 178 slabs of $\sqrt{12} \times \sqrt{12} \times 5$ were then randomly selected as the test set. Compared to the number of all possible configurations, which is more than 3^{45} (45 is the number of atoms in the slab $3 \times 3 \times 5$), the size of the training and the validation set is small. The diversity of the training samples in the fingerprint space is analyzed in the following discussion.

Training samples spanning the fingerprint space are required to enable the NN to make accurate predictions on various atomic local environments in the Monte Carlo simulation process across the composition space. For the ternary alloy CuPdAu, the G^2 fingerprints calculated by three different ηs are in a 9-dimensional space which is hard to visualize and sample. It turns out that features indicating the same surrounding element are highly correlated. Thus, the training samples can be visualized and selected in a 3D space made up of the fingerprints calculated with a certain η.

In the system of FCC(111) surface slabs of the CuPdAu, the fingerprint space generated by a certain η and a cutoff radius of 6 Å contains several triangular planes in the 3D space. These planes are characterized by different lattice constants of the slab and different locations of the atoms in the slab (surface, subsurface, and bulk). Figure 3.1 shows the G^2 fingerprints calculated with η of 0.05 for CuPdAu FCC(111) slabs with different lattice constants and structures. According to the formula of the G^2 symmetry function, the lattice constant of the slab and the position of the atom in the slab determine the plane of the atomic fingerprints, while the composition of the atomic local environment determines the location of the fingerprints in the triangular planes. Quantitative analysis of the density of the points on the triangular planes can be found in Appendix A.

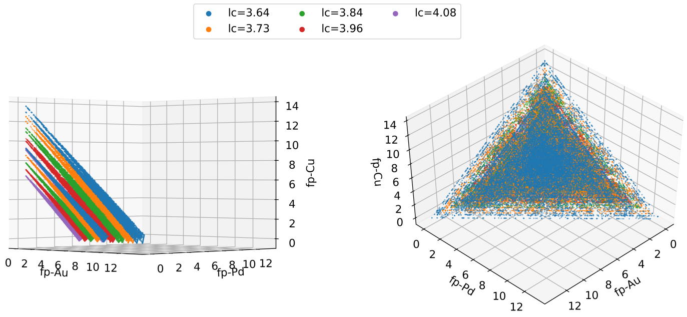

Figure 3.1: Fingerprints calculated by $\eta = 0.05$ for surface slabs with different structures ($1 \times 1 \times 7$, $2 \times 1 \times 7$, $\sqrt{3} \times \sqrt{3} \times 7$, $\sqrt{7} \times \sqrt{7} \times 5$) and lattice constants (3.61 Å, 3.73 Å, 3.84 Å, 3.96 Å and 4.08 Å). The x, y, z axes are the fingerprint values for different surrounding elements in the atomic local environment bounded by the cutoff radius. In the left figure, fingerprints of the same lattice constant are located in three planes, which represents the bulk, subsurface and surface environments from top to bottom. In the right figure, fingerprints in the same plane are distributed around a triangle, where the three angles represent the local environments purely consist of three different elements.

The fingerprints of the bulk atoms in the training set are illustrated in Figure 3.2 which shows that the fingerprints of the dataset almost cover the whole triangular plane. Combined with different lattice constants that lead to different triangular planes in the 3D space, the fingerprints of generated slabs have covered the whole fingerprint space of the CuPdAu FCC(111) slabs.

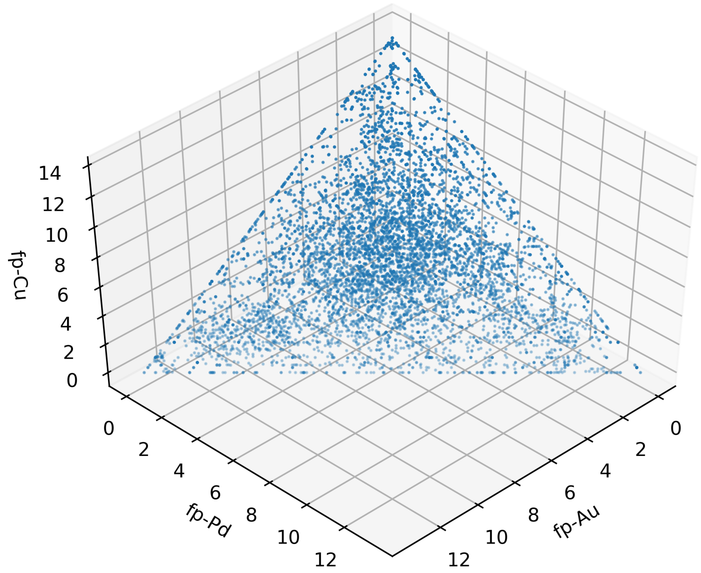

Figure 3.2: Fingerprints of bulk atoms in the training slabs with lattice constant of 3.64 Å. The fingerprints in the figure were calculated with the η of 0.05.

3.2.5 Monte Carlo Simulation

The Monte Carlo (MC) simulations were conducted on the slab FCC(111) $10 \times 10 \times 15$ whose lattice constant was estimated by the overall composition with Vegard's law.[64] The potential energies of the generated configurations were evaluated by the trained NN. In the Monte Carlo simulation, a new configuration was accepted if the change in energy is negative. Otherwise, the new configuration was accepted with the Boltzmann probability

$$p = \exp \frac{-\Delta E}{k_B T} \tag{3.1}$$

where ΔE is the change of the potential energy after an atomic swap.

3.3 Results

3.3.1 Performance of Neural Network

We first show that the neural network was well trained with the 4590 selected FCC(111) slabs. Figure 3.3 (a) illustrates the performance of the NN on the training set and the validation set. Both training and validation sets have a mean absolute error (MAE) of around 2 meV/atom. The residual error distribution of the validation set is pretty similar to that of the training set, which means that there was no apparent overfitting during the training process. The generalization ability of the NN was assessed by its performance on a larger $\sqrt{12} \times \sqrt{12} \times 5$ slab. In the larger unit cell of this slab, there can be a broader range of atomic local environments which are different from the training and validation set. The performance of the NN on the generalization test set is shown in Figure 3.3 (b). Although there is some bias (non-zero mean residual error) on the NN predictions, the size of the MAE is still comparable with that of the training and validation set. The bounded error of the NN on the generalization set supports the analysis of the fingerprints that the training set has covered almost the whole fingerprint space. Therefore, we are confident that the trained NN also has the ability to predict the potential energies for larger slabs in the Monte Carlo simulation.

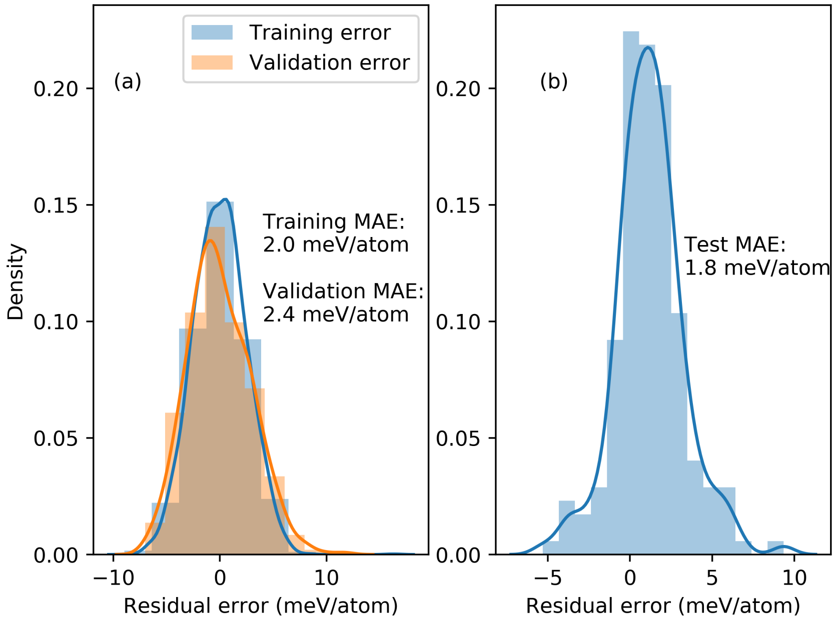

Figure 3.3: Distribution of the NN residual error on the training, validation and test set. Figure (a) shows the MAE and the residual error of the training and the validation set which share the same distribution of the slab configurations. Figure (b) shows the MAE and the residual error of the test set which contains slabs with larger unit cell than the training data.

3.3.2 Monte Carlo Simulation Results

With the trained NN, 24 bulk concentrations across the whole ternary diagram were selected to conduct the MC simulations at a temperature of 600 K. There were a total 15,000 successful MC steps (atom swaps) on each bulk concentration and the last 6000 steps were used to calculate the average surface concentration. Figure 3.4 shows a representative MC process on a bulk concentration of 20:23:47 for Pd:Cu:Au.

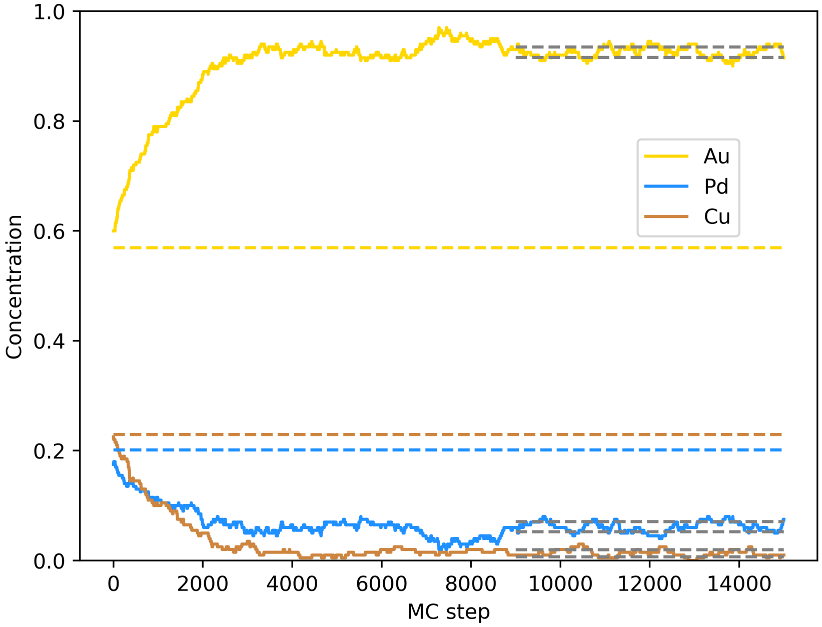

Figure 3.4: MC trajectory of the top layer concentration on a 10 × 10 × 15 FCC(111) slab with bulk concentrations of 20:23:47 for Pd:Cu:Au over 15000 successful steps. The colored dashed lines are the bulk concentrations while the solid lines are the surface concentrations. The last 6000 steps were used to calculated the average surface concentration and the standard deviation which represented by the grey dashed lines.

The MC simulation results for 24 bulk concentrations at a temperature of 600 K are shown in Figure 3.5. The values in the figures were calculated by subtracting bulk concentrations from the surface concentrations for each element. Therefore, positive values mean segregation to surface (indicating an excess at the surface compared to the bulk) while negative values represent depletion from surface. In the simulation results, Pd is depleted from the surface at most bulk concentrations except the area near the PdCu binary alloy, where Pd segregates to the surface. For Cu, it is depleted from the surface at all bulk concentrations, especially near the CuAu area. In contrast, Au segregates to the surface at all bulk concentrations. The simulated segregation tendencies are partially consistent with the reported simulation results using a cluster expansion.[65] Au is qualitatively observed to segregate to the surface at all compositions. There is an inconsistent segregation trend for Pd and Cu in their binary alloy compositions. Simulation predicts the segregation of Pd to the surface whereas Pd is depleted from the surface in experiments. The second discrepancy is that the segregation ability of Cu is underestimated in the simulations and it is overestimated for Au.

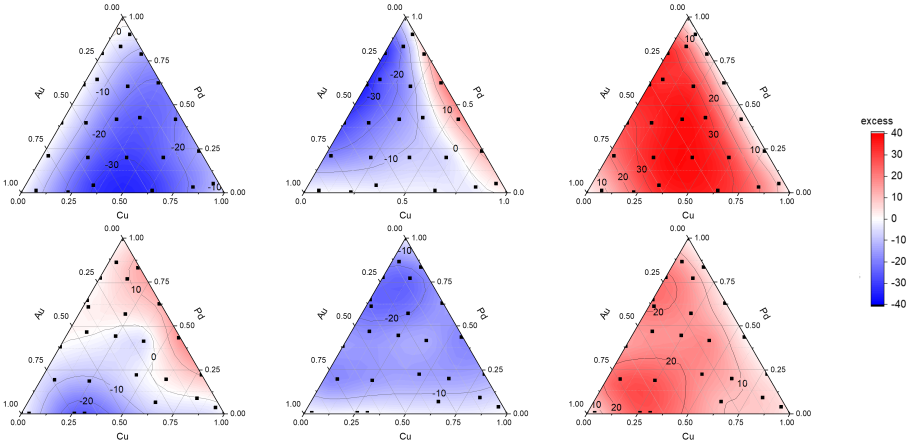

Figure 3.5: MC simulation results for 24 bulk concentrations at 600 K. These plots show the excess surface composition compared to the bulk composition for Cu, Pd and Au. The first row is for the simulation results, and the second row shows the experimental results.[60]

To evaluate the segregation ability of Pd, Cu and Au more quantitatively, the segregation energies for the binary combinations were calculated assuming each pair of two elements in equilibrium. More specifically, the reaction energies for the following three reactions were evaluated based on the simulation and experimental results:

$$Au@Bulk + Pd@Surf \overset{K_{Au-Pd}}{\rightleftharpoons} Pd@Bulk + Au@Surf \quad \Delta G_{Au-Pd} = -RTlnK_{Au-Pd}$$
$$Au@Bulk + Cu@Surf \overset{K_{Au-Cu}}{\rightleftharpoons} Cu@Bulk + Au@Surf \quad \Delta G_{Au-Cu} = -RTlnK_{Au-Cu}$$
$$Pd@Bulk + Cu@Surf \overset{K_{Pd-Cu}}{\rightleftharpoons} Cu@Bulk + Pd@Surf \quad \Delta G_{Pd-Cu} = -RTlnK_{Pd-Cu}$$

These reactions are for the segregation of a certain element against another element from left to right. K_i represents the equilibrium constant for reaction i. For example, K_{A-B} is defined as $c(B@Bulk)c(A@Surf)/c(A@Bulk)c(B@Surf)$. The reaction energy ΔG_{A-B} can be regarded as the relative segregation ability of A over B. A negative value means the preferential segregation of A over B. Figure 3.6 shows the segregation energies of Au-Cu, Au-Pd and Pd-Cu pairs. These data points have excluded the samples with bulk composition less than 0.2 and surface composition of 0 for the involved elements. Similar to the qualitative results above, the segregation energy for the Au-Pd pair based on the simulation is close to the experimental data (on parity), and a negative sign means that Au tends to segregate to the surface compared to Pd in this ternary system. The segregation energy for Au-Cu in the simulation is more negative than the experimental results, which implies that the segregation tendency of Au over Cu is overestimated in the simulations. For Pd-Cu, the simulated segregation energy has the opposite sign compared to the experimental data, which denotes an opposite segregation trend for these two elements. These discrepancies between the simulations and experiments shall be discussed in the rest of this chapter.

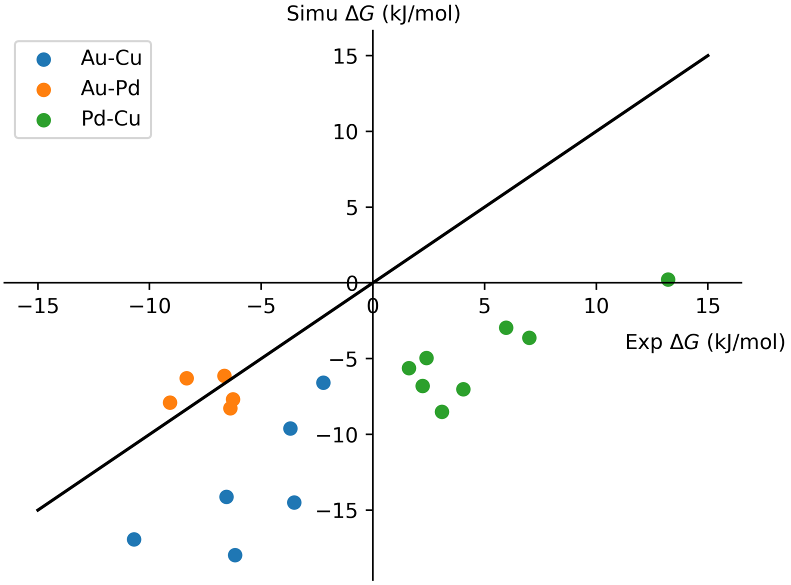

Figure 3.6: Segregation energies for Au-Cu, Au-Pd, Pd-Cu pairs derived from simulation and experimental data. Each dot corresponds to a different bulk composition.

3.3.3 Phase Separation in Experiments

Before exploring any possible calculation factors that may be responsible for the discrepancies, we note that the MC method that we used is limited to simulation of one type of crystal system (FCC in our case). It has been reported that there exists a B2 phase in the phase diagram of the CuPd, which is not considered during the simulations in this work. This phenomenon actually affects the segregation profile of the alloy.[60] According to the experimental observations, the segregation of Cu over Pd is inhibited with the appearance of the B2 phase, and the affected bulk composition ranges from $0.4 < x_{Cu} < 0.7$. The Cu segregation is only reduced, however, it does not invert, so this is not likely to be an explanation for the discrepancy that we see in the simulations.

3.3.4 The Role of Surface Relaxation

Another possible reason for the discrepancy between the simulation and experiment is the effect of surface relaxation. Previous work[49] reported that relaxation energies were not important in capturing segregation trends in Pd-Au because they largely cancel, and so we did not consider them here. It is possible that the larger size difference between Cu, Pd, and Au could be more relevant here.

To determine the impact of surface relaxation on these results, a $2 \times 2 \times 5$ FCC(111) slab was used to compare the potential energy before and after surface relaxation. The bottom three layers of the slab were set with the concentration of 1:1:1 for Au:Pd:Cu, and the lattice constant was fixed according to Vegard's law.[66] 202 unique energy configurations were selected to be evaluated. Since the potential energy difference (ΔE) before and after atoms swap matters in the Monte Carlo simulation, we investigated this quantity in the $2 \times 2 \times 5$ slab. Among 202 unique energy configurations, the ΔEs of 285 atom swaps in the top two layers were evaluated by NN and DFT. The ΔE calculated by the NN without surface relaxation, by DFT with surface relaxation, and their pairwise difference $\Delta\Delta E$ are shown in Figure 3.7. While the ΔE of every atom swap ranges from 0 to 1 eV, the difference of the ΔE calculated by NN without relaxation and DFT with relaxation is less than 0.2 eV, and most of them are less than 0.1 eV. More systematically, Figure 3.8 compares the ΔEs of atomic swaps that occur between two layers and within one layer. For the atomic swaps that occur between two layers, the ΔEs calculated by NN and DFT are always the same sign. Thus we should observe the same segregation trend with and without relaxation. For the atomic swaps within one layer, some inconsistencies take place in the ΔEs near the zero point where the magnitude of the ΔEs is quite small. These kinds of swaps do not influence the surface concentration directly. These results illustrate that the role of surface relaxation is limited in the Monte Carlo simulation through the error cancellation before and after atoms swap. This error cancellation phenomenon also appeared in the AuPd binary alloy.[49] We conclude that the neglect of relaxation on FCC(111) surface is not a likely factor in explaining the discrepancies observed.

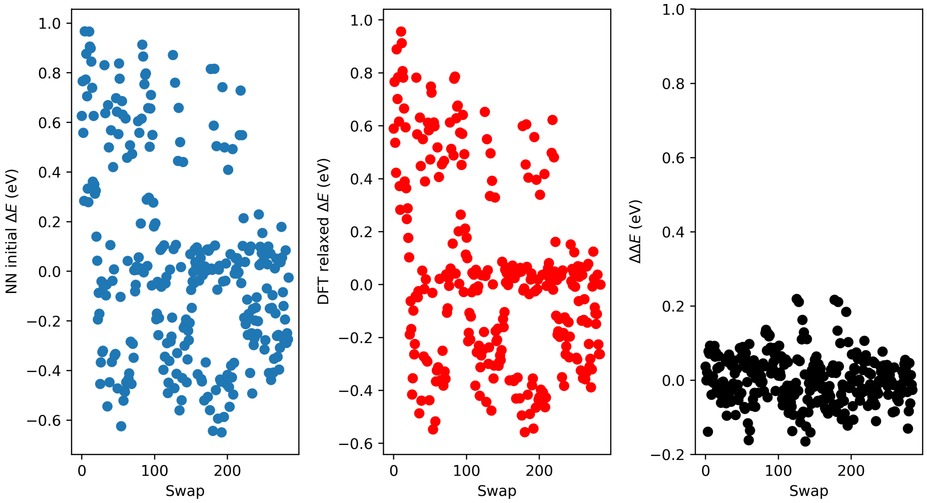

Figure 3.7: Potential energy difference before and after atomic swaps calculated by NN without surface relaxation (left), DFT with surface relaxation (middle), as well as their pairwise difference (right).

3.3.5 Vibrational Contribution to the Surface Segregation

In addition to the phase separation and the surface relaxation, another possible reason for the discrepancy between the measured and predicted segregation of Pd and Cu is the neglect of vibrational contributions to the Helmholtz free energy, which was reported to be useful to get more accurate Cu-Pd phase behavior.[67] In the MC simulations above, one assumption is that the Helmholtz free energy can be approximated by potential energy and configurational entropy. We did not include vibrational contributions to the change in energy. To take it into account, the Helmholtz free energy

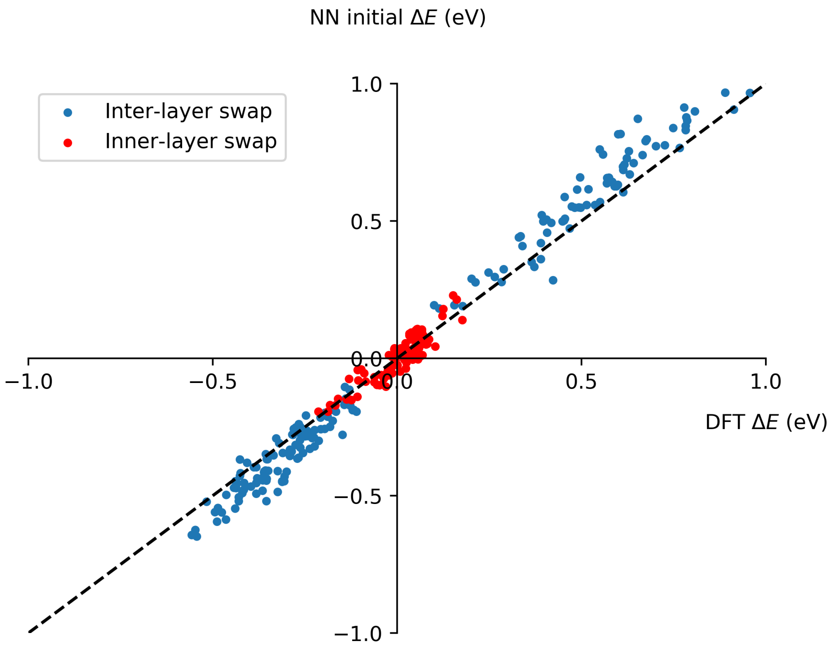

Figure 3.8: Parity plot of the potential energy change before and after atomic swap calculated by NN without surface relaxation and DFT with surface relaxation.

including the vibrational contribution was investigated on a 1 × 1 × 7 slab at 1000 K. All the atoms in the unit cell were free to relax and the final forces on every atom were less than 0.02 eV/Å^2. The vibrational frequencies were calculated by a finite difference method with displacement of 0.01 Å. Table 3.1 summarizes the potential energy and Helmholtz free energy change for segregation of Pd in Cu and Cu in Pd. We found that the vibrational energy contributions are too small to result in a change in sign of the segregation energies, and thus are not likely the source of the discrepancy.

Table 3.1: Potential energy and Helmholtz free energy change for segregation of Pd in Cu and Cu in Pd.

Segregation type	Cu_3PdCu_3 to $PdCu_6$	Pd_3CuPd_3 to $CuPd_6$
DFT Potential change(eV)	-0.014	0.146
Helmholtz free energy change(eV)	-0.063	0.168

3.3.6 Orientation dependence of CuPd surface segregation mode

After investigating the roles of surface relaxation and vibrational contributions in surface segregation of the CuPd FCC(111) surface, a remaining reason could be the orientation dependency of segregation. The experimental surfaces are not single-crystal FCC(111) surfaces; they are polycrystalline. In a related study on a Cu-Pd thin film it was found the surface was FCC(111) textured over a broad range of composition, but in the B2 range a seemingly random distribution of surfaces were observed.[39] Thus, it is likely in this work that there may be other surfaces such as FCC(110) and FCC(100) present, including grain boundaries between these surfaces.

It was reported that CuPd shows different segregation behavior on different FCC surfaces such as (111), (110) and (100).[68] We compare the DFT segregation energies for CuPd FCC(110) and FCC(111) on a $3 \times 3 \times 6$ slab in Table 3.2. The inclusion of surface relaxation did not change the sign of the surface segregation energy for FCC(111). On FCC(111), Pd tends to segregate on the surface while Cu tends to diffuse inward the bulk. However, for FCC(110), surface relaxation is significant and it can change the sign of the surface segregation energy. Without surface relaxation, the DFT calculation shows the same segregation trend as on the FCC(111) for Cu and Pd. When we take the surface relaxation into account, the surface segregation trend is reversed. This calculation means that the surface segregation of CuPd depends on the orientation of the slab, which is consistent with the recent experimental work where the surface segregation behavior of Cu and Pd is related to the surface environment.[68]

To study the details of the surface segregation behavior of CuPd on FCC(110), we performed another set of MC simulations on this orientation. As illustrated in Table 3.2, surface relaxation plays an significant role for

Table 3.2: CuPd segregation energy on FCC(111) and FCC(110).

Surface	Seg. Type	Surf. Relax	Seg. Energy (eV)
FCC(111)	Cu in Pd	No	0.096
FCC(111)	Pd in Cu	No	-0.066
FCC(111)	Cu in Pd	Yes	0.051
FCC(111)	Pd in Cu	Yes	-0.122
FCC(110)	Cu in Pd	No	0.006
FCC(110)	Pd in Cu	No	-0.018
FCC(110)	Cu in Pd	Yes	-0.166
FCC(110)	Pd in Cu	Yes	0.007

surface segregation. Thus, we need to include the surface relaxation energy during the MC simulation. Similar to the way in which we modeled the total potential energy using a NN, we built another NN with the same architecture to model the surface relaxation energy using 1000 slab configurations. Only the atoms in the top two layers have a contribution to the surface relaxation energy during the training and prediction phase. Surface relaxation was considered on both sides of the slab in MC simulations. The segregation profiles for CuPd FCC(110) with and without considering the surface relaxation are shown in Figure 3.9. We could see a qualitatively different MC simulation result here compared to the FCC(111) result above. Here, CuPd(110) with surface relaxation has the same segregation trend with the experimental result, while the segregation profile without surface relaxation included is still far away from the experimental result.

In this section, we discussed how the surface orientation affects the segregation profile of CuPd. Due to the complicated surface environment in reality (e.g., step, terraces, and defects), it is not currently possible to fully simulate every detail of all the local environments on a surface. Thus, we only performed a set of MC simulations on FCC(110) to demonstrate that there can be orientation dependent surface segregation behavior of CuPd. Through the comparison between FCC(111) and FCC(110), we have shown

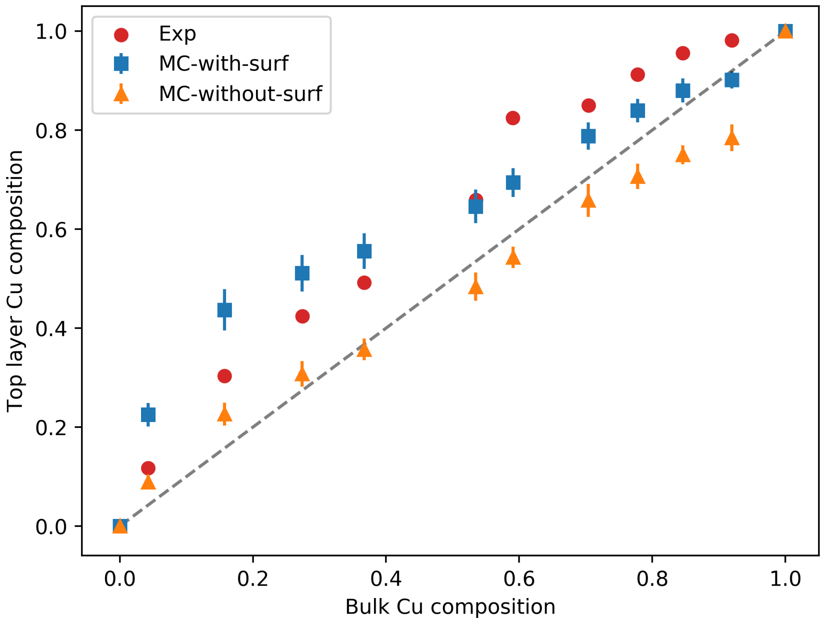

Figure 3.9: Segregation profile of CuPd(110) at 600 K. Blue squares show the MC simulation results with surface relaxation. Orange triangles show the simulation results without surface relaxation. Red points show the experimental results.[60]

that the structure of the surface has significant impact on the segregation profile of CuPd, and this is a probable explanation for the discrepancy between simulation and experiment initially presented in this work as the experimental surfaces are known to be polycrystalline.

3.3.7 DFT errors in alloy formation energies

In the MC simulation results, we also saw an overestimated segregation of Au to the surface for CuAu alloy. This might be explained by the DFT predictions of the formation energy of the alloys. Table 3.3 summarizes some reported formation energies of AuCu alloys,[67,69–72] in which DFT-PBE predicts much higher formation energies than experimental data. This means it underestimated the interaction between Cu and Au, which could lead to the overestimated surface concentration of Au as we observed in the Monte Carlo simulations. In other words, the energy cost in losing Cu-Au bonds from Au segregating to the surface is over-compensated by the reduction in the surface energy of Au at the surface. This aspect of the discrepancy can only be rectified by more accurate DFT functionals.

Table 3.3: Experimental and DFT formation energies for CuAu alloys

Alloy	Experimental (meV/atom)	DFT-PBE (meV/atom)	Ordered
Cu_3Au	-74	-44	$L1_2$
CuAu	-93	-56	$L1_0$
$CuAu_3$	-39	-25	$L1_2$

3.4 Conclusions

With 5278 FCC(111) slab configurations of CuPdAu, we built a neural network to compute FCC(111) ternary alloy slab energies and validated its performance. The training samples were selected randomly with or without some constraints on their bulk concentrations. In the fingerprints space, we showed that the training set nearly spanned the possible atomic local environments, which enabled the NN to predict the DFT potential energy of any larger slab encountered in the MC simulation with a MAE of around 2 meV/atom. The trained NN made it possible to conduct the MC simulation on a 10 × 10 × 15 slab with the first principle potential energy which is computationally unpractical using DFT directly.

Through the combination of NN and MC simulations, the surface concentrations of the slabs with various bulk concentrations were predicted and compared to experimental results. The segregation of Au on the surface was simulated successfully and the simulation results are qualitatively consistent with the experiments for the AuPd parts of the ternary alloy space. For the CuAu part, the simulation result is qualitatively consistent to the experimental data, but MC simulation overestimated the Au segregation due to the low accuracy of the DFT functional (PBE). In terms of the CuPd part, there are some discrepancies between simulated and experimentally observed segregation behavior which we ultimately attribute to limitations in the use of ideal FCC(111) surfaces as models for segregation in polycrystalline films. We discussed the orientation dependent surface segregation behavior of CuPd by comparing the simulation results on FCC(111) and FCC(110) and we showed evidence that FCC(110) would show Cu segregation behavior that is more consistent with the experimental observations.

4 Machine-learning accelerated geometry optimization in molecular simulation

The main results of this chapter are adopted from our published work in Ref. 73.

4.1 Introduction

In last chapter, we focused on simulations under vacuum conditions, which simplified many calculations. However, in reaction conditions, there are complicated interactions between the reaction molecules and the catalytic surfaces. The surface configuration and the reaction taking place on the surface have mutual effects on each other. The surface configurations (e.g., composition, atom arrangement) affect the kinetic and thermodynamic properties of the reactions. In the meanwhile, the molecules and the intermediates of the reactions also have impact on the surface configurations. Adsorption energy is one of the common quantities that represent the complex interactions between the adsorbates and the catalyst surface. Accurate adsorption energies of certain adsorbates over various kinds of catalytic surfaces are basic prerequisites to conduct high-throughput screening for novel catalyst candidates.[74–76] Many studies aim to build up a reliable machine learning model to predict the adsorption energies on different adsorption sites.[77–79] In this case, a training set (a set of adsorption configurations with their corresponding adsorption energies) covering most of the possible configurations is necessary to obtain a reasonable model which affects the reliability of the screening process.

The rate-limiting step to obtain the adsorption energies is often the geometry optimization process. This process usually consists of a sequence of iterative single point calculations with DFT. The structure update is completed by various optimizers like conjugate gradient descent or the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithms. These algorithms start with an initial guess, and then iteratively move the atoms to reduce the forces to a specified tolerance. The forces are typically computed at each step by DFT code. One path to speeding up these calculations is to use a better initial guess. An alternative approach is to use a surrogate model that is computationally cheap, but sufficiently accurate that many steps can be taken with the cheap model before an expensive DFT calculation is requested. Recently, many ML methods have been developed to accelerate the local geometry optimization process with this idea. For example, Peterson[80] used a neural network as the surrogate model to find the transition state, but the uncertainty is not included. Torres et al.[81] and Koistinen et al.[82] used Gaussian Process Regression (GPR) to estimate the uncertainty during the local geometry optimization. Those implementations of GPR are solely based on the Cartesian coordinates of the atoms, which limits the training set to the past geometries of the same configuration size and composition during the optimization. The information of other configurations can not be utilized. There are other applications of active learning in geometry optimization[61,83–86] or in molecular dynamics.[87,88] Most of these methods are also based on active learning with uncertainty measured by GPR or neural network ensemble. In active learning relaxation process, a surrogate model is trained to replace the expensive DFT calculations to perform the energy minimization steps. At each step, the uncertainty of the model prediction is monitored. If the uncertainty exceeds a specified threshold, DFT calls will be requested to get accurate energy and force information for the uncertain configuration. Then, this new data point is used to update the surrogate model.

The work to date has mostly focused on the relaxation of a single configuration, which might have limited acceleration when applied to relaxations of many configurations. For each configuration, the surrogate model essentially starts from scratch, and has no ability to share information between similar configurations. In this chapter, we illustrate and evaluate an online learning method to accelerate the local geometry optimization for multiple configurations simultaneously. More specifically, we focus on two aspects to accelerate the online learning process. The first point is related to the training of the surrogate model that used to relax the target configurations. When the training set gets large, the training of the machine learning model also takes more time, which might result in longer relaxation time than using DFT solely, although with fewer DFT calls. This issue is shared among various ML models including GPR and deep learning models. We note that using a local training dataset is sufficient to conduct the local geometry optimization. Thus, the size of the training set used to update the surrogate model at each step could be limited, which could significantly reduce the training time. The second point of this chapter is to discuss the potential methods that could be adapted to accelerate the active learning relaxation process for large number of configurations. We illustrate three adaptations to three different scenarios: relaxation from scratch, relaxation from a small dataset and relaxation from a large existing dataset. The main point under these methods is that the information of different relaxation trajectories could be shared to accelerate the overall relaxation process. Another objective of this chapter is to provide an overview about the performance of NN-based online learning on various local geometry optimization tasks.

4.2 Methods

4.2.1 ML Surrogate Model for Potential Energy Surface

We have introduced two basic types of machine learning potentials, Gaussian Approximation Potentials (GAP)[31] and Behler Parrinello Neural Networks (BPNN)[18] in Chapter 2. In the work of this chapter, we used a modified version of BPNN, the SingleNN, to model the potential energy surface.[89] SingleNN uses the same ACSFs as the conventional BPNN, but uses a single neural network with multiple output nodes for different elements, rather than a separate neural network for each element. Under the same NN structure (same number of hidden layers and same nodes in each layer), it contains fewer parameters than the BPNN framework. Thus, the training and inference cost is lower. Our SingleNN structure contains two hidden layers with 50 neurons at each layer. The activation function used is hyperbolic tangent (tanh). These hyperparameters were chosen by cross validation among different NN architectures on the dataset of previous work and they are typical for machine learned potentials. The structure of this NN looks relatively over-parameterized considering the small size of the dataset in this work (typically the dataset contains 50 configurations). This is because we want to utilize the benefits of an over-parameterized deep learning model: 1) With high probability, convergence of the training process is easy from a random initialization, and 2) an over-parameterized NN could lead to less correlated models with high probability only using different random initializations,[90,91] such that we could build up a diverse NN ensemble from different random initializations. Empirically, there are also some successful applications of large NN models on small dataset with reasonable generalization ability.[92] Since the capacity of our NN structure is relatively large, we used early stopping to prevent overfitting. Although we take the early stopping to prevent the overfitting, for our specific application, we need to note that the validity of the relaxed structure from the active learning method does not depend on whether there is an overfitting issue during the training stage because 1) the surrogate model is only used when uncertainty is low, 2) if the uncertainty exceeds a threshold the data is augmented by new DFT data, and 3) the final minimum is always validated by DFT.

The NN structure used to predict the atomic energy is shown in Figure 4.1. Mathematically, the atomic energy, total energy and forces predicted by the SingleNN could be formulated by Equations 4.1 - 4.3.

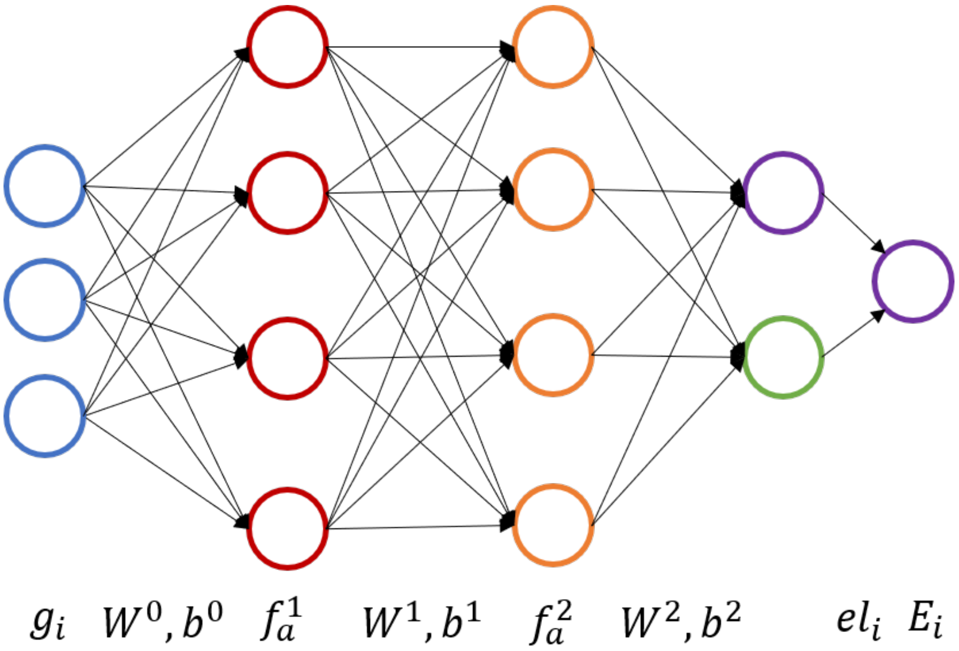

Figure 4.1: SingleNN structure for atomic energy prediction. The use of each variable is shown in Equations 4.1 - 4.3 and described in the text.

$$E_i = \left[\mathbf{W}^{(2)} f_a^{(2)} \left(\mathbf{W}^{(1)} f_a^{(1)} \left(\mathbf{W}^{(0)} \mathbf{g}_i + \mathbf{b}^{(0)} \right) + \mathbf{b}^{(1)} \right) + \mathbf{b}^{(2)} \right]_{el_i} \tag{4.1}$$

$$E_{tot} = \sum_i^N E_i \tag{4.2}$$

$$\mathbf{f}_i = -\frac{\partial E_{tot}}{\partial \mathbf{r}_i} \tag{4.3}$$

In these equations, E_i and $\mathbf{f}_i$ are the energy and forces of atom i, $\mathbf{g}_i$ is the fingerprint vector for atom i, $f_a^{(l)}$, $\mathbf{W}^l$, $\mathbf{b}^{(l)}$ are the activation function, weight matrix and bias at layer l. The subscription el_i is a select operation that takes the entry corresponding to the element of atom i, such that the atomic energies of different elements could be learned separately. N is the number of atoms in a configuration. E_{tot} is the total energy of the configuration.

In the training stage, we applied the L-BFGS optimizer[93] on a mean square error (MSE) loss function to train the model:

$$L = \frac{1}{N} \sum_i^N (E_i - \hat{E}_i)^2 + \lambda \frac{1}{\sum_i M_i} \sum_i^N \sum_j^{M_i} (F_{ij} - \hat{F_{ij}})^2 \tag{4.4}$$

where E_i, $\hat{E_i}$ are true and predicted total energy for configuration i. F_{ij}, $\hat{F_{ij}}$ are true and predicted force of j^{th} component of configuration i. N is the number of configurations and M_i is the number of force components in configuration i. λ is a hyperparameter that specifies the relative importance of the energy loss and the force loss.

To measure the uncertainties of the model predictions, we adopted the NN ensemble method as an approximate estimation.[91] We use 10 NNs in the NN ensemble and each NN has the same structure. As mentioned in the original ensemble method paper, each NN is trained on the same training set without bootstrapping but with different random initialization. This is because different initializations are already able to generate different NN models using the same training set because of over-parameterization.[90]

The prediction uncertainty is estimated by the variance of the model predictions in the ensemble. We used a multiple of the maximum variance in the training set as a criterion to determine if a configuration is uncertain or not. More specifically, Equation 4.5 quantifies this uncertainty threshold,

$$T = \alpha \max_i \mathrm{Var}\left[E_{tot}^i\right] \tag{4.5}$$

where α is the coefficient to control the extent to believe the prediction of the NN ensemble. $\mathrm{Var}\left[E_{tot}^i\right]$ is the prediction variance of the NN ensemble on the total energy of a configuration i in the training set. T is the threshold above which a prediction is considered as uncertain. We chose the α by comparing the performance of different values on a small dataset. For the applications below, setting alpha between 2 to 3 works for all examples and we use 2 as the default value. The intuition is that if the NN ensemble has a similar variance on a test configuration as the variance in the training set, then we could expect the test configuration is close to the region of the training dataset. Thereby, we could expect similar error to the training error. If it is much larger than the maximum variance in the training set, it is probable that extrapolation is occurring, and we should be careful about the prediction. This intuition is shared by different machine learning models like GPR and NN ensemble. For example, Figure 4.2 shows the GPR and NN models for the Lennard Jones potential.[94] Both models have small prediction variance in the region of the training data. As the test data goes far away from the training set, the prediction error and variance also increase.

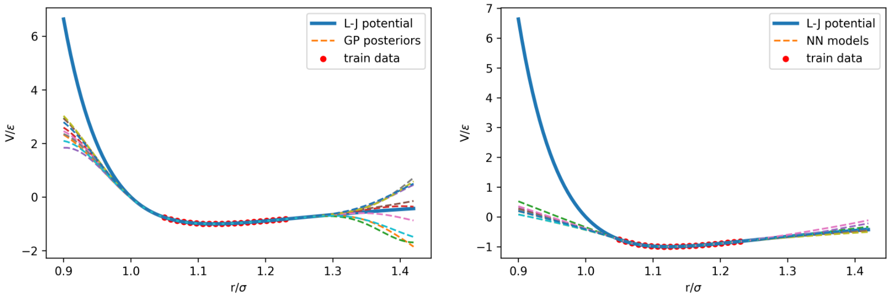

Figure 4.2: Surrogate machine learning models for the Lennard Jones potential. Left plot shows the GPR while the right plot shows the NN ensemble. Both models have low prediction variance in the region of training set and high variance for the data that is far from the training set.

We also compare this NN model with the GPR model in one of our datasets. The details of the GPR formula are attached in Appendix D. Optimization of the hyperparameters like the bandwidth and the data noise term was conducted according to the previous literature reports.[81,82] The data noise in this application could be the DFT convergence error related to the factors like k points and cutoff energy.

4.2.2 Relaxation with Active Learning

The framework of active learning for relaxation is shown in Figure 4.3 which is similar to most active learning frameworks,[85,95] but we process multiple configurations simultaneously to obtain extra acceleration. The rationality of pooling different trajectories together is that the information of similar atomic environment across trajectories could be shared by a common atomic NN surrogate model, which was also observed in a water NN potential.[96] Another benefit of the pooling is that it could be applied in a scalable way. Different configurations could share a common surrogate model and there is no need to assign separate computing resources for training of each trajectory. For the specific procedure, we start from N configurations to be relaxed, build a common NN ensemble for these N configurations. At each step, we conduct relaxation until the model becomes uncertain for each configuration. Then we query DFT for the true energies and forces for these uncertain configurations, which are used to update the surrogate model. During the relaxation process, we limit the size of the training set and keep the configurations of the most recent steps; all previous configurations are discarded in the iterative training of the NN ensemble. This setting is used to reduce the time to train a NN when the available data points grows as the relaxation approaches. Intuitively, this modification is similar to L-BFGS compared to BFGS, which estimates the inverse of the Hessian matrix at a point using the recent gradients instead of full history.[93] However, L-BFGS aims to alleviate the memory problem while we try to reduce the training time for the surrogate model.

Before running the online learning to relax target configurations, several cases should be considered. If no prior data related to the target configurations is available, then the initial model is built on the DFT information of the initial configurations. If there are some existing relaxation trajectories that are related to the target configurations (e.g. alloys with the same elements but different configurations), then this data is incorporated with the DFT data of the initial configurations to set up the initial NN model. This part of reused data also accelerates the overall process of relaxation. Finally, if much training data is available from previous relaxations that are similar to the initial configurations, then it is possible to conduct the relaxation in a offline way using the NN model trained on the prior training set without initially accessing the DFT calculation.

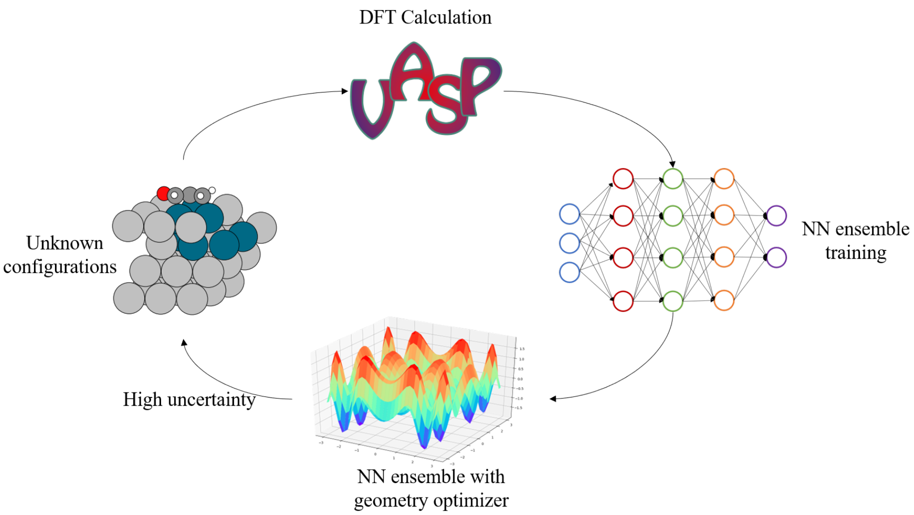

Figure 4.3: Framework for relaxation with online active learning. The overall workflow starts with the initial configurations that needs to be relaxed. At first, the DFT energies and forces are calculated and the NN ensemble is trained with these initial information. Then the model is utilized with optimizers to reduce the energy of the configurations. The relaxation with model stops when encountering with uncertain configurations or reaching the relaxation criterion. The uncertain configurations are submitted for further DFT calculations.

4.2.3 Application Dataset

In the following sections of this chapter, we test the proposed online learning methods on a variety of structures including bare pure metal slabs, bare metal alloy slabs, slabs with an adsorbate, and a nanoparticle with an adsorbate. These structures increase in complexity, and are expected to be increasingly expensive to do geometry optimization with. More specifically, we take Au FCC(100), Au FCC(111), Au FCC(211), Au FCC(643), Au FCC(111) with propylene on the surface, AuPd FCC(111), AgPd FCC(111) with acrolein on the surface, and AuPd icosahedron with CO on edge as the examples for these structures. For the slab, the bottom two layers are fixed and the remaining atoms are free to be relaxed. For nanoparticles, all atoms are free to move during the relaxation. In addition to the geometry relaxation of these structures, we also evaluate this method on two climbing-image nudged elastic band (CINEB) cases:[97] Pt heptamer rearrangement over Pt FCC(111) surface and acetylene hydrogenation over Pd FCC(111) surface. The CINEB algorithm is like a constrained geometry optimization where forces in the direction tangent to the bands are projected out. The basic framework to perform CINEB using NN ensemble is similar to the CINEB based on GPR.[81] In our work, the surrogate model is the NN ensemble instead of the GPR. During the relaxation, when one of the configurations in the CINEB is identified in the uncertain region of the NN ensemble, we query for a DFT calculation for this configuration. This process continues until all configurations are relaxed with certainty, then we query the DFT information for the configuration with highest energy until the energy and force prediction for the highest-energy configuration is certain and the true force is lower than a specified threshold.

The DFT used in this chapter is also performed by the Vienna Ab initio Simulation Package (VASP)[53,54] with Perdew-Burke-Ernzerhof generalized gradient approximation (GGA-PBE) as the exchange-correlation functional.[57,58] For the Pt heptamer rearrangement case, we used EMT (implemeted in ASE[63]) as the calculator for energy and forces, because the size of this system (unit cell with 343 Pt atoms) is too large for DFT. The related dataset, relaxation trajectory, configurations in the NEB as well as the code used to conduct the active learning geometry optimization are available in GitHub,[98] in which the code to calculate the fingerprints is modified based on the functions of SimpleNN.[99]

4.3 Results

4.3.1 Active learning for geometry optimization of single configuration

Usually, geometry optimization is performed for each configuration separately. For example, one may be interested in the relaxed geometry of an occupied adsorption site. Then, the geometry optimization would be conducted on an initial guess of the configuration. Active learning could be integrated into the optimization trajectory to accelerate the process by using a surrogate model with uncertainty. With the example of Au slabs with and without an adsorbate, we evaluated the performance of active learning on single configuration relaxation, and compare it with the quasi-Newton optimizer built in VASP (RMM-DIIS).[100] As shown in Figure 4.4, the acceleration for the bare slabs is not as significant as it is for the slab with propylene on the top. The more complex surface FCC(643) gains more acceleration than simpler surfaces FCC(100), FCC(111), and FCC(211). The results suggest that the surrogate model requires a minimum number of steps or configurations to build up a sufficient approximated potential energy surface, and then to show acceleration. These results show that with active learning the number of DFT calls may be reduced by a factor of two to four for geometry optimizations that require 20 or more relaxation steps.

4.3.2 Further acceleration by information sharing among configurations and utilizing prior data

There are multiple ways to use machine learning to accelerate geometry optimization. First one may build a surrogate machine learned model from the relaxation trajectory of a single configuration as it develops, using the

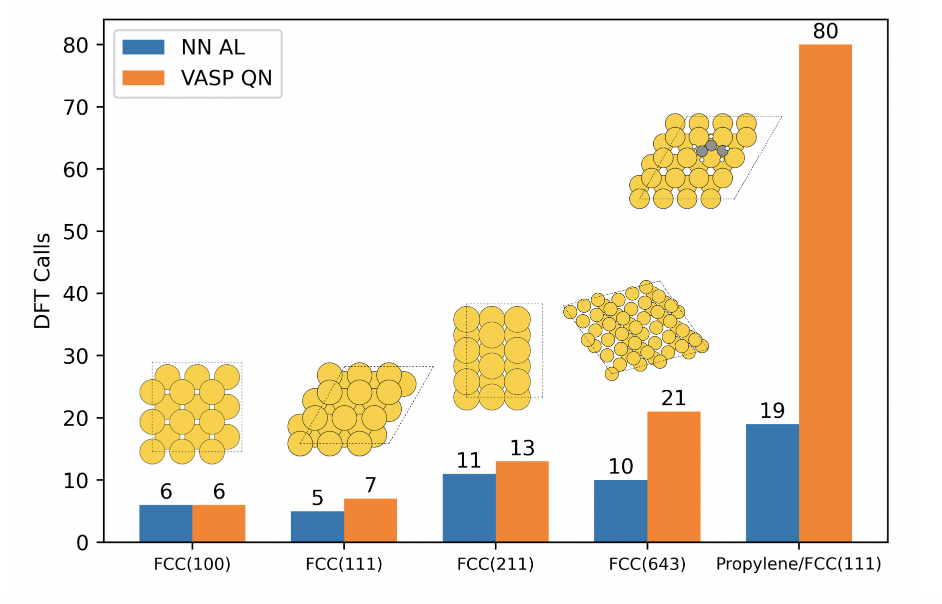

Figure 4.4: Comparison of the number of DFT calls between active learning with NN ensemble and quasi-Newton built in VASP when each configuration is relaxed independently.

surrogate model when it is sufficiently accurate. Alternatively, one can relax many (related) configurations in parallel and train a single surrogate machine learning model on the collection of developing trajectories (the multiple method). Finally, if one has access to relaxation trajectories from several previously relaxed configurations, one can pretrain a surrogate machine learning model and then use it (the warm up method).

We compare the performance of active learning with these different strategies: single configuration, multiple configurations and multiple configurations with warm up (pre-training) on the example of an adsorbed acrolein molecule on the AgPd FCC(111) system. This system is more complex than the examples in previous section with less symmetry and it is expected to take more relaxation steps to find a minimum energy geometry. Here we use the same query strategy for new DFT single point calculations, but with different settings for the initialization. For the single configuration active learning, the method only focuses on relaxing one configuration at each time. The surrogate model starts with the DFT information of the target configuration. At each relaxation step, it relaxes this configuration and queries DFT calculation for one uncertain configuration. For the multiple configurations setting, the DFT energies and forces of all target initial configurations are used to initialize a single NN model. Then all configurations are optimized until each configuration is fully relaxed or goes into uncertain region of the surrogate model. In terms of the warm up setting, it requires some prior DFT data related to the target configurations that need to be relaxed, such that the surrogate model could be pre-trained with this prior DFT information which serves as the prior beliefs for the potential energy surface.

The performance of above three methods on 13 different acrolein/AgPd configurations are shown in Figure 4.5. With standard DFT/quasi-Newton (DFT/QN) geometry optimization, it takes about 193 DFT steps on average to relax the geometries. All three methods in our work and the GPR model show acceleration, while the NN methods present better performance over the GPR model. The hyperparameters of the GPR model are referenced from previous literature reports.[81,82] We note that the hyperparameters from the reported literatures might not be the optimal for our system, but even still we observe acceleration of about four times fewer steps with the GPR, 11 times fewer steps for the single configuration, and 13 times fewer steps for the multiple configurations. The pretrained warm-up shows the largest acceleration indicating that the surrogate model is more accurate and has performed better. Clearly, the information sharing through the surrogate model accelerates the active learning relaxation process. The large reduction in the number of DFT calls required directly translates to saved time and computing resources. In the limit of a fully trained machine learned potential, one can expect no additional DFT calculations are required for a new relaxation, but in our experience and in the literature it takes thousands of DFT calculations to obtain that.

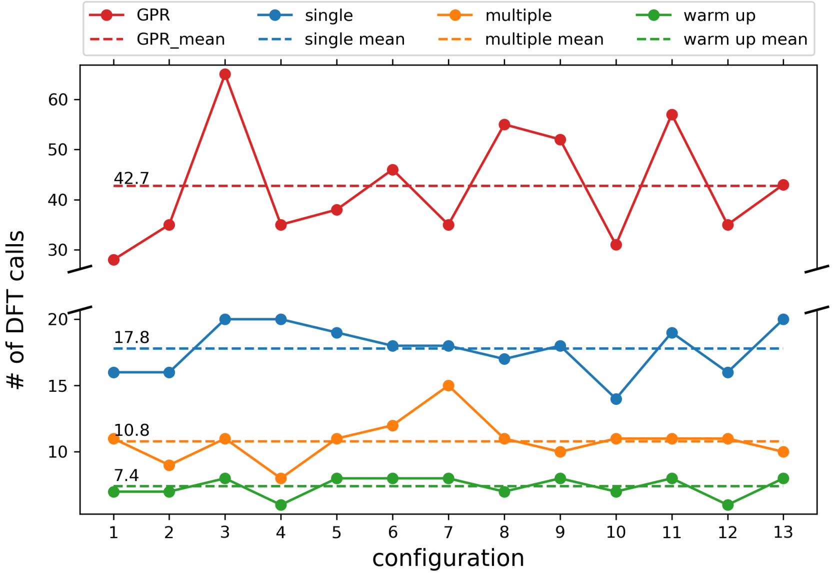

Figure 4.5: Number of DFT calls for three different active learning settings for the relaxation of acrolein/AgPd(111). The blue line represents the single configuration mode, the orange line is for the multiple configurations mode and the green line denotes the multiple configurations with warm up. The red line serves as a baseline which is the performance of GPR model implemented according to previous literatures.[81,82] For comparison, with no ML it takes about 193 DFT calls to converge.

A related scenario is when we have prior data about the target configurations that we want to relax. For example, if we have the active learning relaxation trajectories for many configurations of acrolein/AgPd and we want to relax the remaining configurations. In this case we can utilize the existing data to build up a model to approximate the PES of the acrolein and AgPd, and then conduct the relaxation process offline since it is possible that the information required to relax the remaining configurations has been included in the existing trajectories. We show the offline relaxation performance in Figure 4.6, in which 243 acrolein/AgPd relaxation trajectories are used to train a NN model. Then, another 13 configurations are relaxed using this model. Without accessing any DFT calls, the NN could reduce the maximum force of the configurations from 0.7 eV/Å to below 0.1 eV/Å, which could serve as a preprocessing step if lower forces are required. The NN ensembles provide uncertainty estimates, which would be useful for determining if the pretrained models are sufficiently accurate for new configurations that are not similar to the training set.

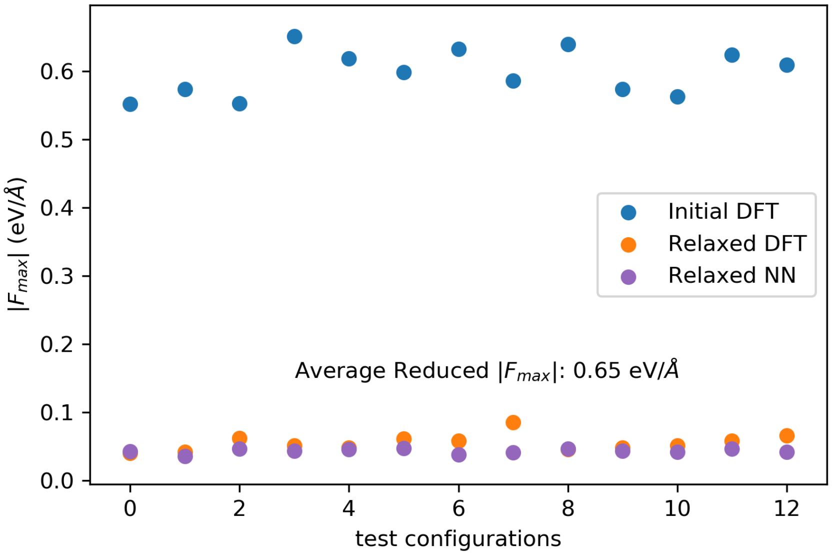

Figure 4.6: Offline relaxation on 13 acrolein/AgPd configurations using NN trained on 243 existing relaxation trajectories. Blue points show the maximum DFT forces for the initial configurations. Orange scatters are the maximum DFT forces for the NN relaxed configurations while purple dots are the NN maximum forces.

In summary, this section shows that machine learning surrogate models can be trained on the fly or in advance in a variety of ways to accelerate geometry optimization. The biggest on the fly acceleration occurs when multiple similar configurations are relaxed in parallel with shared training data in a single surrogate model. Further acceleration can be obtained if training data already exists to pre-train the surrogate model on. In the next section we show the acceleration is observed for many different atomistic systems, and the degree of acceleration is system dependent.

4.3.3 Performance of the active learning on more complex systems and nudged elastic band calculations

To explore the ability of active learning with multiple configurations to accelerate geometry optimization, we evaluate this method on three different chemical structures: bare AuPd FCC(111) slab, CO on an AuPd icosahedron nanoparticle and acrolein on AgPd FCC(111) surfaces shown in the illustration example. We measured the required DFT calls to fully relax the configurations and compared it with the built-in VASP quasi-Newton optimizer RMM-DIIS. We relaxed the configurations until the maximum force on the atoms is less than 0.05 eV/Å. The results are shown in Figure 4.7. Active learning accelerates the relaxation process to different extents across these three systems. For the simpler case like the AuPd bare slab, the acceleration ratio is about 50% compared to the original VASP optimizer. For more complicated (i.e. lower symmetry and more atomic degrees of freedom) systems, the acceleration is more significant, reducing the number of DFT calls by more than 90%. This result shows that active learning is suitable for relaxing more complicated structures. Once the NN has a reasonable representation of the PES of target configurations by calling the first several DFT calculations, this surrogate model could be used to fine-tune the structure as a replacement of DFT calls.

In addition to local geometry optimization presented in aforementioned cases, we also evaluated the NN ensemble-based active learning method in two climbing image NEB (CINEB) examples: Pt heptamer rearrangement

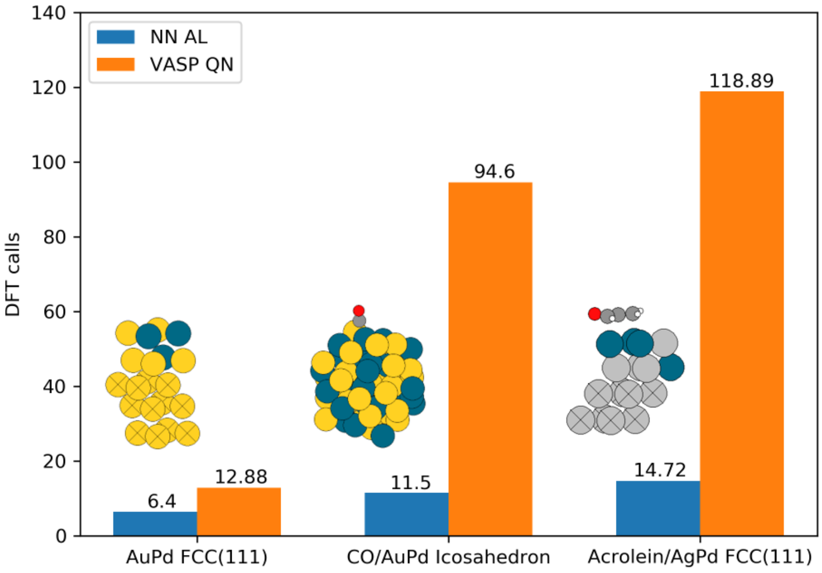

Figure 4.7: Comparison of active learning (AL) and VASP quasi-Newton (QN) method on relaxing three different structures: bare AuPd slab, CO on AuPd icosahedron and acrolein on AgPd slab.

over Pt FCC(111) surface and acetylene hydrogenation over Pd FCC(111) surface. We used an effective medium theory (EMT) calculator for the heptamer and DFT for the hydrogenation reaction.[12] We used EMT for heptamer because of the large size of the Pt slab. This example also shows that the NN ensemble method is not limited to DFT. We note that EMT is a theory with relatively lower complexity than DFT, thus, besides the EMT example, we also included a DFT example on acetylene hydrogenation. These two examples show the NN ensemble could be applied on both two theories with different complexities. The reaction curves generated by the NN ensemble with active learning and the corresponding VASP or EMT calculator are shown in Figure 4.8. With the same initial and final state, the NN ensemble found practically the same transition state as VASP or EMT for these two system. The corresponding activation energies have 6 meV and

4 meV error compared to the one from EMT or DFT which is within convergence tolerance. The required DFT or EMT calls are much fewer than those without active learning as shown in Table 4.1. In the case of acetylene hydrogenation, there are some mismatched energies between NN and VASP for the intermediate configurations except the transition state. This is caused by the intrinsic setting of the low scaling CINEB method based on active learning.[81] Only DFT data for the configuration with the highest energy is evaluated for the convergence criterion. This problem could be alleviated by modifying the convergence criterion to include the energy and forces of other images in the elastic band, such that all images in the band are fully relaxed instead of only considering the highest-energy configuration.[82] However, for the purpose of CINEB, the NN ensemble with active learning could accelerate the process to find the transition state by finding the configuration with the highest energy.

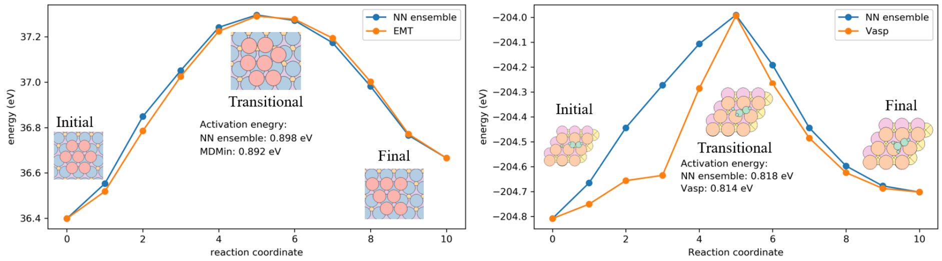

Figure 4.8: Climbing NEB curves generated by NN ensemble and (a) EMT for Pt heptamer rearrangement (b) DFT for acetylene hydrogenation over Pd FCC(111) surface.

Table 4.1: EMT or DFT calls queried by NN ensemble with active learning, EMT with MDMin and VASP with built-in quasi-Newton optimizer for Pt heptamer rearrangement and acetylene hydrogenation.

	Pt heptamer rearrangement	Acetylene Hydrogenation
	(EMT)	(VASP)
Calculator	596 calls	1109 calls
NN ensemble with AL	9 calls	30 calls

4.3.4 Limiting the training data to recent configurations for training efficiency

With the active learning approach we add training data as the geometry optimizations proceed. This also adds (re)-training time which grows as the size of the training set. In the first few steps from scratch, this is not a problem since the training process could be completed quickly because of the small size of the training set. The time cost for training is negligible compared to the DFT calculations. However, when the size of the training set grows large along with the relaxation steps, the required time to train a model with high accuracy also scales up. Figure 4.9 illustrates the training time for NN over the active learning iterations. The initial training set consists of 13 different acrolein/AgPd configurations. At each iteration, uncertain configurations are added into the training set and the surrogate model is updated. The training time scales linearly with the size of the training set, which could be time-consuming when the iterations increase.

It is not always necessary to use all of the training data however. We found that the correlation (or similarity) between two configurations in the relaxation trajectories decreases as the number of steps between them increases. The correlation between two configurations can be illustrated by averaging the Pearson correlations between corresponding atomic fingerprints in two configurations. There is usually reasonable similarity between the initial and final states (assuming a reasonable initial guess is used), so to highlight the change in similarity we subtracted the final state correlation from each configuration because the relaxation is local. The descending correlation shown in Figure 4.10 for a relaxation trajectory suggests we may only need to focus on utilizing the configurations in the most recent steps to perform local geometry relaxation.

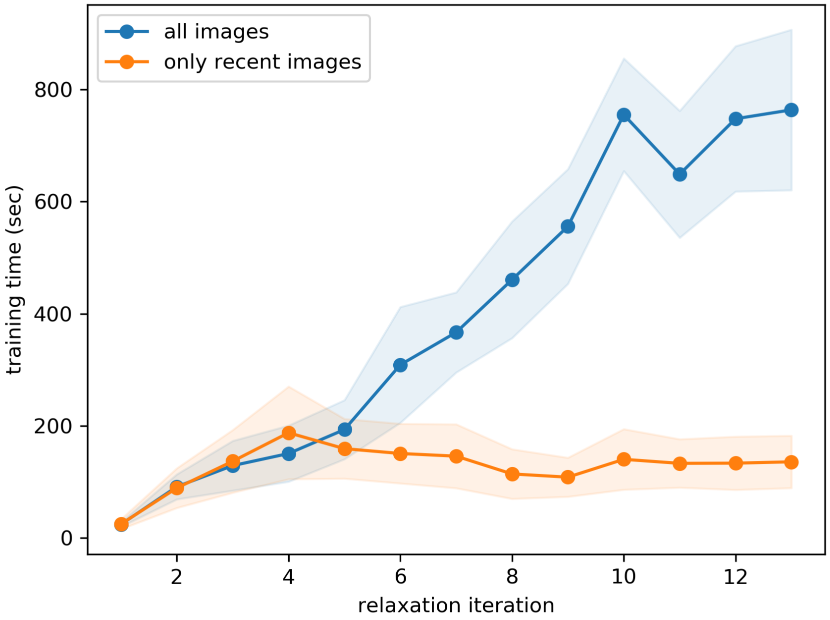

Figure 4.9: Time spent on the training process using a single NN with 2 layers and 50 neurons at each layer over iterations. The blue line shows the time for the model trained on all queried configurations while the orange line shows the time for training on the training set with fixed size. The experiment is repeated 10 times and the shaded area is the standard deviation for the 10 experiments. Time measured on 4 CPU cores.

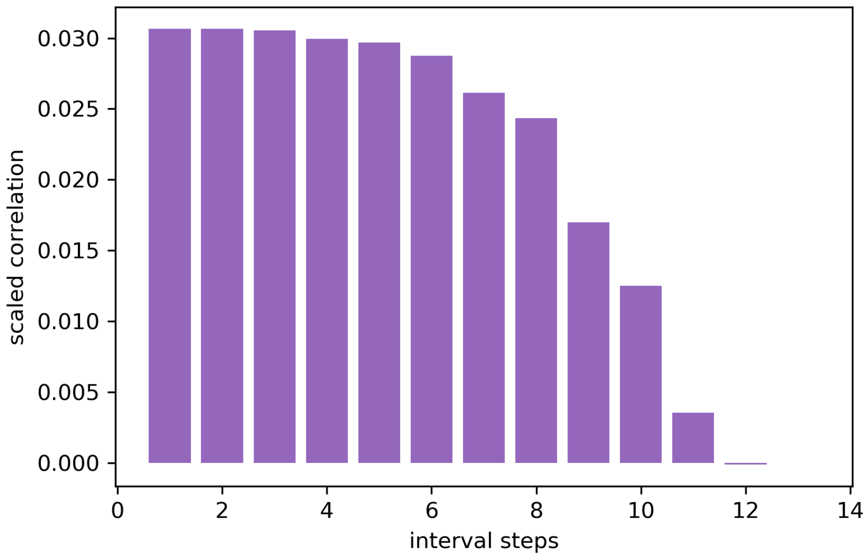

Figure 4.10: Scaled Pearson correlation coefficient between the intermediate configurations and the final relaxed configuration. The Pearson correlation is scaled by the base correlation between the initial configuration and the final configuration.

As a result of Figure 4.10, it appears in this system at least that after about five steps, the new steps are decreasingly correlated with the initial steps. Therefore, if we only focus on recent steps (e.g. the five most recent steps) and only use these configurations to update the surrogate model, the training time could be controlled as almost constant as the active learning proceeds (see Figure 4.9). We note in this case that the training time is still small compared to the time required for a single DFT calculation which is about 1.5 hours for the Acrolein/AgPd unit cell with the VASP settings in this work. When the total training set continues to grow or there are fewer computational resources available for training, the local training set could be more preferable. We note that there are cheaper probabilistic models like GPR that could be used for small dataset. But given the growing size of the available data and the wide applications of deep learning models, a cheaper way to access the uncertainty estimation for deep learning models is valuable.

4.4 Conclusion

Active learning has demonstrated promising performance to accelerate structure optimization in various applications. In this chapter, we illustrate that active learning with multiple configurations could achieve further acceleration compared to active learning with a single configuration by sharing the information across different configurations using a common NN ensemble. On the basis of that, we also provide three active learning modes for three scenarios with different amounts of prior data. By integrating the prior data into the active learning framework, more calls to expensive energy and force calculators are saved. To explore the generalization ability of this method, we compared the number of required underlying energetic calculations between the active learning, built-in VASP quasi-Newton optimizer and MDMin in ASE in various local geometry optimization tasks. The results show that active learning reduces the amount of DFT or EMT calls by 50% - 90% based on different systems. From bare slabs to surfaces with adsorbates, the acceleration becomes more significant. In addition to the surface relaxation, we also applied this method on the climbing NEB for Pt heptamer rearrangement and acetylene hydrogenation. In these examples, the acceleration is even more apparent (˜98%) while keeping almost the same transition state with the underlying ground truth energy and force calculators. In conclusion, the work in this chapter shows the potential of this NN ensemble-based active learning method in various computational surface science and catalysis tasks.

5 Simulating Surface Segregation and Aggregation of AgPd Alloy in the Presence of Acrolein

The main results of this chapter have been published in Ref. 101, which is a product of the collaboration with Mingjie Liu in our group.

5.1 Introduction

In Chapter 3 and Chapter 4, we demonstrate the use of ML potential to accelerate the Monte Carlo simulation as well as the geometry optimization. In this chapter, we aim to combine these two applications together to explore the surface segregation and aggregation phenomenon of an alloy surface in the presence of acrolein molecules. Specifically, we target to investigate the relationship between the bulk composition of the AgPd alloy and its surface configurations under the acrolein atmosphere. Here, we start with the motivation to study the segregation behavior of AgPd in the presence of acrolein molecules.

Acrolein is the simplest α, β-unsaturated aldehyde, and it has been widely used as a model species to study the selective hydrogenation of the α, β-unsaturated aldehydes over metallic catalysts.[102–105] The selective hydrogenation that converts an α, β-unsaturated aldehyde into an α, β-unsaturated alcohol is an important but difficult reaction step to produce lots of fine chemicals, for example, perfumes and pharmaceuticals.[106] It is difficult because the C=C bond is thermodynamically preferred to be hydrogenated rather than the C=O bond. Catalysts can be used to improve the selectivity towards the C=O bond hydrogenation of this reaction. For example, it was experimentally reported that metallic catalysts based on Ag or Au were able to convert acrolein to propenol with high selectivity.[103,107,108] However, the improvement of the selectivity came with a decrease in the reaction rate. Thus, several metallic catalysts based on Pd, Pt, and Ru were studied and their potential to improve the reaction rate was reported.[104,109] But these catalysts tend to hydrogenate the acrolein into propanal, allyl alcohol, and propanol with no satisfying selectivity.

To address the issue that high selectivity and reaction rate are hard to be satisfied at the same time, researchers put their focus on single-atom alloy (SAA) catalysts,[2] which shows the potential to break the constraint between two ends of the selectivity and reaction rate. The application of the SAA catalysts has been studied in many previous works. For example, Aich et al. reported a doubled reaction rate of the hydrogenation of the acrolein which was catalyzed by the 0.01% Pd /8% Ag alloy nanoparticles supported on SiO_2, compared to the reaction with pure Ag on SiO_2 as the catalyst. Also, the selectivity of this reaction was still kept at 31% compared to 37% using pure Ag.[102] In addition, a drop of the selectivity from 19% to 9% was reported by Muir et al. when they alloyed 0.2% of Pd into Ag.[110] These previous works show different reaction results using the AgPd as the catalysts for the acrolein hydrogenation reaction. Different experimental conditions such as the alloy composition, temperature, pressure, and supports have a significant impact on the experimental results. Thus, theoretical investigation on the reaction mechanism and the behavior of the catalysts during reaction will be much beneficial to the design of better catalysts.

Many efforts have been made to investigate the detailed adsorption properties of acrolein on different metallic surfaces.[111–114] However, most of the computational work focused on the acrolein configuration given a fixed surface configuration. For example, the acrolein adsorption mode on a pure metallic surface. A few works also tried to investigate the adsorption of acrolein over the single-atom AgPd alloy[102] using DFT, but their computations were only limited to a small surface unit cell with only one dopant atom on the host metals. This constraint saved a lot of computational complexity by eliminating an enormous number of unique alloy configurations. However, the interaction between the adsorbate and the metallic surface atoms is mutual. Simulations are required to thoroughly study the effect of this interaction on both the adsorption properties as well as the metallic surface configurations. These kinds of simulations were previously limited by the computational power because the surface relaxation with adsorbates itself is time-consuming. This computational complexity grows dramatically with the large space of the surface configurations. Using the active learning framework introduced in Chapter 4, we are able to obtain the relaxed adsorption structures for multiple configurations with an affordable computational cost. Combining the adsorption data and the ML potentials, the MC simulations with DFT accuracy are feasible for the AgPd alloys in the presence of the acrolein.

In the following sections of this chapter, we introduce the ML-SGCMC workflow that studies the surface configuration of AgPd FCC(111) surfaces with different bulk compositions and under the presence of the acrolein. This workflow is shown in Figure 5.1. Two main parts of the workflow are the ML models and the semi-grand canonical Monte Carlo (SGCMC) simulations. Three ML models are used as the surrogate models for the bulk potential energies, slab potential energies, and adsorption energies. The bulk potential energies and the SGCMC simulations are combined to obtain the Δ chemical potentials for different AgPd bulk compositions. Added by the slab potential energies and the adsorption energies, we are able to access the change of the total potential energy of the whole configuration (acrolein/AgPd) during the simulations. Consequently, simulating the surface segregation and aggregation including the effect of the acrolein adsorption becomes feasible using this workflow. In our work, we firstly show the DFT calculations on a small unit cell to study the effect of acrolein on the segregation behavior of the AgPd alloy. We then expand our scope by considering the AgPd slab with a larger scale and more configurations under the help of the ML-SGCMC workflow.

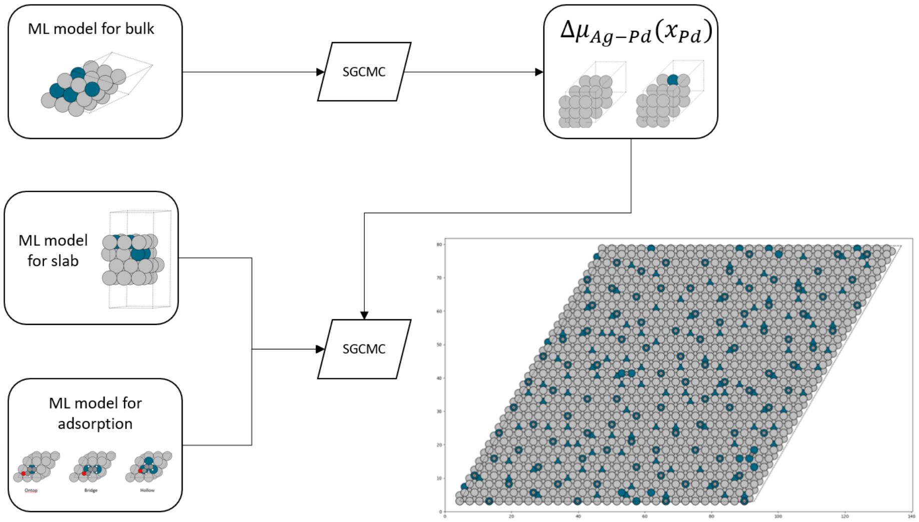

Figure 5.1: ML-SGCMC workflow used to simulate the surface surface segregation and aggregation phenomenon of AgPd in the presence of the acrolein. Three ML models serve as the surrogate models for the bulk potential, slab potential and the adsorption energies. SGCMC simulations are powered by these three ML models to study the Δ chemical potentials for different bulk compositions, as well as the surface segregation and aggregation phenomenon.

5.2 Methods

5.2.1 DFT Calculations for Potential Energies and Adsorption Energies

ML surrogate models were trained on DFT potential energies and adsorption energies. VASP was again used to calculate the energetic data for the bulk and the slab unit cells, as well as the adsorption configurations.[53,54,115,116] The exchange-correlation functional used in our calculations was Perdew-Burke Ernzerhof (PBE).[57,58] The planewave cutoff energy was set as 450 eV. The density of k-points in the Monkhorst-Pack mesh[59] was approximately 3375 k-points per reciprocal atom. The convergence of the chosen parameters was around 0.01 eV. The geometry optimization of the adsorption configurations was accelerated by the active learning framework illustrated in Chapter 4. The corresponding convergence criterion was set as 0.03 eV/Å for the forces.

5.2.2 DFT investigation on the Pd Segregation with Acrolein

We firstly used pure DFT calculations to explore the effect of acrolein on the segregation behavior of the dilute Pd atoms hosted by Ag atoms. The segregation behavior is mainly controlled by segregation energy. Equation 5.1 shows the form of the segregation energy of n Pd atoms on the Ag FCC(111) surface.

$$\Delta E_{seg}^{Pd_n} = (E_{slab}^{Pd_nAg} + nE_{bulk}^{Ag}) - (E_{slab}^{Ag} + nE_{bulk}^{AgPd}), \tag{5.1}$$

Here, $\Delta E_{seg}^{Pd_n}$ is the total segregation energy contributed by n Pd atoms, which corresponds to the energy change of n Pd atoms relocated from the bulk environment to the surface. $E_{slab}^{Pd_nAg}$ is the potential energy of a AgPd slab with

n Pd atoms located on the surface, E_{bulk}^{Ag} and E_{slab}^{Ag} is the potential energy of pure Ag bulk and slab respectively. Lastly, E_{bulk}^{AgPd} is the potential energy of a AgPd bulk unit cell with only one Pd atom. Limited by computational cost, the slab unit cell was set as $3 \times 3 \times 4$, and the bulk unit cell was set as $3 \times 3 \times 3$ in our calculations. The lattice constant of these unit cells was set as the one of pure Ag which is 4.163 Å. This is because our calculations here assume a dilute limit of Pd composition, which should have little influence on the lattice constant of the alloy unit cells.

The adsorption of acrolein affects the segregation behavior of AgPd via the change of the adsorption energies on different AgPd surface configurations or adsorption sites. The formula to calculate the adsorption energy is

$$\Delta E_{ads}^{acrolein/Pd_nAg} = E_{acrolein/Pd_nAg} - E_{Pd_nAg} - E_{acrolein}, \tag{5.2}$$

Here, $\Delta E_{ads}^{acrolein/Pd_nAg}$ is the adsorption energy of acrolein on AgPd with n Pd atoms on the surface. $E_{acrolein/Pd_nAg}$, E_{Pd_nAg}, and $E_{acrolein}$ are the potential energies of the AgPd slab with and without acrolein on the top respectively. $E_{acrolein}$ is the potential energy of acrolein in the gas phase.

The overall segregation energy considering the effect of the acrolein adsorption can be calculated by combining the raw segregation energy and the adsorption energy, which is represented in Equation 5.3.

$$\Delta E_{seg}^{acrolein/Pd_nAg} = \Delta E_{seg}^{Pd_n} + \Delta E_{ads}^{acrolein/Pd_nAg} - \Delta E_{ads}^{acrolein/Ag} \tag{5.3}$$

Here, $\Delta E_{seg}^{acrolein/Pd_nAg}$ represents the segregation energy for n Pd atoms including the adsorption of acrolein. The segregation of Pd with acrolein is favorable if this quantity is negative. This equation corresponds to the case that the initial state is a pure Ag slab with acrolein on the top plus n Ag bulk unit cells each with a single Pd atom. The final state is a AgPd slab with n Pd atoms on the surface plus n pure Ag bulk unit cells. When we relax the slabs, only the top two layers with or without acrolein are free to be relaxed.

5.2.3 ML surrogate models for Potential and Adsorption Energies

ML surrogate models were developed to provide fast and cheap estimation of the potential energies which were required during the Monte Carlo simulations. We used three ML models for AgPd bulk potential energies, slab potential energies and acrolein adsorption energies separately.

Similar to the ML model used in Chapter 4, we chose the SingleNN as the surrogate models for the bulk and slab potential energies.[89] We still used the ACSFs to encode atomic environment into numerical vectors as the input features to NNs.[18,19] The cutoff radius used to calculate the symmetry functions was set as 6.5 Å. Both of the radial and angular ACSFs were used in our models. The parameters to calculate the ACSFs are attached in Appendix B. The SingleNN used in this work has two hidden layers, each with 30 nodes. We adopted hyperbolic tangent (tanh) as the nonlinear activation function for the SingleNN.

The training data for bulk potential energies were bulk unit cells of FCC $2 \times 3 \times 3$ and FCC $3 \times 3 \times 3$. Bulk unit cells with different configurations were generated to include rich atomic environments. Specifically, the unit cell of FCC $2 \times 3 \times 3$ was fully enumerated to provide atomic environments with Pd composition ranging from 0% to 100%. These configurations set the boundaries of the region in the feature space that the SingleNN might be used. In addition to this small unit cell, we also included a bulk unit cell $3 \times 3 \times 3$ to provide atomic environments with higher resolution of Pd compositions.

Since we focused more on the dilute limit of Pd composition, thus, we only sampled configurations with Pd concentrations of 0% to 22%. During the enumeration, we eliminated symmetrically equivalent bulk configurations via unique effective medium theory (EMT)[12] energies. We used 5 different lattice constants ranging from 3.956 Å (pure Pd) to 4.163 Å (pure Ag) to generate the FCC $2 \times 3 \times 3$ bulk unit cells. For the $3 \times 3 \times 3$ unit cells, we included 3 different lattice constants from 4.060 Å (50%/50% PdAg) to 4.163 Å (pure Ag). Overall, the bulk training dataset had 123 configurations of FCC $2 \times 3 \times 3$ and 690 configurations of FCC $3 \times 3 \times 3$.

In terms of the training data for slab potential energies, we generated FCC slab configurations on a unit cell of $3 \times 3 \times 4$. We only enumerated the configurations of the top two layers of atoms while keeping the bottom two layers with pure Ag atoms because we only focused on the Pd dilute limit. Symmetrically equivalent configurations were still eliminated using unique EMT energies. These configurations were further subsampled by an energy interval of 0.01 eV. Consequently, there were 203 configurations in our slab training set.

In MC simulations, the most direct energy data we need is the total potential energy of the whole system which is the AgPd slab with acrolein adsorbed on the surface. Using the SingleNN model for the bulk and slab energies, we are able to conduct SGCMC on a bare slab. But if we want to conduct SGCMC on a AgPd slab with acrolein, we still need to calculate surface relaxation energies and adsorption energies. Here, we used another quantity, pseudo-adsorption energy, to avoid the explicit calculation of the surface relaxation energy. This quantity is defined as

$$E^*_{ads} = E_{ads} + E_{slab,relax} + E_{acrolein} = E_{acrolein+slab} - E_{slab} \tag{5.4}$$

where E^*_{ads} is the pseudo-adsorption energy, it also accounts for the effect of the surface relaxation. E_{ads} is the true adsorption energy, $E_{slab,relax}$ is the surface relaxation energy, and $E_{acrolein}$ is the potential energy of acrolein in gas phase. $E_{acrolein+slab}$ is the total energy of acrolein on the top of the slab, which is the most direct quantity used in MC simulation. Finally, E_{slab} is the energy of a clean slab without surface relaxation. Once we have a surrogate model for pseudo-adsorption energies, we can combine it with the slab energy model to obtain the total energy of acrolein on a slab. These models provide sufficient data to run SGCMC simulations.

The DFT training data for pseudo-adsorption energies was generated from a FCC(111) slab of $3 \times 3 \times 4$ with acrolein adsorbed on the surface. We considered three different adsorption sites on the top layer: ontop site, bridge site, and hollow site. Also, EMT was used to generate configurations with unique potential energy in our dataset. Consequently, there were 540 configurations in the training set. The initial state for each configuration was an acrolein placed 2.0 Å above each adsorption site, then we used the active learning method to get the relaxed geometry. We note that the adsorption energy for each adsorption site is not unique. This value could be affected by many factors, such as the initial state and the optimization method. It is computationally intractable to calculate the exact distribution of this value. Therefore, we used one value calculated from one initial state and one optimization process to approximate the adsorption energy for each adsorption site.

Numerical representations for the adsorption sites are required to build a surrogate ML model. We still chose the ACSFs[18,19] as the descriptor. However, an adsorption site usually contains multiple atoms, which makes it hard to generate a fixed-length vector to represent an adsorption site. Thus, we placed a pseudo atom above each adsorption site and then calculate the ACSF fingerprint for this pseudo atom. This method was inspired by the work related to deep learning representations for fast heterogeneous catalyst screening.[117] Since ACSF encodes the surrounding environment around a center atom into a numerical vector, in this way, the adsorption site could be represented by the ACSF fingerprint of the pseudo atom. After that, we adopted k-nearest neighbors method with k equals to 1 to predict the pseudo adsorption energies.[118] Basically, for an adsorption configuration, we calculate its ACSF fingerprint, then we searched for the most similar adsorption site in our reference dataset using the Euclidean distance of the fingerprints. The predicted pseudo-adsorption energy is the one of the nearest adsorption site. We note some adsorption configurations share the same ACSF fingerprint in the 540 training configurations. Potential causes are the accuracy of the EMT and the cutoff radius used to calculate the ACSF fingerprints. If multiple adsorption sites occur to be the nearest neighbors for a query adsorption site, we use the lowest pseudo-adsorption energy as the prediction since it is energetically more favorable.

5.2.4 Semi-grand Canonical Monte Carlo Simulation

In this section, we introduce the details of the SGCMC simulations via ML potentials. In simulations, the total number of atoms in the system was fixed while the identity of each atom was allowed to change. These simulations were conducted on a slab with 4 layers. Two bottom layers were pure Ag atoms since we focused on the Pd dilute limit. During the simulations, when the slab atoms were exchanged with the atoms in a bulk AgPd reservoir, we need the Δ chemical potential ($\Delta\mu_{Ag-Pd}$) to calculate the corresponding energy cost for that exchange. In our setting, $\Delta\mu_{Ag-Pd}$ is the energy cost to flip the identity of an bulk atom from Ag to Pd. The simulation temperate was set as 373 K.

A separate SGCMC simulations were performed to get the relationship between the bulk compositions and the $\Delta\mu_{Ag-Pd}$. A bulk unit cell of 10×10×10 was used to run these simulations. The lattice constant of this bulk unit cell was 4.163 Å which was the value of pure Ag. In simulations for each value of $\Delta\mu_{Ag-Pd}$, we initialized the bulk unit cell with a random AgPd configuration. Then at each step, a random atom was selected to flip its element type (from Ag to Pd or from Pd to Ag). The acceptance ratio of this change was determined by Equation 5.5. ΔE_{bulk} was calculated by the bulk surrogate model mentioned above. The sign before $\Delta\mu_{Ag-Pd}$ depends on the change of the atom identity. If the atom is changed from Ag to Pd, the sign is negative. Otherwise, we take a positive sign. The average of the bulk compositions at the final equilibrium steps is the corresponding bulk composition for the specified $\Delta\mu_{Ag-Pd}$.

$$Q = min(1, exp(-\frac{\Delta E_{bulk} \pm \Delta\mu_{Ag-Pd}}{k_B T})) \tag{5.5}$$

After obtaining the $\Delta\mu_{Ag-Pd}$ for each bulk composition, we were able to perform the SGCMC simulations to study the surface segregation and aggregation behaviors on a AgPd slab. Both vacuum conditions and acrolein adsorptions were considered. For the simulations on bare slab, we used a FCC slab of 12 × 12 × 4. The lattice constant was also set to be the value of pure Ag (4.163 Å). Atoms in the top two layers were initialized randomly while the bottom two layers were fixed as pure Ag atoms. At each simulation step, we tried to change the element type of a random atom in the top two layers. The corresponding energy cost was calculated by the surrogate model for slab potential energies and the $\Delta\mu_{Ag-Pd}$. The mathematical form of the accept ratio for this exchange is shown in Equation 5.6. Just like what we did for the simulations in bulk, the sign before $\Delta\mu_{Ag-Pd}$ depends on the change of the atomic identity. If the atom is changed from Ag to Pd, a negative sign is used. Otherwise, a positive sign is used.

$$Q = min(1, exp(-\frac{\Delta E_{slab} \pm \Delta\mu_{Ag-Pd}}{k_B T})) \tag{5.6}$$

Lastly, we move forward to the simulations for AgPd slab with acrolein on the surface. A larger slab of 30 × 30 × 4 was used to support the adsorption of more acrolein molecules. We still focused on the Pd dilute limit, so the lattice constant for this slab was 4.163 Å, and the bottom two layers were filled with Ag atoms. We also limited the distances among acrolein molecules to be at least 6.5 Å (around a coverage of 1/9 monolayer), which eliminated the interactions among acrolein molecules. This is because our DFT training data was not sufficient to study the interactions among the adsorbed acrolein molecules. A larger slab is required for DFT calculation if the interactions among acrolein molecules are desired, which is computationally expensive. Similar to the simulations for the bare slab, here, we still started each simulation from a random configuration for the top two layers. Then we identified each adsorption site (ontop, bridge, and hollow) and placed acrolein molecules in an order of ascending pseudo-adsorption energy (starting from most negative). Note that after an adsorption site was occupied, all other sites within the radius 6.5 Å were not valid for another occupation. At each MC step, we randomly chose an atom in the top two layers and tried to change its element type. The effect of the acrolein adsorption was considered when calculating the energy change for a MC step. If the selected atom was located at the first layer, we reassigned the acrolein for the adsorption sites within 6.5 Å away from the selected atom. The energy change was the sum of the change in slab energy and the change in the pseudo-adsorption energies. Both energy changes could be calculated by corresponding surrogate ML models. The explicit form of the acceptance probability is shown in Equation 5.7. If the selected atom was located in the second layer, we did not reassign the acrolein molecules, since an atom change in the second layer would not break or form an adsorption site in the first layer.

$$Q = min(1, exp(-\frac{\Delta E_{slab} + \Delta E^*_{ads} \pm \Delta \mu_{Ag-Pd}}{k_B T})) \tag{5.7}$$

5.3 Results

5.3.1 DFT Calculation for the Pd Segregation with Acrolein

First, we try to understand the impact of acrolein on the surface segregation behavior of AgPd using DFT calculations. We calculated Pd segregation energies and acrolein adsorption energies on 16 different AgPd FCC slab configurations (see Appendix C). We considered slabs with different numbers of Pd atoms on the first layer (from 0 to 4). The DFT calculation results are shown in Figure 5.2. The general trend is that as the number of Pd atoms increases in the first layer, the corresponding segregation energy under a vacuum condition also increases, while the adsorption energy becomes more negative. It means that the attractive interaction between the Pd atoms and the acrolein has the potential to induce the Pd to segregate to the top layer by overcoming the segregation energy. The requirement is that the total energy reduction because of the adsorption is able to compensate for the energy increase caused by Pd segregation. According to the DFT calculations, only one configuration (configuration 5) with a single Pd atom in the top layer satisfies this requirement, which means that a single Pd atom on a Ag slab is energetically favorable under the acrolein atmosphere. There are likely other configurations that are energetically favorable, but it is too computationally expensive to do an exhaustive search using DFT.

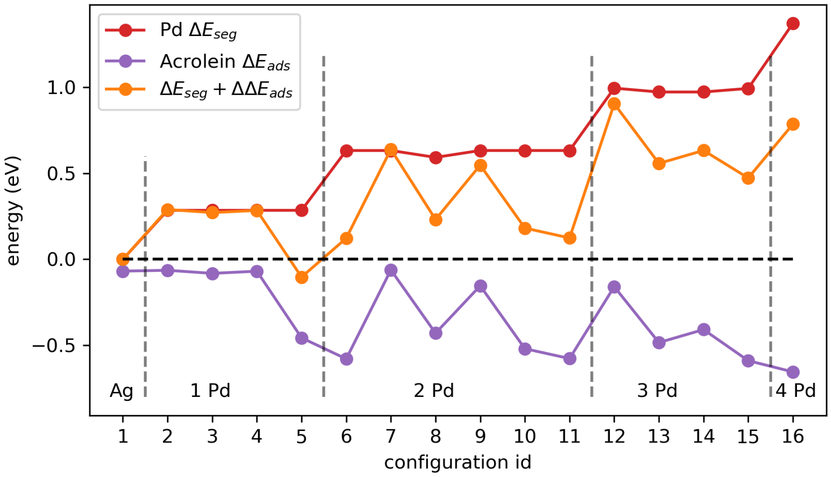

Figure 5.2: Pd segregation energies under vacuum (Pd ΔE_{seg}), acrolein adsorption energies (Acrolein ΔE_{ads}), and Pd segregation energies with acrolein ($\Delta E_{seg} + \Delta\Delta E_{ads}$) of different AgPd configurations. $\Delta\Delta E_{ads}$ means the difference of the acrolein adsorption energy on an adsorption site and on pure Ag.

5.3.2 Performance of the ML Surrogate Models

Before we get into the simulation results, we check the performance of the ML surrogate models, which is significant to the reliability of the simulation results.

SingleNN was used as the surrogate model for the bulk and slab potential energies. Each dataset was randomly divided into training, validation, and test sets with a ratio of 8:1:1. For the bulk surrogate model, which was used to obtain the relationship between the bulk composition and the $\Delta\mu_{Ag-Pd}$, its model performance is shown in Figure 5.3. The root mean square error (RMSE) is about 1.6 meV/atom on the test set. The residual error is distributed evenly across the data space. The prediction difference of the SingleNN models trained from training sets split from different random seeds is shown in the right plot of Figure 5.3. The value of the prediction difference is similar to the test error, which indicates that we can run multiple simulations using the models trained from different datasets to estimate the effect of model error on the simulation results.

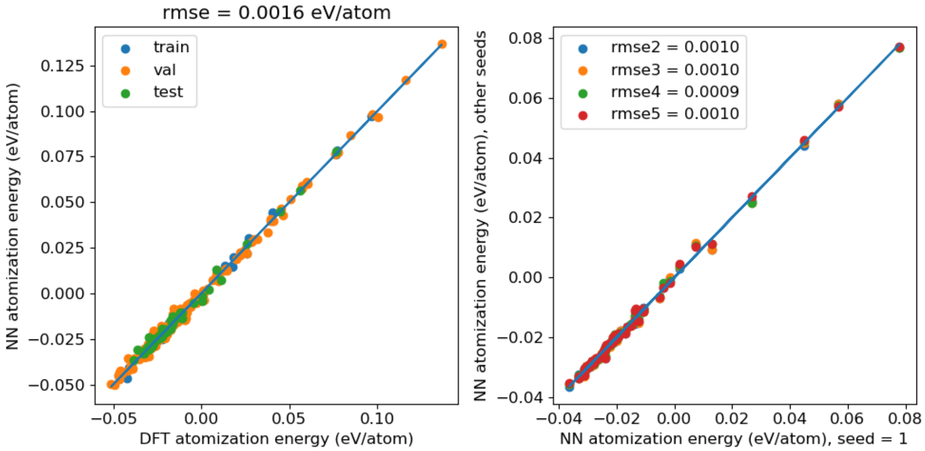

Figure 5.3: Parity plots for the surrogate model (SingleNN) of AgPd bulk potential energies. Left figure shows the performance on the training, validation, and test set. Right figure shows the prediction difference of the SingleNN models trained from different training setss (split by differnet random seeds).

Similarly, the performance of the SingleNN model for the slab potential energies is shown in Figure 5.4. The RMSE on the test set is about 0.5 meV/atom. The prediction difference caused by random dataset split is also similar to the test error.

Different from the SingleNN model used for the bulk and slab potential energies, we used k-nearest neighbor method with k equals 1 as the surrogate model for pseudo-adsorption energies. For each data point, we find its closest point in our reference dataset, then we use that label as the prediction. The accuracy of this model was analyzed by leave-one-out cross validation. For

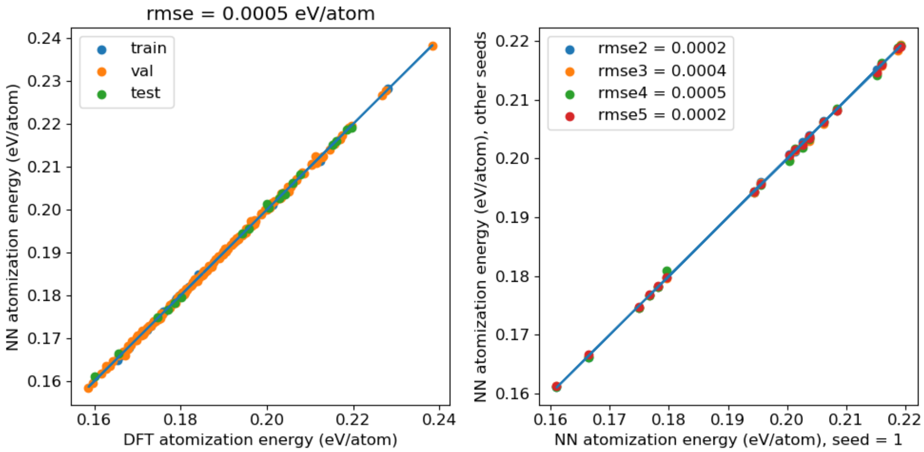

Figure 5.4: Parity plots for the surrogate model (SingleNN) of the AgPd slab potential energies. Left figure shows the performance on the training, validation, and test set. Right figure shows the prediction difference of the SingleNN models trained from different training set (split by differnet random seeds).

each data point in the dataset (287 configurations in total), we search for its nearest neighbor in the remaining 286 configurations. The model performance is shown in Figure 5.5. Figure 5.5 (a) shows the parity plot between the DFT energy and the model prediction energy. The overall RMSE is about 0.1 eV. However, different adsorption sites show different levels of RMSE. Ontop site has the smallest error of 0.038 eV. Bridge site and hollow site have relatively larger errors which are 0.121 eV and 0.093 eV respectively. Figure 5.5 (b) is a histogram plot of the error distribution. Most of the errors are less than 0.05 eV. We also found that the magnitude of the error depended on the configurations of the test and the label data points. If they have a different number of Pd atoms in the sublayer, the error is relatively large (as shown in Figure 5.5 (c)). Otherwise, the error is relatively small (Figure 5.5 (d)). We note that the actual RMSE of the pseudo-adsorption energy during the simulations will be smaller than 0.01 eV. Because in our dataset, the unit cell is limited to 3 × 3 × 4, and the atoms in the unit cell are repeated in two horizontal directions. One more Pd atom in the sublayer of a 3 × 3 × 4 unit cell actually has a double effect on the adsorption site in the top layer. However, during the simulations, new configurations will only have the effect of one more or less Pd atom in the sublayer because of the much larger size of the unit cell. Therefore, the actual RMSE in the simulations will be about 0.05 eV.

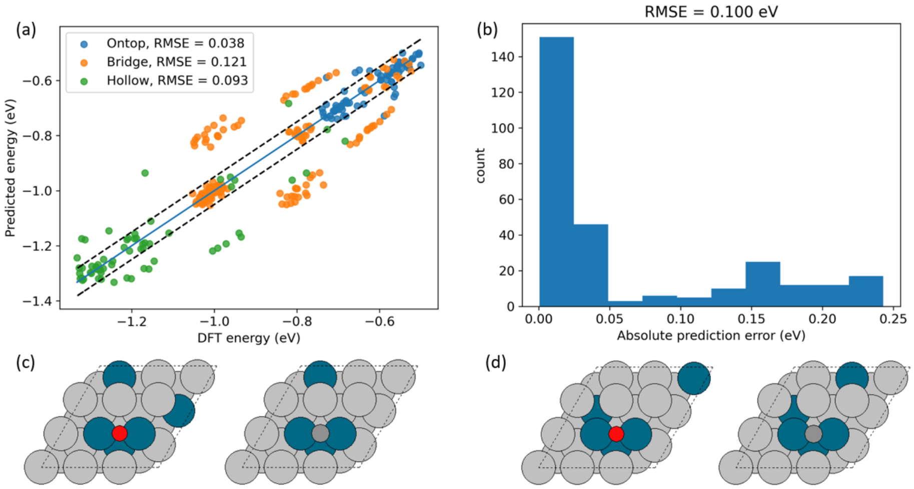

Figure 5.5: (a) Parity plot for the pseudo-adsorption energy surrogate model. Dashed lines indicate an error of 0.05 eV. (b) Histogram of the prediction error. (c) Typical configuration with large prediction error. Configuration with red pseudo atom is the test configuration while the grey pseudo atom indicates the label configuration. They have different number of Pd atoms in the sublayer. (d) Typical configuration with small prediction error. Red and grey pseudo atom indicate the test and label configuration respectively. They have the same number of Pd atoms in the sublayer.

5.3.3 Semi-grand Canonical Monte Carlo Simulation Results

We used SGCMC simulations to study the AgPd bulk chemical potential, the surface segregation and aggregation of the AgPd slabs under vacuum and acrolein conditions. In the following parts, we show and discuss the simulation results on these three tasks.

First, we studied the relationship between the AgPd bulk composition and $\Delta\mu_{Ag-Pd}$. We performed simulations on a bulk unit cell of $10 \times 10 \times 10$ with $\Delta\mu_{Ag-Pd}$ ranging from -2.9 eV to -2.35 eV. The simulation temperature was 373 K. For each round of simulations, we conducted 100000 MC steps such that each atom was expected to be involved in 100 MC steps and the bulk composition was expected to reach an equilibrium state. Then we took the average of the compositions at the last 50% steps as the bulk composition value for the given $\Delta\mu_{Ag-Pd}$ value. Bulk SingleNN models trained from 5 different training sets were used to evaluate the simulation variance. The relationship between $\Delta\mu_{Ag-Pd}$ and the bulk composition is shown in Figure 5.6. $\Delta\mu_{Ag-Pd}$ and the bulk composition has a linear relationship when the Pd bulk composition is below 10%. The slop suddenly becomes steeper after 10% of the Pd composition. The standard deviation of the simulation results using different models is small when the Pd concentration is below 20%, which means the effect of the bulk surrogate model error on the simulation results is also small in this range. We also investigated the effect of the lattice constant on the simulation results. Figure 5.6 also shows that the lattice constant essentially has no apparent effect on the simulations with Pd concentrations below 10%.

With a function mapping from Pd bulk composition to $\Delta\mu_{Ag-Pd}$, we simulated the segregation of Pd under the vacuum condition using the SingleNN model for slab potential energies. Pd bulk concentrations between 5% to 30% were simulated, which corresponded to -2.70 eV to -2.35 eV for $\Delta\mu_{Ag-Pd}$. Similar to the bulk simulations, each atom was expected to be changed by 100 times, which resulted in 28,800 total MC steps for each simulation. The last 50% of configurations were used to calculate the equilibrium states for the Pd concentrations in the first and second layers.

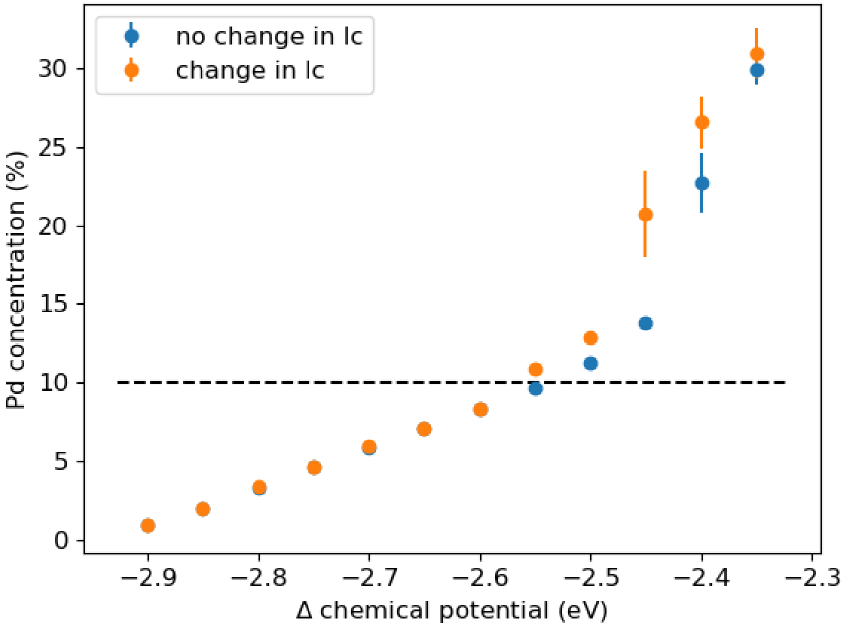

Figure 5.6: Relationship between the AgPd bulk composition and $\Delta\mu_{Ag-Pd}$. Blue points show the results of the simulations without changing the lattice constant with the bulk compositions. Orange points show the simulations with changing the lattice constant according to the Vegard's law.[66] The error bars show the standard deviation of the simulations using 5 bulk SingleNN models trained from the datasets with different split.

Corresponding results are shown in Figure 5.7. The first layer is completely dominated by Ag atoms across the considered bulk compositions. The Pd atoms only appear in the first layer when the Pd bulk composition increases over 20%. It means that Pd atoms are not favored to segregate to the top layer under the vacuum condition. This observation is consistent with previous reports about the segregation behavior of AgPd.[119–121] Another observation is the enrichment of the Pd atoms in the second layer. The concentration of Pd in the second layer is always greater than the value of the bulk concentration. Similar phenomenon can also be found in AuPd alloys.[49]

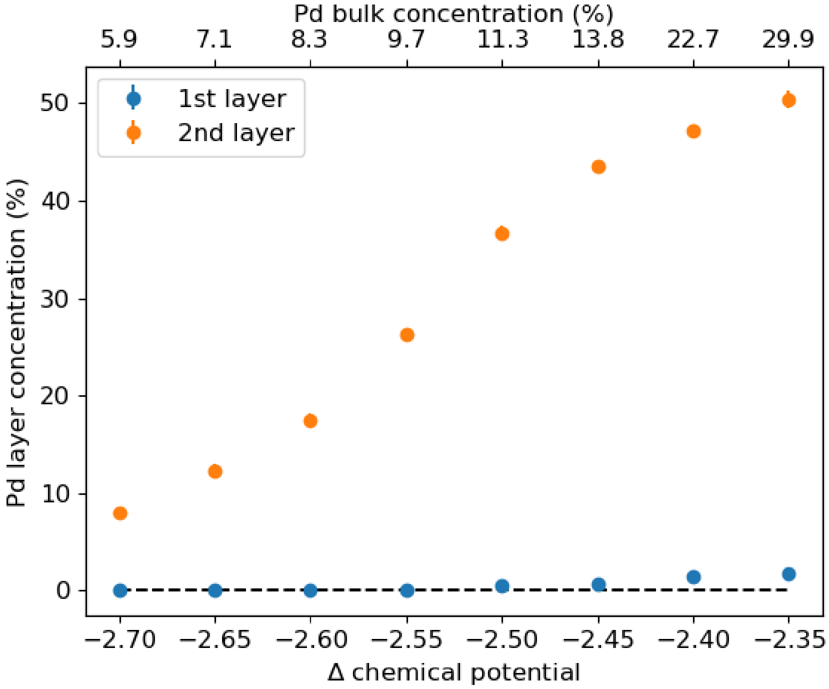

Figure 5.7: SGCMC simulation results for the AgPd slab under vacuum condition. Pd concentrations in the first and second layer are colored as blue and orange. Error bars represent the standard deviations of simulations using 5 different SingleNN model trained from different dataset split.

Furthermore, the segregation and aggregation behaviors of the AgPd slab with acrolein on the surface were studied using the SGCMC simulations at

373 K. We simulated 9 bulk concentrations ranging from 5% to 10%. Their $\Delta\mu_{Ag-Pd}$ values were from -2.74 eV to -2.58 eV with an incremental step of 0.02 eV. There were 500,000 total MC steps in each simulation and each atom was expected to change its element type by 278 times. More MC steps were used here because we wanted to investigate the details of the adsorption site (ontop, bridge and hollow) distribution, which requires sufficient MC steps on the atoms in the first layer. The last 50% of the accepted configurations were used to calculate the equilibrium quantities (Pd compositions, ratio of each adsorption site, etc.). Different SingleNN models were also used to estimate the effect of the model error on the simulation results. Since the surrogate model had an error of 0.05 eV during the simulations, we trained additional four models on the training set whose labels were added by Gaussian noise with a standard deviation of 0.05 eV. Additional SGCMC simulations were performed using these models and the standard deviation of the simulation results were calculated.

The segregation and site distribution results are shown in Figure 5.8 and Figure 5.9 respectively. When the Pd bulk composition is between 5% to 6.5%, the Pd surface composition is about 11%. It means with the help of acrolein, Pd tends to segregate to the surface, which is in contrast to the simulation result under the vacuum condition (Ag segregates to the surface). Also, the surface concentration is kept at 11% which means the adsorption sites are dominated by the ontop site. It is also presented in the site distribution plot. This plateau is because there is a gap between the energy bars for the configurations with one and two Pd atoms on the surface. Accumulation of the Pd bulk composition is needed to break this energetic gap. The bridge site starts to increase when the Pd bulk composition is greater than 6.5%. The hollow site appears after the Pd bulk composition reaches 8.3%. For the Pd concentration in the second layer, it has a steady increase along with the Pd bulk composition, which is similar to the results under the vacuum condition. We can also see that the second-layer Pd composition is more sensitive to the model error, it has a larger standard deviation in these simulations. However, it has a small influence on the Pd concentration in the first layer. In addition, the main trend of the site distribution is not affected by the small model error. Thus, the main conclusions for the Pd segregation and aggregation still hold even if the adsorption energy surrogate model has an error of 0.05 eV.

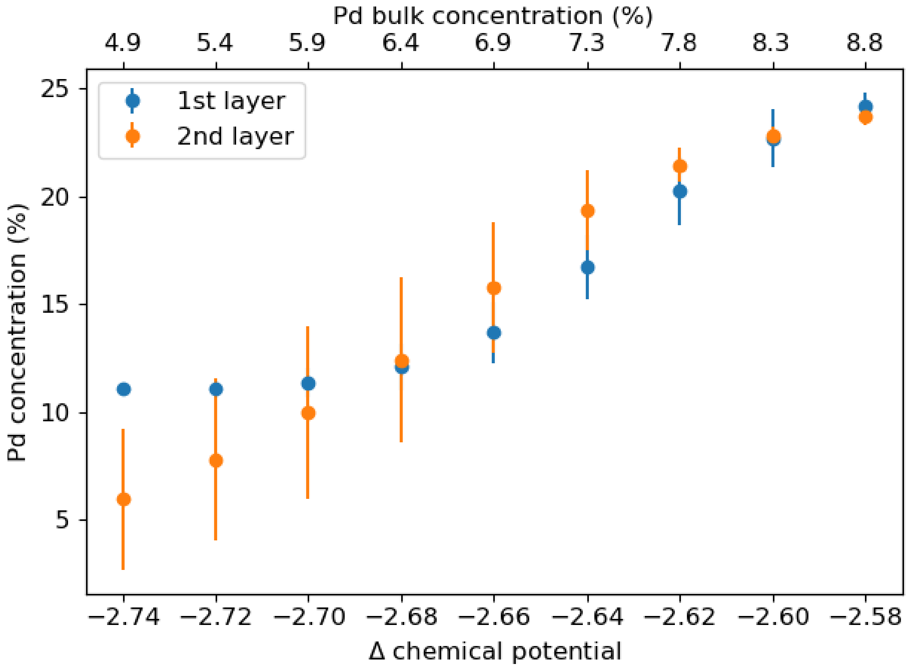

Figure 5.8: AgPd segregation results in the presence of acrolein on the surface. Error bars represent the standard deviation of the simulations performed with different SingleNN models.

5.4 Conclusion

In this chapter, we illustrated the ML-SGCMC framework that combined ML potentials with SGCMC to make simulations on large unit cells achievable. The segregation and aggregation behaviors of the AgPd alloy were investigated

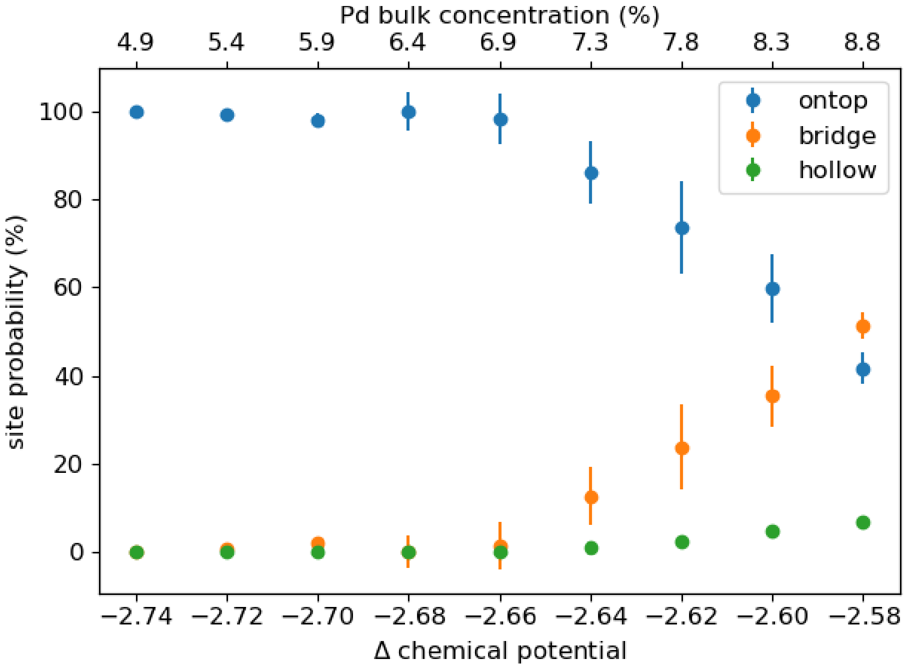

Figure 5.9: Pd adsorption site distribution in the presence of acrolein. Error bars represent the standard deviation of the simulations performed with different SingleNN models.

at the Pd-dilute bulk compositions. In both pure DFT calculations and ML-SGCMC simulations, Pd atoms did not segregate to the surface under the vacuum condition. However, the attractive interaction between acrolein and Pd was able to induce Pd atoms to the surface. When the Pd bulk composition was below 6.9%, the Pd atoms were present on the surface in a single-atom form. After that, the ratio of the bridge site and the hollow site increased with the Pd bulk composition. The effect of the ML surrogate model error on the simulation results was also investigated in this chapter. As shown in the results section, for Pd-dilute bulk compositions (under 10%), the model error had little impact on the segregation and aggregation behaviors of the AgPd alloy in our simulations.

Several assumptions were made to make the calculations and simulations tractable. For example, there was no interaction among the acrolein molecules on the surface. To include this effect, more DFT calculations for the slabs containing more than one acrolein molecule are required, which is computationally unaffordable. In addition, the actual reaction conditions are much more complicated. There are more adsorbates on the surface, such as hydrogen, allyl alcohol, and other intermediates. Simulations including these complex factors are still challenging. Besides solving these assumptions, improving the accuracy of the adsorption energy model would also be beneficial to the reliability of the simulations. Overall, in this chapter, we present a basic framework to accelerate the SGCMC simulations with adsorbates using ML potentials, although there are still many aspects to be improved.

6 Searching Similar Molecular Structures with Neural Network Embeddings

6.1 Introduction

The applications of ML are not limited to regression tasks, many other fields also benefit from the development of the ML technique. In this chapter, we demonstrate a search framework for related molecular structures in large databases powered by ML representations. The main results in this chapter are from a collaboration with Mingjie Liu in our group. Here, we start by introducing the motivations of our work.

Data is the central part of almost all machine learning applications. With the increasing capacity to generate and store more data, efficient methods to retrieve target data of interest become much more in-demand. In the chemistry field, the sizes of the datasets grew dramatically in past years. For example, Materials Project has more than 140 thousand inorganic compounds, 530 thousand nanoporous materials, and their properties.[11] PubChem includes more than 100 million compounds.[122] Open Catalyst 2020 (OC20) provides the DFT calculations of more than 130 million adsorption structures.[9] Given the huge sizes of existing datasets and potentially larger datasets in the future, we need fast methods to explore and search in these datasets.

Usually, researchers may want to search for molecules or materials with similar properties in applications like discovering new drugs or cheaper materials.[123–125] Many similarity search methods have been developed for this purpose.[126,127] In general, a similarity search approach consists of three essential components: molecular representation method, quantitative metric to measure the similarity, and a search algorithm. The search process usually starts with one or more query molecules (e.g., configurations known for their reaction activities). Then the representation method converts them into numerical representations which could be used to calculate the pair-wise similarities. After that, the search algorithm retrieves molecular candidates on the basis of the similarity measurement. The retrieved molecules are ranked by their similarities to the query molecule(s) in descending order. Significant efforts have been spent on designing fingerprints to represent molecules. For example, SMIfp (SMILES fingerprint) converts a molecule into a 34-dimension scalar fingerprint.[128] Each element of the fingerprint counts the occurrences of 34 symbols in SMILES, where SMILES (Simplified Molecular Input Line Entry System) is a chemical language and information system used to represent different atoms and bonds with ASCII characters.[129,130] The substructure-based fingerprint is also a popular choice to represent molecules. Each item of the fingerprint encodes whether or not a substructure is present in a molecule. Typical examples include the Molecular ACCess System (MACCS) and the Barnard Chemical Information Ltd. (BCI) fingerprint.[131,132] MACCS uses 166 structural fragments as the keys while BCI contains 1052 substructures. These fingerprints rely on a pre-built library of substructures as the keys, which limits their applications only for relaxed molecules. For structures like complex adsorbates on surfaces or unrelaxed geometries, representation methods with higher resolution by focusing on geometrical details are preferred.

In the past decade, the development of deep learning methods has changed the way to represent data like text and images. Deep learning models like convolutional neural network (CNN) and recurrent neural network (RNN) have been widely applied in computer vision and language processing tasks.[14,133–135] For most of the deep learning models, the last layer of the deep neural network represents the input data as a numerical vector which contains rich information about the data. This vector representation is also called embedding. Since the output usually linearly depends on the embedding, we can also regard the embedding as a nonlinear dimensional transform of the input into a space where the output is linear. The promising performance of the deep learning models in various tasks implies the embedding must represent the data in a reasonable way. Therefore, these neural network embeddings have been applied in many information retrieval systems involving image and text.[136–138] For molecular data, several graph neural network (GNN) models have been proposed to learn the embeddings to represent the atomic configurations, such as the CGCNN and the GemNet.[21,23] The atomic embeddings contain information like the element type of the central atom, position, and elemental information of the neighboring atoms. When applied in specific tasks (e.g., energy and force prediction), it is reasonable to think that neural networks could be trained to generate atomic embeddings in a space where the specific property (e.g., atomic energy) is linearly related to the embedding vectors. Therefore, the distance between the embeddings in this space could serve as a similarity measure of a specific property.

In this work, we demonstrate a method based on neural network embeddings to search for similar molecular structures. This method can be applied to any atomistic system including organic molecules, bulk materials, and adsorption systems. When combined with approximate nearest neighbor search methods,[139] neural network embeddings can be used to retrieve similar atomic structures efficiently in large databases. We also show that the similarity is related to the specific property which is used to train the neural network models. Therefore, this method has the potential to search for similar molecular structures in a property-oriented way.

6.2 Methods

6.2.1 Searching Similar Molecules via Neural Network Embeddings

In this section, we introduce the overall framework to search for similar molecules in a database using neural network embeddings. This framework is shown in Figure 6.1. The whole workflow can be divided into two stages: preparation and query. In the preparation stage, we use a database of molecules to train a neural network model. Then we use the trained model to calculate the embeddings of the atoms in the database. These atom embeddings are processed into a specific data structure (e.g., space-partitioning tree) for future searching by an approximate nearest neighbor (ANN) search algorithm. In the query stage, given a query molecule, we use the same trained neural network model to get the embeddings for the atoms in the molecule. Then we retrieve neighboring atom embeddings using an ANN search method and return the corresponding molecules as the results for the query event. If the query wants to find similar atomic environments, then the atoms corresponding to the embeddings are directly returned. More details of each step will be discussed in the following sections.

6.2.2 GemNet to Generate Atom Embeddings

In this work, we used GemNet,[23] a type of graph neural network, to generate atomic embeddings as the real-value fingerprints. GemNet uses embedding layers and message passing to encode the information like element type, radial distances of atom pairs, angles of atom triplets, and

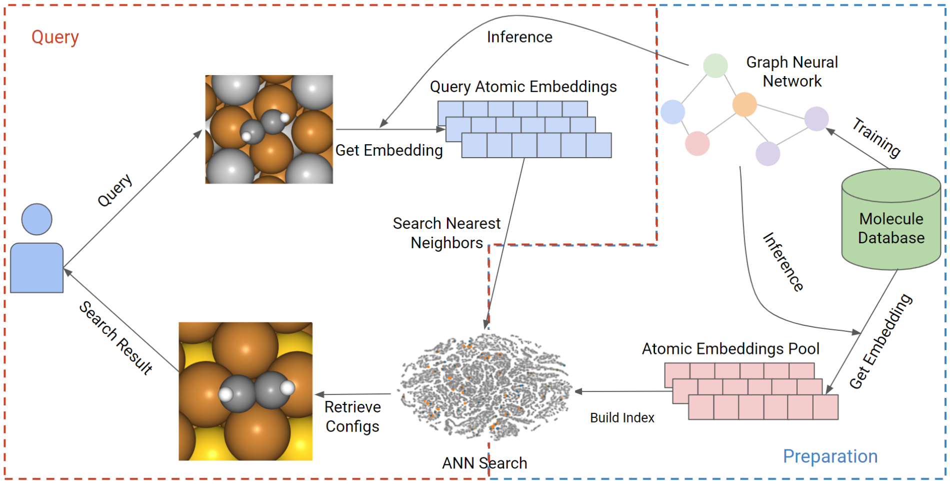

Figure 6.1: Framework for similar molecules search using neural network embeddings and ANN search methods. The red frame encloses the query stage and the blue frame denotes the preparation stage.

dihedral angles of atom quadruplets into a real-value vector (atom embedding) to describe an atom and its surrounding environment.[23] The message passing in GemNet can be described by Equation 6.1.[23]

$$\begin{multline}\tilde{\mathbf{m}}_{ca} = \sum_{b\in\mathcal{N}_a^{int}\setminus\{c\},\, d\in\mathcal{N}_b^{emb}\setminus\{a,c\}} \Big(\left(W_{SBF1} e_{SBF}\left(x_{ca},\phi_{cab},\theta_{cabd}\right)\right)^T \mathbf{W} \\ \left(\left(W_{CBF2}W_{CBF1}e_{CBF}\left(x_{ba},\phi_{abd}\right)\right) \odot \left(W_{RBF2}W_{RBF1}e_{RBF}\left(x_{db}\right)\right) \odot \mathbf{m}_{db}\right) \Big) \tag{6.1}\end{multline}$$

where a, b, c, d are four atoms in a quadruplet whose relative geometric information is encoded. $\mathbf{m}_{db}$ denotes the directional embedding between atom d and b. $\tilde{\mathbf{m}}_{ca}$ means the updated directional embedding between atom c and a. W represents a weight matrix, $\mathbf{W}$ denotes a weight tensor. $\odot$ is an element-wise multiplication operator. e means a basis function whose type is annotated by different subscripts. RBF means radial basis function for the distance information of atom pairs. CBF means cosine basis function for the angular information of atom triplets. SBF denotes spherical basis function for the dihedral angular information of atom quadruplets. For the mathematical definitions of these basis functions, please refer to the GemNet paper.[23]

The atom embedding $\mathbf{h}_i$ is initialized randomly based on the atom type. Then, nonlinear layers are applied on both atom embeddings ($\mathbf{h}_i$) and directional embeddings ($\mathbf{m}_{ij}$) to update them in an interdependent way. Finally, the atom embeddings undergo a linear transformation to output target atomic properties like atomic energy. In our work, the GemNet model (directional embeddings, atom embeddings, weight matrices) is trained with potential energy data. In other words, the atom embeddings are used to predict atomic contribution to the total potential energy. Therefore, atom embeddings that are closed in this embedding space are expected to be energetically similar or relevant. The dimension of the atom embeddings is 128 in our work. Other hyperparameters can be found in Table E.1. We note that the implementation of the GemNet model was adopted from the GitHub repository of the OC20 Project.[9]

6.2.3 Approximate Nearest Neighbor (ANN) Search

Atomic embeddings are vectors of real values. Typical distance metrics for real vectors include Euclidean distance, inner product, and cosine similarity. A straightforward way to find k nearest neighbors for a query vector in a vector database is to calculate the distance between every candidate vector and the query vector. In the meanwhile, maintain a priority queue with a size of k. However, it has a time complexity of $O(nd + n \log k)$, where n is the number of candidates in the dataset and d is the dimension of the vectors. This computational complexity makes searching slow for large databases and high-dimensional embeddings. Exhaust searching for the exact k closest results is usually time- and resource-consuming for large databases.

Therefore, many ANN search methods and libraries have been developed to find the approximate results with much less time and resources.[139] The ANN search methods can be classified as hashing-based, quantization-based, tree-based, or graph-based methods according to the techniques used to accelerate the search process.[140] Typical examples include locality-sensitive hashing, SPTAG, and ScaNN.[141–143] In our work, we used the Faiss library to implement the ANN search part.[144] Faiss is a library containing implementations for several ANN search algorithms. Its IndexIVFPQ mode is used in our work. The bases of IndexIVFPQ are inverted file system and product quantization.[145,146] The inverted file system is built by applying k-means clustering on a database of vectors to form a set of centroids. These centroids allow rapid access to a small fraction of near vectors for a query vector, which avoids exhaust comparison against each vector in a database. Then, the search efficiency is further enhanced by using product quantization on the residual query vector (subtract the corresponding centroid from the query vector). The essential idea of product quantization is dividing a vector into small subvectors, applying k-means clustering on these subvectors, and using the corresponding centroids of subvectors to represent the original vector. Recording the centroid index uses less memory compared to saving the whole real vector. The parameters of the Faiss IndexIVFPQ method used in our work are attached in Table E.2. Faiss supports similarity metrics like L_2 distance and inner product. L_2 distance was used in our work.

In this work, the similarity search is operated on two aspects, at the atomic level and the molecular level. At the atomic level, we can directly use ANN search on the atom embeddings. However, for the nearest molecules search, we need to convert the similarity of atom embeddings into the similarity of the molecules. For each atom in a molecule, we search for q

approximate nearest atom embeddings in a database. These atom embeddings are considered matched for the query atom. The corresponding molecules containing these matched atoms are added to a candidate set. After looping over all atoms in the query molecule, we rank the molecules in the candidate set according to the number of matched atoms in descending order (sum of Euclidean distances between the matched atom embeddings is used to break ties). Top k molecules are the k approximate nearest neighbors. Large q prefers the candidates that are globally similar to the query molecule while small q favors the molecules containing local environments with high similarity to the query molecule. We used q around 10 times k in our work.

6.2.4 Datasets

We demonstrate the ANN search on three datasets across organic molecules, bulk materials and surfaces. For the organic molecules, we applied our search method on the QM9 dataset.[147,148] QM9 contains properties of 134k small organic molecules with elements of C, H, O, N, and F. In terms of the bulk materials, we adopted the Materials Project dataset which includes more than 126k bulk crystals.[11] QM9 and Materials Project databases were obtained from the SchNetPack package.[27] For the surface systems, we used the IS2RS subset from the newly released OC20 dataset,[9] which contains about 460k relaxed adsorption configurations. For each of the above dataset, we train a GemNet model on their potential energy data to learn the atom embeddings. After that, the atom embeddings were used in the search tasks.

6.3 Results

In this section, we demonstrate the GemNet embedding-based ANN search results for different molecular systems: small organic molecules, metallic bulk materials, and metallic surfaces with adsorbates.

6.3.1 ANN Search for Organic Molecules

The first case of ANN search is for small organic molecules, which was performed in the QM9 dataset. The whole dataset was split into the training and validation sets randomly with a ratio of 8:2. A GemNet model was built on the training set. The energy mean absolute error (MAE) on the training and validation sets was 4.57 eV and 4.97 eV separately. We then used this model to obtain the embeddings for atoms in the QM9 dataset and search for similar molecules in this database. We chose several molecules and functional groups as the queries to search for similar (sub)structures. The examples include molecules of benzene and toluene, as well as groups of hydroxyl, amino, and imino. Here, we only discuss the results for benzene and a joint search of amino and hydroxyl groups. Results for other examples can be found in Appendix F. For benzene, we used the GemNet embedding of each atom as the query vector to search for similar atomic environments. Then sort the candidate molecules based on their number of matched atoms and the sum of the L_2 distances as mentioned in section 6.2.3. The searched molecules are shown in Figure 6.2. The top left molecule is the query benzene while the Figure 6.2 b to Figure 6.2 f are the nearest 5 molecules. They all contain a 6-atom ring structure with some small difference against the query benzene. Basically, the 6 atoms in the ring are all carbon. Except in Figure 6.2 c and Figure 6.2 f, one carbon atom is replaced by a nitrogen atom. There are also some extra groups on the rings like hydroxyl and amino groups. But generally, these searched molecules are similar to benzene in terms of elemental and geometrical features.

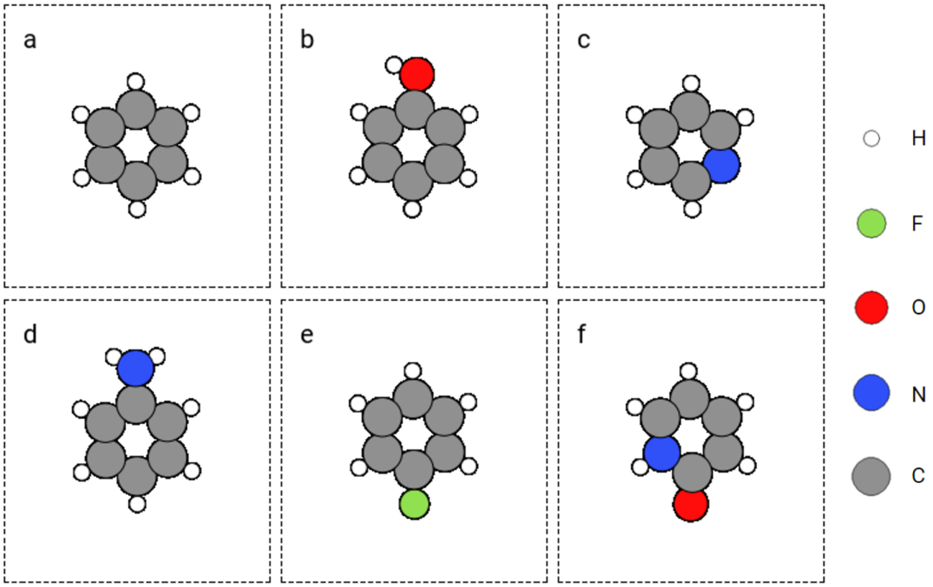

Figure 6.2: Retrieved similar molecules (top 5) for benzene. Figure a is benzene used as the query molecule. Figures b to f show the nearest molecules in the QM9 dataset.

We also compare the results to the search based on the Morgan fingerprint with the Tanimoto coefficient. Morgan fingerprints are a way to represent a molecule using a 2048-bit vector by building atom identifiers based on atomic environments like atom types, connected bonds, and neighboring atoms.[149,150] The Tanimoto coefficient is a common metric used to measure the similarity between two bit vectors by taking the ratio of intersection over union.[151] The search results of the Morgan fingerprints and the Tanimoto coefficient are shown in Figure 6.3. The main difference from the GemNet result is in Figure 6.3 b and Figure 6.3 c, where larger rings are retrieved in the search results. These two molecules are less similar to benzene from the aspect of atom numbers and bond angles in the ring structure. According to the top 5 nearest molecules, GemNet embedding retrieves more similar molecules than Morgan fingerprint.

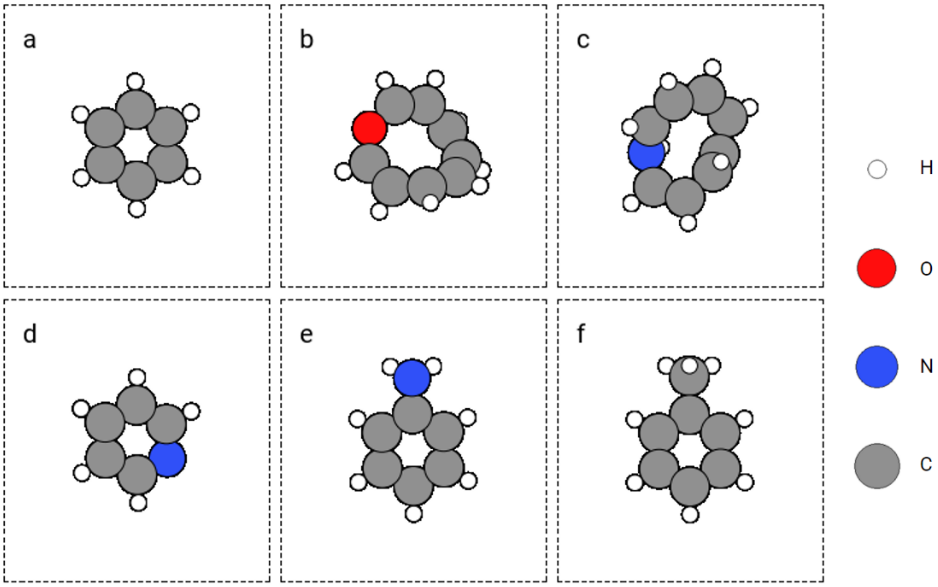

Figure 6.3: Searched similar molecules (top 5) using Morgan fingerprint with Tanimoto coefficient as a distance measure.

In addition to the qualitative evaluation of the similarity by visual comparison over the elemental and geometrical features, we also analyze the similarity among the molecules by investigating their relevance in the energetic embedding space. We built Gaussian process regression (GPR) models using the searched molecules as the training set. We used the FLARE package as the implementation of the GPR models.[85] The hyperparameters for the GPR model are attached in Table E.3. During the training process, we iteratively added the search molecules one by one into the training set and updated the GP model, then we used the GPR model to predict the energy of benzene and compared the prediction with the true label. The results of the GPR models are shown in Figure 6.4. We not only included the results from the training set searched using GemNet embeddings and Morgan fingerprints, but we also used a set of random molecules from the QM9 dataset as the baseline. In Figure 6.4, we can see that as we add more configurations into the training set, the prediction error and standard deviation are generally decrease. However, using molecules searched in different ways, the GPR models have different performances. GPR model trained on the molecules retrieved by GemNet embeddings has the smallest prediction error (0.04 eV) and standard deviation (0.02 eV). The GPR model from Morgan fingerprints has a larger error (3.64 eV) and standard deviation (17.14 eV). The GPR model from the random molecules has the largest error and prediction uncertainty, which is 5.76 eV and 43.44 eV respectively. These results imply that GemNet embedding has a representation of the atomic environments that is more relevant to the energetic property of the molecules. This is not surprising since the GemNet model was trained on the energy data of the molecules and the atomic energy prediction is a linear regression on the atom embeddings.

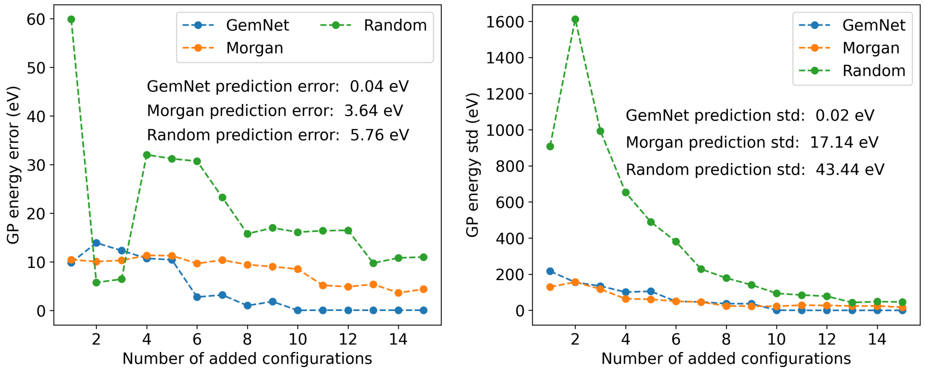

Figure 6.4: Prediction performance of GPR models trained on molecules searched using GemNet embeddings and Morgan fingerprints, as well as a set of molecules randomly sampled from the QM9 dataset. Left figure shows the prediction error while the right figure shows the standard deviation (std) of the GPR prediction. Number of added configuration means the numebr of molecules added into the training set to build the GP regression model. The annotations in the figures are the mininum prediction error and standard deviation for the models trained on different number of configurations.

Besides the search for a whole molecule, we can also use the GemNet embeddings to search for substructures. Here, we demonstrate an example of searching for a molecule containing similar substructures to the hydroxyl group of the butanol and the amino group of the glycine. The search procedure is similar to the method for benzene but has an additional step to join the search results from hydroxyl and amino groups, which is similar to an ”and” operator on two sets. Figure 6.5 shows the query substructures and the searched molecule. The atoms in the query and matched substructures are marked as crossed. Both hydroxyl and amino groups are retrieved in the resulting molecule. In addition, the retrieved hydroxyl and amino groups are somehow similar to the queries. For the hydroxyl groups, they are both at the end of a three-carbon chain for the query and searched molecules. For the amino groups, they are at the end of a two-carbon chain and there is a -OH group at the other end.

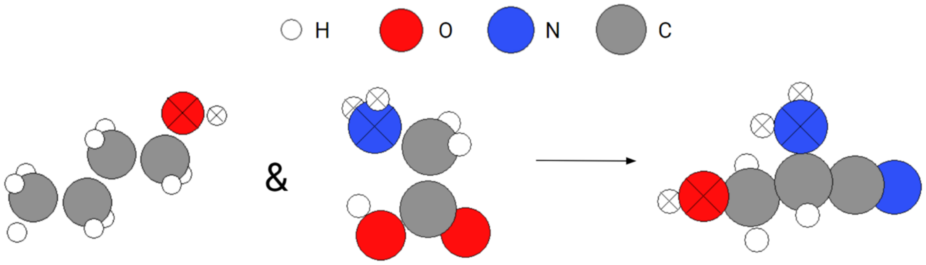

Figure 6.5: Joint search result for hydroxyl and amino groups. The query substructures are marked as crossed at the left of the arrow. The retrieved molecule is at the right side with matched atoms also marked as crossed.

6.3.2 ANN Search for Bulk Local Environments

In addition to the organic molecules, we also applied the ANN search method on metallic bulk systems with the Materials Project dataset.[11] Similar to the QM9 case, the whole Materials Project database was split into the training and validation sets randomly with a ratio of 8:2. A GemNet model was trained on the training set. The energy MAE on the training and validation set was 0.62 eV and 1.42 eV respectively. There is an apparent gap between the accuracy of the GemNet model on the training and validation set. We attribute this gap to the configuration extrapolation in the validation set. At the point we stop the training, there was no increase of the MAE on the validation set along with the training steps, which implied the model was not located in the overfitting region. We then used the trained model to search for similar molecules in the Materials Project training dataset.

With an example below, we show that the neural network embedding-based ANN search method is able to find similar bulk environments in this database. The query example is an oxygen atom in a $Al_2Cu_3O_6$ bulk cell. The query and searched atoms are shown in Figure 6.6. The distances of the searched atoms to the query atom and their ranks are shown in Table 6.1. The query oxygen atom is the atom 9 in Figure 6.6 a, which is closely neighboring to a copper atom (atom 3). There is also an aluminum atom (atom 0) near the query oxygen atom. These three atoms form an angle around 135° with the aluminum and copper atoms at two ends and the oxygen atom at the vertex. There is also another oxygen atom (atom 5) at the opposite position to the query oxygen atom across the copper atom. These geometrical features also appear in the searched atoms in Figure 6.6 b (atoms 10, 13, 15, 17) and Figure 6.6 c (atoms 6, 8). Periodic conditions should be considered when examing the geometrical similarity for atom 10 and atom 13 in Figure 6.6 b. In Figure 6.6 d, atom 17 is also the searched atom and it is ranked as 7th in all atomic environments although its neighboring environment looks not so similar to the query atom. This is because there are no more similar atoms like the previous ones in the remaining pool. As shown in Table 6.1, the Euclidean distance of the atom embeddings for the searched atoms in Figure 6.6 b and Figure 6.6 c to the query oxygen atom ranges between 0.06 - 0.11. This distance jumps to 0.17 for atom 17 in Figure 6.6 d. The distance of the atom embeddings also implies that atom 17 of Figure 6.6 d is not so similar to the query oxygen atom from the view of the GemNet model. It is also worth noting that during the searching process, we did not explicitly restrict the searching pool to be oxygen atoms. This atomic identity feature was already encoded into the GemNet atomic embeddings, and this is why the retrieved atoms are all oxygen atoms in Figure 6.6 although with different local environments. For more examples, please refer to Appendix F.

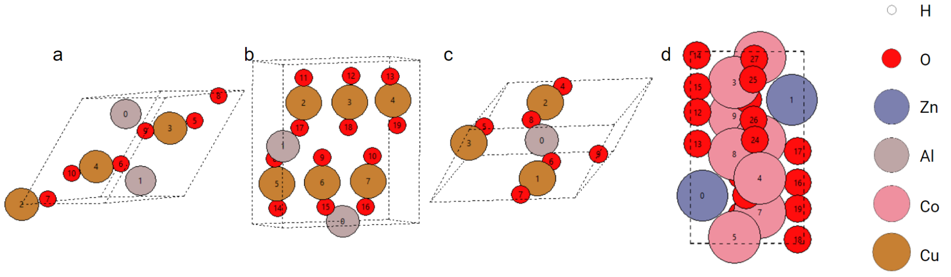

Figure 6.6: Top 7 nearest atoms to the query oxygen atom in the Materials Project dataset. Atom 9 in figure a is the query atom. Atom 10, 13, 15, and 17 in figure b, atom 6, 8 in figure c, and atom 17 in figure d are the searched atoms.

Table 6.1: Euclidean distances between the GemNet embeddings of the searched atoms (excluding the query atom itself) and the query atom. Configuration ID b, c and d correspond to the configurations b, c and d shown in Figure 6.6.

Rank	Configuration ID	Atom ID	Distance
1	b	13	0.06
2	c	6	0.07
3	c	8	0.07
4	b	17	0.08
5	b	15	0.09
6	b	10	0.11
7	d	17	0.17

6.3.3 ANN Search for Surfaces

In this section, we move on to a more complicated system, metallic surfaces with adsorbates. Relaxed configurations in the OC20 dataset were used in this case. There are more than 460k configurations in the training set and about 24k configurations in the validation set. A GemNet model was trained on the training set with the energy MAEs of 0.82 eV and 0.92 eV on the training and validation sets respectively. Atom embeddings were generated by this GemNet model to be searched during the query events. We illustrate the application of the GemNet embeddings to search for similar adsorption configurations via two examples. The first example is an oxygen atom adsorbed on a tilted hollow site consisting of two Pd atoms and one Ag atom. The local configurations are shown in Figure 6.7 a. During the searching, we did not explicitly provide information like element types of the central and surrounding atoms. Only the GemNet embeddings were used to measure the similarities. According to the search result in Figure 6.7, this elemental information as well as the geometrical pattern of the adsorption site have already been encoded into the GemNet embeddings. On the one hand, the retrieved atoms are all oxygen atoms. On the other hand, the adsorption sites are all hollow sites with two Pd atoms. In addition to these apparent features, we also present the similarity between the query atom and the searched atoms via the density of states projected to these atoms (ADOS). The ADOS data was calculated by the Vienna Ab initio Simulation Package (VASP).[53] The ADOS data is shown in Figure 6.8. For the searched atoms, their ADOS curves almost overlap with the query oxygen atom. Their cosine similarities are all above 0.6. As a comparison, we show the ADOS data of four randomly selected oxygen atoms (see detailed configurations in Figure F.6) in the OC20 dataset in Figure 6.9. These random atoms have different ADOS from the query atom and their cosine similarities are generally smaller than the searched ones. This example shows that the GemNet embeddings are able to search for elementally and geometrically similar local environments for a single atom adsorbed on metallic surfaces. These similarities also lead to the similarity in the density of states (DOS). This example also implies a potential application of searching for similar local structures using the projected DOS with vector search methods. Since similar DOS suggests similar elemental and geometrical environments, as well as potentially similar chemical properties. Storing DOS data when building a database with some extra recourses would be beneficial to this kind of application in the future.

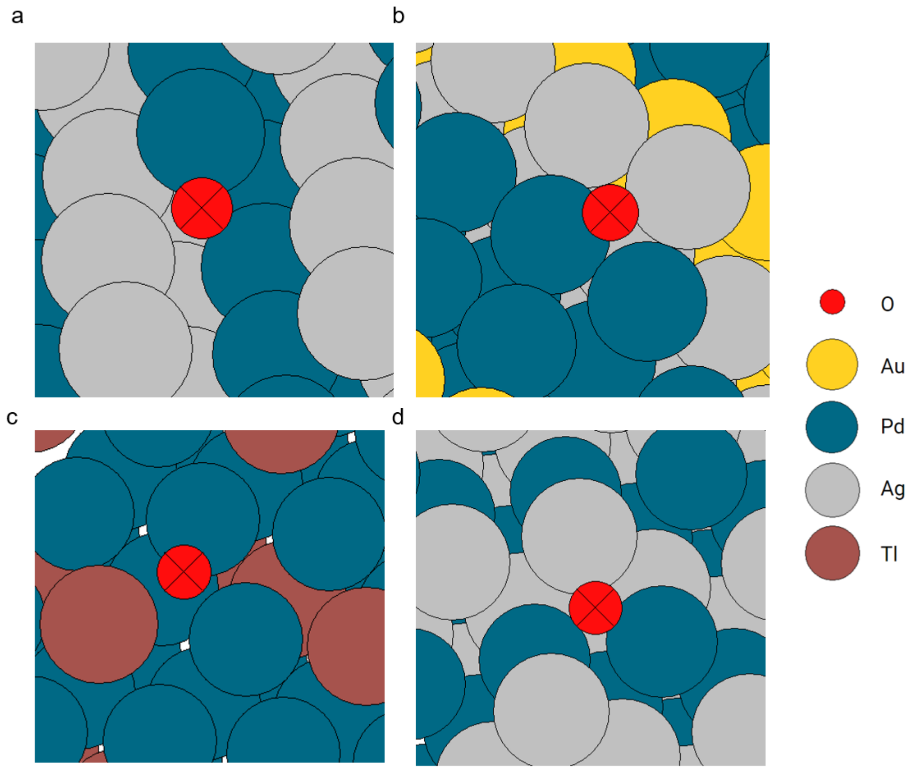

Figure 6.7: Local configurations of the query and searched atoms (marked as crossed). Configuration a is the query oxygen and configurations b to d are the retrieved atoms.

Next, we demonstrate that our method not only works for simple atoms like oxygen, but also for larger adsorbates like acetylene. In the OC20 dataset, we search for similar atoms with embeddings similar to that of the two carbon atoms in the query acetylene. We did not include the searching for similar atom

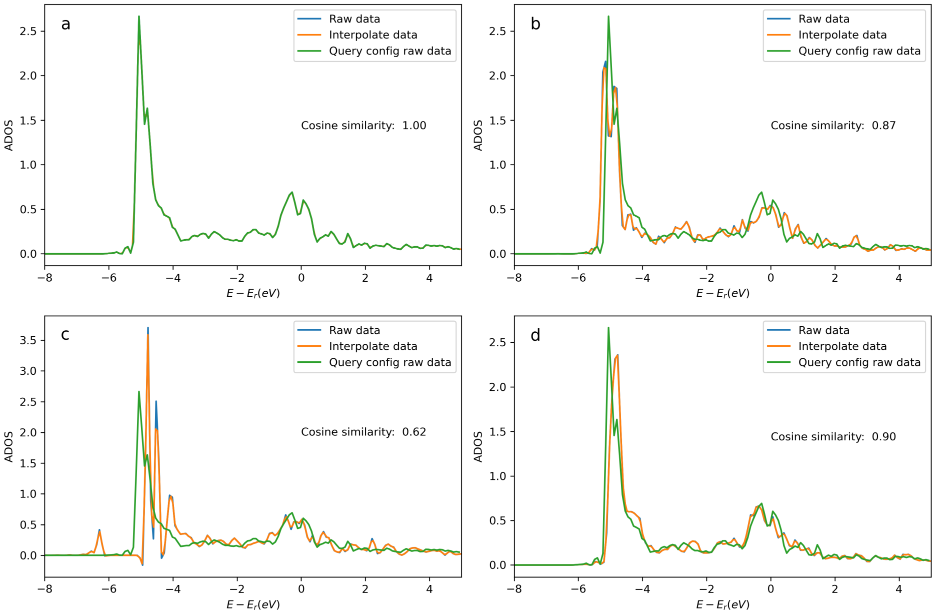

Figure 6.8: Density of states projected onto the p-orbital of the query and searched oxygen atoms. Figures a to d correspond the configurations a to d in Figure 6.7. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data.

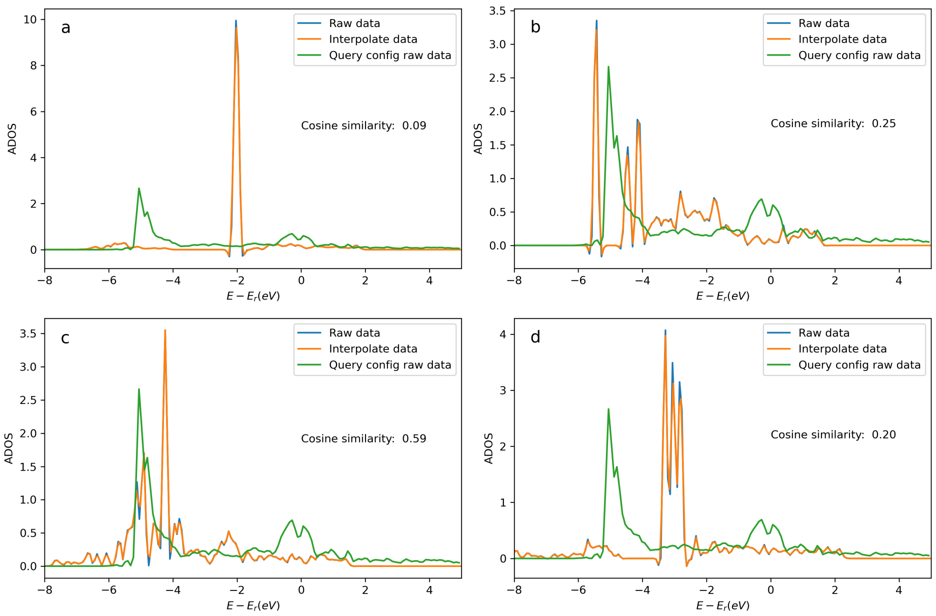

Figure 6.9: Density of states projected onto the p-orbital of four randomly selected oxygen atoms. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data.

embeddings to the hydrogen atoms since the carbon atom is the main feature of acetylene. Ignoring hydrogen atoms is also adopted in other molecular fingerprints like the SMILES.[129] The query and searched configurations are shown in Figure 6.10. The query object is an acetylene molecule adsorbed on a hollow site formed by three Cu atoms. The retrieved adsorption configurations are similar to the query one. The first point is that the searched adsorbates are all acetylene without explicitly setting the search pool to be acetylene molecules. The second point is that the adsorption sites of the top two results (Figure 6.10 b and c) are hollow sites with three Cu atoms which are the same as the query one. This is not so clear in Figure 6.10 b, but Figure F.7 b shows more details of the local structure. Similar to the oxygen case, we also compare the ADOS of the query and searched configurations. Figure 6.11 shows the ADOS of a selected carbon atom of acetylene molecule in these systems. We can see the ADOS of the searched configurations are similar to the query one, and their cosine similarities are all above 0.65 which is much higher than that of four randomly selected configurations shown in Figure 6.12. The similarities in terms of the adsorbates, adsorption sites, and DOS between the query and searched configurations suggest that our method also works well for adsorption systems with large adsorbates.

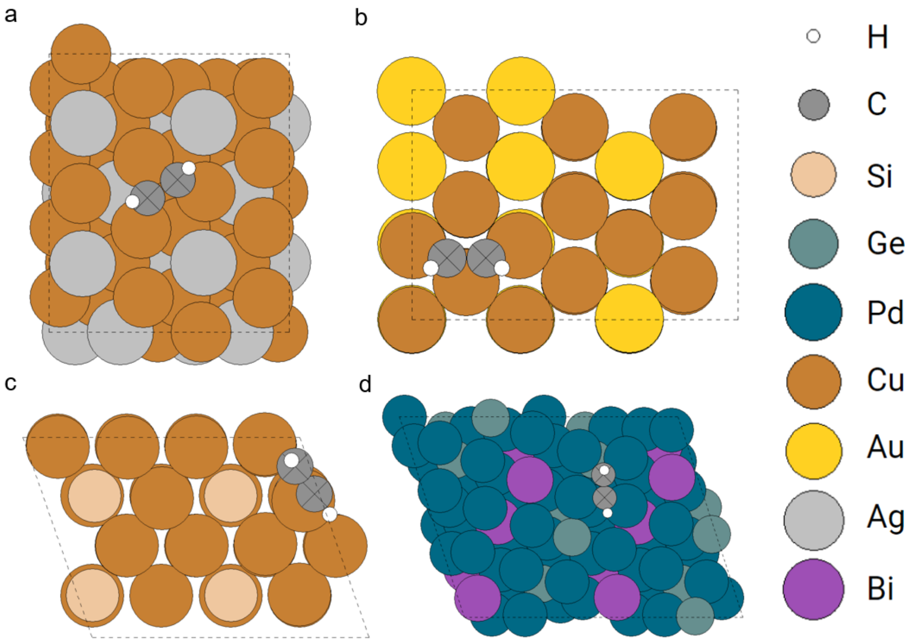

Figure 6.10: Configurations of the query (config. a) and top 3 retrieved acetylene adsorption configurations (config. b to d). The query and matched carbon atoms are marked as crossed.

6.4 Conclusion

In this chapter, we went through the applications of the neural network embedding-based approximate nearest neighbor search framework to search for similar molecular (sub)structures in large databases. We discussed two components of this framework: the neural network embedding and the approximate nearest neighbor search. The former enables us to represent

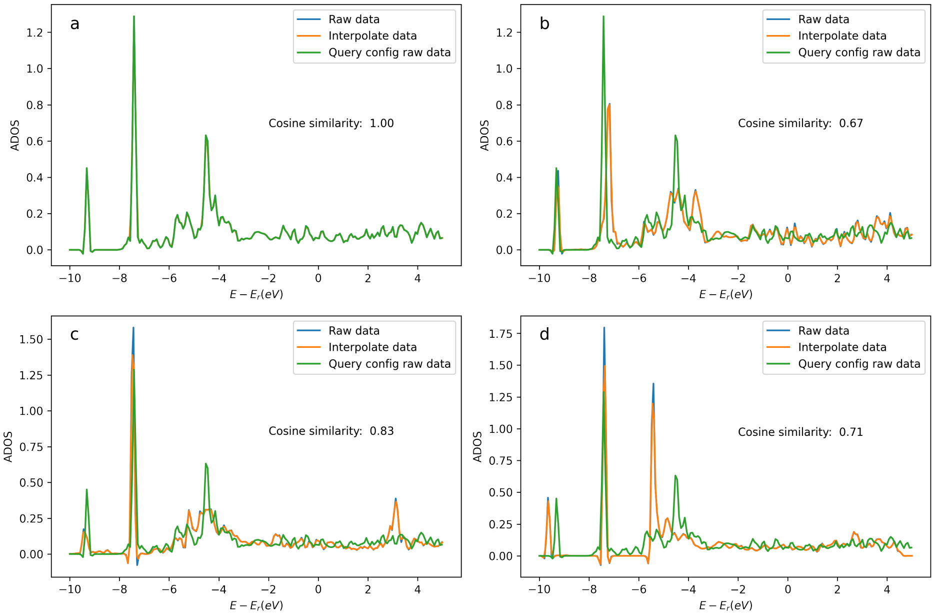

Figure 6.11: Density of states projected onto the p-orbital of the selected query and searched carbon atoms. Figures a to d correspond the configurations a to d in Figure 6.10. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data.

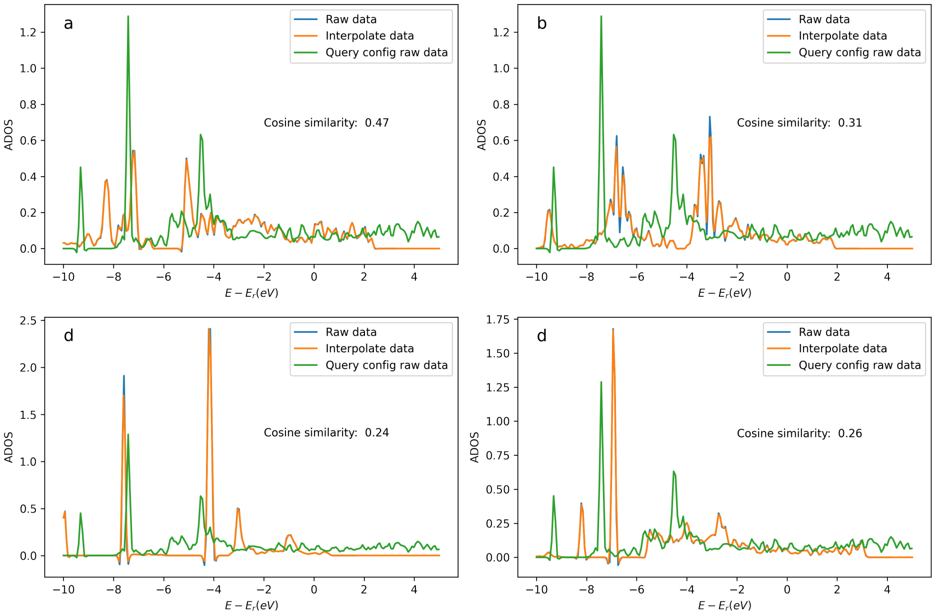

Figure 6.12: Density of states projected onto the p-orbital of the carbon atoms in four randomly selected acetylene adsorption configurations. The blue curve is the original DOS energy and density data. The orange line is the linearly interpreted data from the original DOS data to make the energy stamps to be the same across the configurations. Cosine similarity was calculated using the interpreted data.

local atomic configurations precisely. The latter provides us with a fast and cheap way to search for neighboring real vectors in a large database. In our work, we used GemNet and Faiss as the neural network model and the ANN search implementation. However, the usage of this framework is not limited to these two examples. Any molecular descriptors and other deep learning models can be used to generate the representing vectors for atoms or molecules. But factors like scaling, expressiveness, and computational cost need to be considered when we make a choice. Also, any vector search methods can be used as the search engine. A cheap, fast, and user-friendly package would be favorable. Then, with examples across organic molecules, metallic bulks, and metallic surfaces with adsorbates, we illustrated the ability of this framework to find similar configurations in different databases. We presented the similarities from different aspects: elemental types, geometrical features, energetic relevance, and the density of states. These examples also reflect the generalizability of this framework for different types of molecular systems.

7 Evaluation of the Degree of Rate Control via Automatic Differentiation

The results in this chapter are adopted from our published work in Ref. 152.

7.1 Introduction

In the chapters above, we discuss the applications of ML potentials to accelerate molecular simulations and the search for similar atomistic structures. However, reactions in the real world are usually too complicated to include every detail in simulations. Therefore, focusing on some significant steps of the reactions could save lots of computational costs while keeping the main features of the reactions. For example, we can allocate more computational resources to the rate-determining steps in a complex reaction system to make sure these important steps are calculated as accurately as possible. Among various tools, the degree of rate control (DRC) is a versatile concept proposed by Campbell to measure the kinetic contribution of each reaction step to a target reaction rate.[153,154] There are many applications of this concept in the research areas of catalysis and microkinetic modeling.[155–159] These applications include the investigation of the relationship between the reaction conditions and the rate-determining steps,[155] using the degree of rate control to screen catalysts,[157] exploring the mechanism of reactions,[155,156] and so forth. In this chapter, we move forward with the topic that how to evaluate the DRC accurately and efficiently.

We firstly present the mathematical forms of the DRC. The kinetic DRC is defined as the derivative of the natural log of the rate with respect to the standard-state free energy of the transition state or the forward kinetic rate constant given the corresponding equilibrium constant is fixed:

$$X_{RC,i} = \frac{k_i}{r}\left(\frac{\partial r}{\partial k_i}\right)_{k_{j\neq i},K_i} = \left(\frac{\partial \ln r}{\partial \ln k_i}\right)_{k_{j\neq i},K_i} = \left(\frac{\partial \ln r}{\partial\left(\frac{-\Delta G_i^{0,TS}}{RT}\right)}\right)_{k_{j\neq i},K_i} \tag{7.1}$$

where r is the net rate of the production of interest, k_i is the forward kinetic constant of step i and K_i is the equilibrium constant of step i. The thermodynamic version of the DRC extends the application of this concept to the standard-state free energy of the intermediate species in the reaction system, which is mathematically defined as

$$X_{TRC,n} = \left(\frac{\partial \ln r}{\partial \frac{-G_n^0}{RT}}\right)_{G^0_{m\neq n},G_i^{0,TS}} \tag{7.2}$$

where G_m^0 is the standard-state free energy of species m, $G_i^{0,TS}$ is the transition-state energy of step i. Several variants of the DRC have been proposed to fulfill different purposes, including the DRC for selectivity,[160] for transient kinetics,[154,161,162] and for uncertain parameters.[163]

From a practical perspective, a simple and common way to assess the DRC is using finite difference (FD) approximations for the derivatives.[161,162,164] These are fairly straightforward to implement and only require a few more lines of code in addition to the original simulation code. Mathematically, the centered difference approximation is formulated to approximate the derivatives in

$$X_{RC,i} = \frac{k_i}{r}\left(\frac{\partial r}{\partial k_i}\right)_{k_{j\neq i},K_i} \approx \frac{k_i}{r}\frac{\delta r}{\delta k_i} \tag{7.3}$$

where δk_i is the perturbation applied on k_i and δr is the change of the net reaction rate resulted from the perturbation of k_i. Although the finite difference approximation is a popular choice because of its ease of understanding and implementation, one must be careful to choose the magnitude of the perturbation. The change of the kinetic constant should be small enough such that the response of the net reaction is linear, but not so small that goes beyond the precision limitation of the computer.[161]

Formally, there is a trade-off between the truncation error and the rounding error in this approach. The scale of the truncation error is $O(\delta)$ for the first-order derivative, which prefers small δ. However, when the δ is smaller than the precision limitation, then the value is no longer reliable. In real practice, the process of finding the suitable step size takes much time. Since for each try, we need to run the finite difference once. In addition, FD requires $O(p)$ rounds of function calls or forward simulations to get the derivatives of p parameters, which is time-consuming for large size of parameters. For example, Bjarne et al. took 300 CPU-hours to conduct the sensitivity analysis when investigating the mechanism of CO_2 hydrogenation on Ni(111) using the finite difference method.[164]

To avoid these issues, sensitivity analysis methods like the direct sensitivity analysis and the adjoint sensitivity analysis[165] are usually adopted by common differential equation solvers[166–168] to provide the derivative of the numerical solution to the parameters of differential equations. The direct sensitivity analysis converts the solution sensitivity with respect to the parameters of differential equations into n extra (number of parameters) differential equations, which are solved simultaneously with the original differential equations. The adjoint sensitivity analysis requires the definition of some scalar functional of the numerical solution and the parameters. Then the sensitivities are given by an integration. For the mathematical details, one could refer to the introduction of these two methods in the chemical kinetic systems by Sandu.[169]

To be applied in the calculation of the DRC, the partial derivatives of the reaction rate to the concentrations and the kinetic parameters still need to be solved, since the sensitivity analysis methods only provide the derivatives of the numerical solution (commonly the concentration or coverages in the chemical kinetic systems) to the parameters. One of the tools to integrate these derivatives is automatic differentiation (AD),[170,171] which automatically evaluates the derivatives of a function that is built on a set of atomic operations and functions (e.g., addition, multiplication, exp, log, etc.). The derivatives are generated by chain rule based on the derivatives of these elementary operations. For the DRC case, the numerical integration and post functions from the kinetic constants to the reaction rate could be regarded as a sequence of the atomic operations and functions. Thus, the derivatives of the reaction rate to the kinetic constants could be evaluated by the chain rule applied on this sequence of basic operations.

In this chapter, we adopt AD in the evaluation of the DRC, which we show provides more accurate and faster evaluation than FD. In section 7.2.1, we introduce the working mechanism of the AD to obtain the derivatives of a function automatically. In section 7.3, we illustrate three case studies to check the correctness of the AD and to show its advantages over FD. Specifically, we take the hypothetical reaction scheme from Foley's non-steady DRC work as a simple case.[162] For a slightly more complicated example, we use the water-gas shift reaction adapted from Motagamwala's maximum rate work.[172] Next, we show the application of AD to calculate the DRC of a more complex mechanism, the propylene partial oxidation on

Cu(100),[173] which involves 17 elementary steps. Finally, we demonstrate how to use AD in the utilization of the distributed evaluation of local sensitivity analysis (DELSA)[174] to deal with the uncertain range of the kinetic parameters, which is a hybrid local-global sensitivity analysis method to identify the important parameters and the importance distribution over an uncertain range.

7.2 Methods

7.2.1 Automatic Differentiation

Automatic differentiation has two modes to calculate the derivatives: the forward mode and the reverse mode.[170,171] In the forward mode, the computational graph starts with the input variables, and grows along the elementary operations and functions applied on the input variables. During the forward expansion, the function evaluations and derivative calculations take place simultaneously. In the reverse mode, there are two rounds of evaluations. The first one is the forward evaluation of the function values starts from the input variables. The second round is the back-propagation of the derivatives from the function output to the input.

We use the example $y = \ln(3x_1 - 2x_2) + x_1x_2$ with $(x_1, x_2) = (1, 1)$ as a prototype to illustrate the workflow of the AD. The computational graph is shown in Figure 7.1. The details of the forward AD and the reverse mode are shown in Table 7.1 and Table 7.2. The comparison between the forward and the reverse mode is clear in Table 7.1 and Table 7.2. In the forward mode, all derivatives of the intermediate and the final results with respect to a specified input variable (x_1 in this case) are calculated in one forward propagation. However, in the reverse mode, the derivative of a specified scalar output with respect to all the intermediate and input variables are obtained in one backward propagation. Thus, the forward mode is suitable for the functions with fewer input dimensions, while the reverse mode is more efficient for the functions with fewer output dimensions.

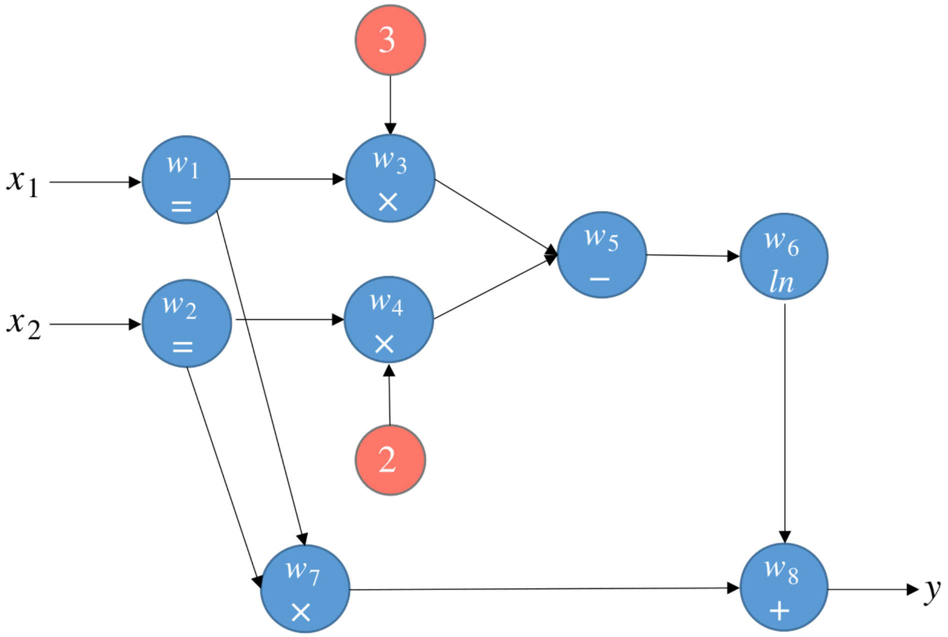

Figure 7.1: Computational graph for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Each blue node contains the corresponding variable and the operation that applied on its parent node(s). For instance, w_3 is the result of the multiplication between a constant 3 and another node variable w_1.

Similar to the example above, AD could also be applied on the numerical integration of the ODE systems. Consider the initial-value problem

$$\dot{\boldsymbol{\theta}} = \boldsymbol{f}(\boldsymbol{\theta}, \boldsymbol{k}, \boldsymbol{K}, t),\ \boldsymbol{\theta}(\boldsymbol{k}, \boldsymbol{K}, t = 0) = \boldsymbol{\theta}_0 \tag{7.4}$$

where $\boldsymbol{\theta}$ is the state vector, $\boldsymbol{f}$ is the state derivative vector, $\boldsymbol{k}$ and $\boldsymbol{K}$ are the parameters of this ODE system, and $\boldsymbol{\theta}_0$ is the initial state vector. For the sake of simplicity, we assume the explicit forward Euler method is used to solve this ODE. Thus, the update equation is

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + h\boldsymbol{f}(\boldsymbol{\theta}_n, \boldsymbol{k}, \boldsymbol{K}, t_n) \tag{7.5}$$

Table 7.1: Forward AD for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Left side shows the forward function evaluations. Right side shows the derivative of $\frac{\partial y}{\partial x_1}$. Head dot means $\frac{\partial}{\partial x_1}$

Function evaluation	Derivative calculation (from top to bottom)
$w_1 = x_1 = 1$	$\dot{w}_1 = \dot{x_1} = 1$
$w_2 = x_2 = 1$	$\dot{w}_2 = \dot{x_2} = 0$
$w_3 = 3w_1 = 3$	$\dot{w}_3 = 3\dot{w}_1 = 3$
$w_4 = 2w_2 = 2$	$\dot{w}_4 = 2\dot{w}_2 = 0$
$w_5 = w_3 - w_4 = 1$	$\dot{w}_5 = \dot{w}_3 - \dot{w}_4 = 3$
$w_6 = \ln w_5 = 0$	$\dot{w}_6 = \frac{1}{w_5}\dot{w}_5 = 3$
$w_7 = w_1w_2 = 1$	$\dot{w}_7 = w_1\dot{w}_2 + \dot{w}_1w_2 = 1$
$w_8 = w_6 + w_7 = 1$	$\dot{w}_8 = \dot{w}_6 + \dot{w}_7 = 4$
$y = w_8 = 1$	$\dot{y} = \dot{w}_8 = 4$

Table 7.2: Reverse AD for the example $y = \ln(3x_1 - 2x_2) + x_1x_2$. Left side shows the forward function evaluations. Right side shows the back-propagation of the derivative of $\frac{\partial y}{\partial x_1}$.

Function evaluation	Derivative calculation (from bottom to top)
$w_1 = x_1 = 1$	$\frac{\partial y}{\partial x_1} = \frac{\partial y}{\partial w_1}\frac{\partial w_1}{\partial x_1} = 4$
$w_2 = x_2 = 1$	$\frac{\partial y}{\partial x_2} = \frac{\partial y}{\partial w_2}\frac{\partial w_2}{\partial x_2} = -1$
$w_3 = 3w_1 = 3$	$\frac{\partial y}{\partial w_1} = \frac{\partial y}{\partial w_1} + \frac{\partial y}{\partial w_3}\frac{\partial w_3}{\partial w_1} = 4$
$w_4 = 2w_2 = 2$	$\frac{\partial y}{\partial w_2} = \frac{\partial y}{\partial w_2} + \frac{\partial y}{\partial w_4}\frac{\partial w_4}{\partial w_2} = -1$
$w_5 = w_3 - w_4 = 1$	$\frac{\partial y}{\partial w_3} = \frac{\partial y}{\partial w_5}\frac{\partial w_5}{\partial w_3} = 1$
	$\frac{\partial y}{\partial w_4} = \frac{\partial y}{\partial w_5}\frac{\partial w_5}{\partial w_4} = -1$
$w_6 = \ln w_5 = 0$	$\frac{\partial y}{\partial w_5} = \frac{\partial y}{\partial w_6}\frac{\partial w_6}{\partial w_5} = 1$
$w_7 = w_1w_2 = 1$	$\frac{\partial y}{\partial w_1} = \frac{\partial y}{\partial w_7}\frac{\partial w_7}{\partial w_1} = 1$
	$\frac{\partial y}{\partial w_2} = \frac{\partial y}{\partial w_7}\frac{\partial w_7}{\partial w_2} = 1$
$w_8 = w_6 + w_7 = 1$	$\frac{\partial y}{\partial w_6} = \frac{\partial y}{\partial w_8}\frac{\partial w_8}{\partial w_6} = 1$
	$\frac{\partial y}{\partial w_7} = \frac{\partial y}{\partial w_8}\frac{\partial w_8}{\partial w_7} = 1$
$y = w_8 = 1$	$\frac{\partial y}{\partial w_8} = \frac{\partial y}{\partial y}\frac{\partial y}{\partial w_8} = 1$

where h is the step size. Therefore, to get the derivative of $\boldsymbol{\theta}_{n+1}$ with respect to $\boldsymbol{k}$, we have

$$\frac{d\boldsymbol{\theta}_{n+1}}{d\boldsymbol{k}} = \frac{d\boldsymbol{\theta}_n}{d\boldsymbol{k}} + h\frac{d\boldsymbol{f}(\boldsymbol{\theta}_n, \boldsymbol{k}, \boldsymbol{K}, t_n)}{d\boldsymbol{k}} \tag{7.6}$$

In the forward AD, the derivative of the terms at the RHS of Eq 7.6 with respect to $\boldsymbol{k}$ is propagated to the LHS. The computational complexity to get the derivatives are in $O(np)$ where n is the number of time steps and p is the number of parameters.[175] In the reverse AD, the derivative of $\boldsymbol{\theta}_{n+1}$ with respect to the LHS is back propagated to the variables in the RHS. For this mode, the computational complexity is $O(n+p)$.[175] Although reverse AD has lower scale of computational cost, its advantage only appears in using adjoint method to calculate the derivatives for large ODE systems. This is because the current tape-based implementation of reverse-mode AD is optimized for the usage in deep learning models with intensive matrix multiplications, which is not optimally suitable for differential equations that tend to be defined by nonlinear functions with scalar operations.[175]

In both modes, the derivatives of the ODE solution with respect to the parameters could be obtained automatically. Even if there are post operations applied on the ODE solution (e.g., conversion from the concentrations to reaction rates), the derivatives of the final results to the parameters can be calculated in an end-to-end way as long as these operations are in the same computational graph. We note that some ODE solvers may have the step size (h) that is dependent on the parameters, in this case, the derivative of the step size with respect to the parameters are enforced to be zero during the implementation.[176]

In the past decade, various AD packages have been developed for applications in machine learning.[171] Typical examples include Pytorch[25] and

Jax[177] in Python, and ForwardDiff[178] in Julia. To integrate the AD into the solution of an ODE system, the ODE solver should be compatible to these AD packages and the operations and functions in the ODE solver should be included in the computational graph of the AD packages. There are several modules satisfying these requirements such as the torchdiffeq[179] for Pytorch, the DifferentialEquations[180] for ForwardDiff and, the PyBaMM[181] for Jax. In the work of this chapter, we use the `ForwardDiff` and `DifferentialEquations` in the Julia language since they provides various differentiable ODE solvers that could handle various non-stiff and stiff problems.

7.2.2 Distributed Evaluation of Local Sensitivity Analysis

Distributed evaluation of local sensitivity analysis (DELSA)[174] is a hybrid local-global sensitivity analysis method to measure the distribution of parameter sensitivity across the parameter space with low computational cost. Basically, it is an extension to local sensitivity analysis that takes the uncertainty of the parameters into account. The importance of a parameter over the parameter space is measured by a local sensitivity statistic like the median or the mean of a set of samples drawn from the parameter space. Local sensitivity analysis is then conducted on each parameter sample. This makes DELSA much cheaper to get the detailed distribution of the importance over the parameter space than the Sobol's indices which is a popular global sensitivity analysis method based on the variance decomposition.[182,183] Mathematically, the first-order sensitivity measure for j^{th} parameter at sample i is defined as

$$S_{ij} = \frac{\left|\frac{\partial y_i}{\partial \theta_{ij}}\right|^2 s_j^2}{V(y_i)} \tag{7.7}$$

where S_{ij} is the sensitivity measure for the parameter θ_{ij} at sample i, y_i is the model output or an element of the model output of sample i, s_j is the prior variance of θ_j, and $V(y_i)$ is the total variance of the model output y_i, which can be evaluated using the first-order-second-moment method:[184]

$$V(y_i) = \left(\frac{\partial y_i}{\partial \boldsymbol{\theta}}\right)^T \left(\boldsymbol{X}^T \boldsymbol{\omega} \boldsymbol{X}\right) \left(\frac{\partial y_i}{\partial \boldsymbol{\theta}}\right) \tag{7.8}$$

where $\boldsymbol{X}$ is a matrix of $(n_{obs} + n_{prior})$ rows and n_{param} columns. n_{obs}, n_{prior}, n_{param} are the number of observations, the number of prior information equations and the number of parameters respectively. In the application of DELSA, there is no observation and the prior information is the variance of each parameter. Therefore, $n_{obs} = 0$ and $n_{prior} = n_{param}$, and each row of $\boldsymbol{X}$ has zeros except for one which indicates the parameter associated with the prior information. $\boldsymbol{\omega}$ is a diagonal matrix contains the reciprocal of the prior variance of each parameter. More details of the structure of $\boldsymbol{X}$ and $\boldsymbol{\omega}$ could refer to the related works.[174,184,185]

Intuitively, the sensitivity measure S_{ij} captures the contribution of θ_{ij} to the total uncertainty of the output y_i. The parameters with more contribution to the uncertainty are considered more important. This sensitivity measure gives similar importance evaluation to the Sobol's method[182] for the uncertain parameters in the previous reports.[174,186] In addition, DELSA could be easily integrated to the original local sensitivity implementation which is the DRC with automatic differentiation in our work.

7.3 Results

We first show that AD can reproduce the DRC results of previous reports[162,172] in the first two simple cases. Then, we compare the performance of the FD and the AD on another more complicated reaction mechanism of propylene oxidation.[173] Lastly, we show the application of AD to deal with uncertain parameters by manually introducing an uncertain range for the kinetic constants in the propylene oxidation case. We note that the transient DRC below corresponds to the case that we perturb the parameters at zero time, and then check how the response variable is changed at time t. The reason for this interpretation is that when one derives the analytical transient solution, e.g. $f(k, t)$ then the derivative $\partial f/\partial k \approx [f(k + \delta k, t) - f(k, t)]/\delta k$ where $f(k + \delta k, t)$ represents the solution accumulated from $t = 0$ to t with $k = k + \delta k$. We show that AD in this case is fully consistent with that interpretation. There is another definition of the transient DRC proposed in Campbell's works.[161,187] Under this definition, people need to simulate the reactions using the original parameters from $t = 0$ to t, then perturb the kinetic parameters and simulate the reactions forward with a tiny time step δt to reach to a new quasi-steady state to get the DRC at t. This definition corresponds to the interpretation that how the perturbation of the kinetic parameters at t affect the reaction rate at near future which can be approximately viewed at time t. These two definitions can be used for different purposes. In the cases below, we use the former definition to illustrate the application of the AD in DRC calculation.

7.3.1 Case I: Hypothetical Two-Step Reaction

In this case, we investigate the transient DRC for a hypothetical reaction. We first consider the hypothetical reaction mechanism in Table 7.3 and the derivation of analytical solutions for the rate and DRC.[162]

which leads to the following differential equations (Eqs 7.9 - 7.10):

Table 7.3: Hypothetical Two-Step Catalytic Reaction (case I).[162]

Step id	Elementary Step	k_i	k_{-i}
1	$A + * \leftrightharpoons A*$	10^{-5}	0
2	$A * + B \rightarrow C + *$	1	NA

$$\frac{1}{L}\frac{\theta_{A*}(t)}{dt} = k_1 a_A(t)\theta_*(t) - k_{-1}\theta_{A*}(t) - k_2 a_B(t)\theta_{A*}(t) \tag{7.9}$$

$$\frac{1}{L}\frac{\theta_*(t)}{dt} = -k_1 a_A(t)\theta_*(t) + k_{-1}\theta_{A*}(t) + k_2 a_B(t)\theta_{A*}(t) \tag{7.10}$$

where θ_is are the coverages, a_is are the thermodynamic activities and L is the number of active sites. The net reaction rate is defined as the rate to produce C per active site:

$$\frac{r_C(t)}{L} = k_2 a_B(t)\theta_{A*}(t) \tag{7.11}$$

The analytical solution for $r_C(t)/L$ is

$$\frac{r_C(t)}{L} = \frac{k_1 a_A k_2 a_B}{k_1 a_A + k_{-1} + k_2 a_B}\left(1 - e^{-(k_1 a_A + k_{-1} + k_2 a_B)t}\right) + k_2 a_B \theta_{A*,0} e^{-(k_1 a_A + k_{-1} + k_2 a_B)t} \tag{7.12}$$

where $\theta_{A*,0}$ is the coverage of $A*$ at $t = 0$.

According to the parameters in the original paper,[162] $a_A = 1$, $a_B(t < 0) = 1$ and $a_B(t \geq 0) = 3$. Since k_{-1} is set as 0 and $k_1 \ll k_2$, thus

$$\frac{r_C(t)}{L} = (k_1 a_A)\left(1 - e^{-(k_2 a_B)t}\right) + k_2 a_B \theta_{A*,0} e^{-(k_2 a_B)t} \tag{7.13}$$

which has the corresponding DRC as:

$$X_{RC,1} = \frac{k_1 L}{r_C(t)} a_A \left(1 - e^{-(k_2 a_B)t}\right) \tag{7.14}$$

$$X_{RC,2} = \frac{k_2 L}{r_C(t)} \left[k_1 a_A a_B t e^{-(k_2 a_B)t} + a_B \theta_{A*,0} e^{-(k_2 a_B)t} - k_2 a_B^2 \theta_{A*,0} t e^{-(k_2 a_B)t}\right] \tag{7.15}$$

We note that this transient DRC is based on the definition of Eq 7.1, which is different from the modified version proposed by Bhan.[162] Upon Eq 7.14 - 7.15, notice that at $t = 0^+$, $X_{RC,1}$ is 0 and $X_{RC,2}$ is 1. As $t \to \infty$, $X_{RC,1}$ grows to 1 and $X_{RC,2}$ decreases to 0. The solutions from the analytical form and the AD are shown in Figure 7.2, in which the solution of the AD matches the analytical solution over the whole time range. The DRC in the Figure 7.2 does not obey the sum of kinetic DRC equaling one for all time (only at steady state) because the rate during the transient process also depends on both the time and the kinetic constants. This issue is discussed in more details in the non-steady DRC paper 162. The main message conveyed by this simple example is that the AD is practically equivalent to having the analytical derivatives to evaluate the transient DRC defined by Eq 7.1, but without the need to analytically derive the expressions or to approximate them with finite differences.

7.3.2 Case II: Redox Mechanism for Water-gas Shift

Case I was a hypothetical example and it only contained two steps with manually set kinetic parameters, which is relatively simple. In case II, we consider the DRC for a more complicated reaction mechanism for the water-gas shift reaction[172] which is listed in Table 7.4. The pressures for the gas-phase species P_{CO}, P_{H_2O}, P_{H_2}, P_{CO_2} are 0.07 atm, 0.21 atm, 0.38 atm, and 0.085

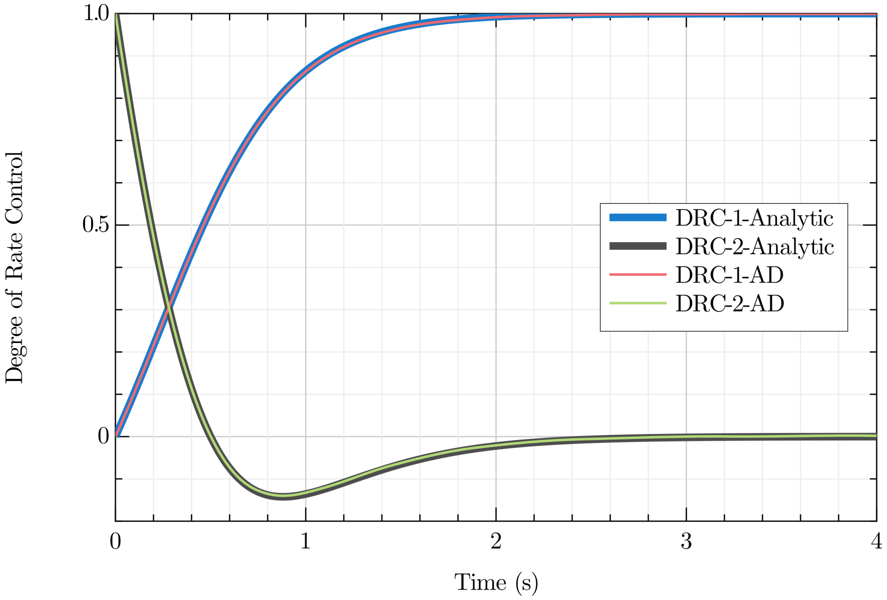

Figure 7.2: Degree of rate control for the hypothetical reaction of case 1 evaluated by analytic solution and automatic differentiation.

atm, respectively. The target net rate is of the production of the hydrogen, which could be expressed as (Eq 7.16):

$$r_{H_2} = k_7\theta_{H*}^2 - k_{-7}\frac{P_{H_2}}{P}\theta_*^2 \tag{7.16}$$

For this reaction system, the DRC evaluated by the AD are shown in Figure 7.3. Among the seven steps of this reaction mechanism, only two steps are identified as important to the net rate. Step 4 (dissociation of $OH*$) has a DRC of 0.88 while step 5 (formation of CO_2*) has a DRC of 0.12. The other 5 steps have little influence on the net rate of the whole reaction system. This result is consistent with the calculations of the original paper[172] and illustrates the reliability of the AD to evaluation the DRC of a moderately complicated reaction system.

Table 7.4: Redox mechanism for water-gas shift (case II).[172]

Step id	Elementary Step	$K_{eq,i}$	k_i
1	$CO + * \leftrightharpoons CO*$	2.15×10^2	1.33×10^8
2	$H_2O + * \leftrightharpoons H_2O*$	5.93×10^{-5}	2.01×10^{11}
3	$H_2O * +* \leftrightharpoons H * +OH*$	6.28×10^{-2}	2.64×10^6
4	$OH * +* \leftrightharpoons H * +O*$	1.18×10^{-5}	5.24×10^1
5	$CO * +O* \leftrightharpoons CO_2 * +*$	1.03×10^3	2.05×10^5
6	$CO_2* \leftrightharpoons CO_2 + *$	1.92×10^5	1.48×10^{12}
7	$2H* \leftrightharpoons H_2 + 2*$	4.50×10^1	5.32×10^2

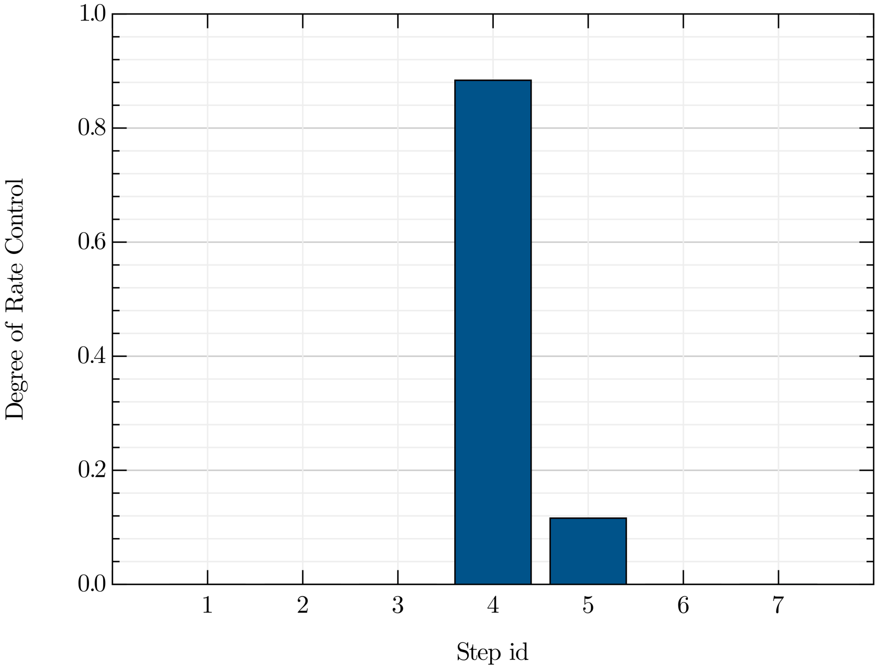

Figure 7.3: Degree of rate control for the water-gas shift reaction (case II).[172]

7.3.3 Case III: Propylene Partial Oxidation

Our third case is the transient DRC for the propylene partial oxidation[173] which could be an environmentally friendly route to produce propylene oxide. It has 17 elementary steps and the kinetic constants for each step are shown in Table 7.5. The partial pressures for the gas-phase species $P_{C_3H_6}$ and P_{O_2} are 0.1 bar and 0.05 bar respectively. The reaction simulation temperature is 500 K. The net rate of interest is the desorption of propylene oxide, which is the sum of the step 13 and 14 (Eq 7.17):

$$r_{PO} = k_{13}\theta_{PO_1*} - k_{-13}\frac{P_{PO}}{P}\theta_{v*} + k_{14}\theta_{PO_2*} - k_{-14}\frac{P_{PO}}{P}\theta_{v*} \tag{7.17}$$

The DRC results for this case are shown in Figure 7.4, where the adsorption of propylene as the III type and the desorption of PO_2* have the major positive contribution to the generation of $PO(g)$. These positive DRCs are reasonable since the adsorption of propylene provides the material to produce $PO(g)$ and the desorption of PO_2* directly generates $PO(g)$. On the other side, the desorption of O_2* hinders the production of $PO(g)$ since this step results in more $v*$ produced and more $O*$ and $O1*$ consumed, which benefits the negative direction of step 13 and 14.

We then compare the DRC calculation results using FD and AD in this case. The temperature is set as 350 K in this comparison. We evaluate the transient DRC defined by Eq 7.1 using the FD and the AD methods separately. The transient DRC for step 3, 4, 14, and 17 are shown in Figure 7.5. Different perturbation magnitudes (10^{-4}, 10^{-11}, 10^{-14}) are used in the FD method. There is an optimal choice of the perturbation size to reduce the truncation error and the rounding error. A large perturbation size suffers from the truncation error resulting from the nonlinearity of the target

Table 7.5: Elementary steps (step 1 to step 17 from top to bottom) for the propylene partial oxidation (case III).[173] E_a and E_a^{-r} are the activation energies for the forward and reverse directions. A and A^{-r} are the pre-exponential factor of the forward and reverse reactions. v represents the oxygen vacancy.

Elementary Step	$E_a(eV)$	A	$E_a^{-r}(eV)$	A^{-r}
$C_3H_6(g) + * \leftrightharpoons C_3H_6(I)*$	0.00	1.87×10^8	0.42	1.00×10^{13}
$C_3H_6(g) + * \leftrightharpoons C_3H_6(II)*$	0.00	1.87×10^8	0.57	1.00×10^{13}
$C_3H_6(g) + * \leftrightharpoons C_3H_6(III)*$	0.00	1.87×10^8	0.58	1.00×10^{13}
$C_3H_6(I) * +O* \leftrightharpoons C_3H_5 * +OH*$	0.36	8.03×10^{12}	1.40	8.93×10^{12}
$C_3H_6(II) * +O* \leftrightharpoons OMP_1*$	0.59	1.17×10^{13}	1.10	1.78×10^{13}
$C_3H_6(III) + O* \leftrightharpoons OMP_2*$	0.31	1.28×10^{13}	0.95	1.92×10^{13}
$C_3H_5 * +O* \leftrightharpoons C_3H_4O * +H_2O*$	0.30	1.14×10^{13}	1.19	7.14×10^{12}
$C_3H_5O * +OH* \leftrightharpoons C_3H_4O * +H_2O*$	0.54	1.09×10^{13}	1.72	1.54×10^{13}
$OMP_1* \leftrightharpoons PO_1 * +*$	0.62	4.10×10^{13}	0.83	1.53×10^{13}
$OMP_2* \leftrightharpoons PO_2 * +*$	0.77	3.24×10^{13}	0.89	1.31×10^{13}
$H_2O* \leftrightharpoons H_2O(g) + v*$	0.76	1.00×10^{13}	0.00	2.85×10^8
$C_3H_4O* \leftrightharpoons C_3H_4O(g) + v*$	0.10	1.00×10^{13}	0.00	1.62×10^8
$PO_1* \leftrightharpoons PO(g) + v*$	0.90	1.00×10^{13}	0.00	1.59×10^8
$PO_2* \leftrightharpoons PO(g) + v*$	0.96	1.00×10^{13}	0.00	1.59×10^8
$O_2 * +v* \leftrightharpoons O * +O1*$	0.00	1.00×10^{13}	1.43	1.00×10^{13}
$2O1* \leftrightharpoons O_2 * +*$	0.00	1.00×10^{13}	1.18	1.00×10^{13}
$O_2* \leftrightharpoons O_2(g) + *$	1.36	1.00×10^{13}	0.00	2.14×10^8

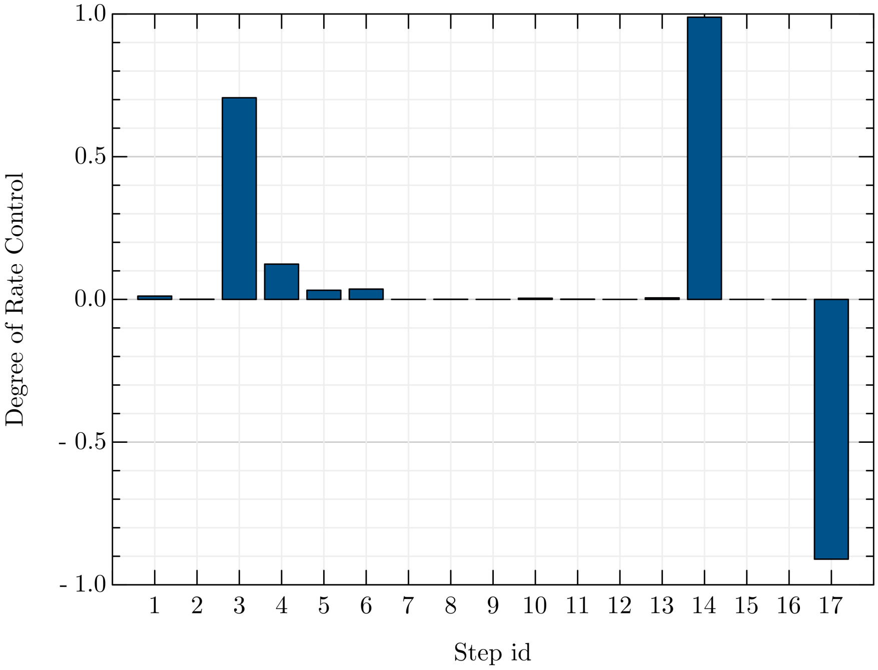

Figure 7.4: Degree of rate control for the propylene partial oxidation reaction (case III).[173]

function, which is the transient part from $t = 0$ to $t = 5$. Too small of a perturbation makes the solution affected by the rounding error due to the limited precision for floating numbers. We also calculate the percentage difference for the FD method compared to the solution of the AD method in Table 7.6. In this case, 10^{-11} is the best perturbation size among these candidates, whose result is the most aligned with the solution of the AD method. Case III illustrates that although with finite difference it is possible to get a reliable transient DRC for a complicated system, it highly depends on the choice of the perturbation size. This issue does not hold for the automatic differentiation since there is no truncation error during the derivative evaluation process of the AD.

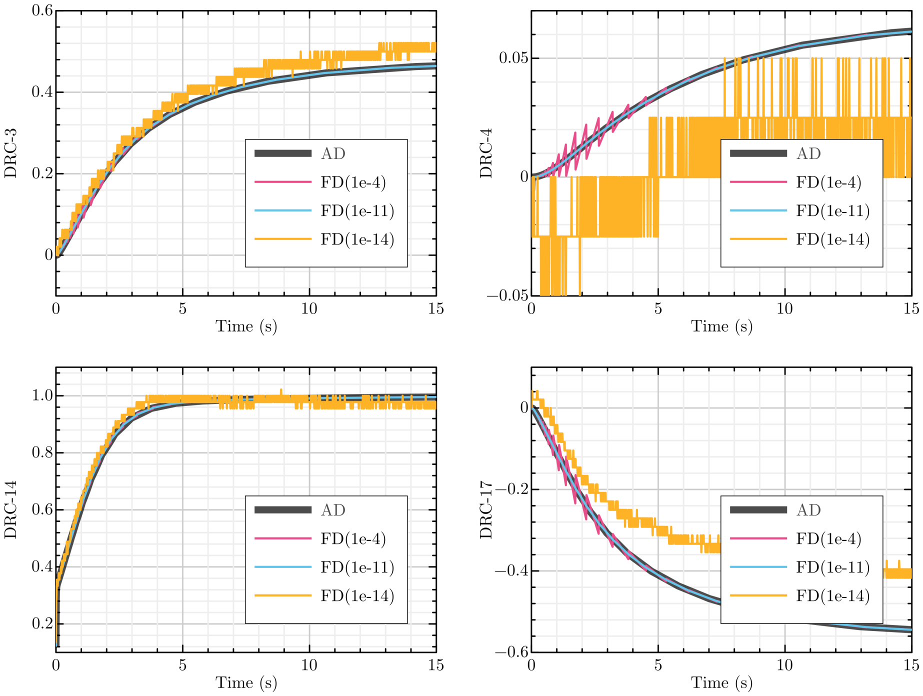

Figure 7.5: Transient degree of rate control evaluated by the finite difference and automatic differentiation for the propylene partial oxidation reactions 3, 4, 14 and 17 (case III).[173] Different perturbation sizes are used in the FD method as indicated in the legend.

Table 7.6: Average percentage difference betweent the solutions of the FD with different perturbations and the AD method over the time axis.

Step id	Perturb e^{-4}	Perturb e^{-11}	Perturb e^{-14}
3	1%	0%	11%
4	7%	1%	845%
14	0%	0%	3%
17	4%	0%	55%

7.3.4 Degree of Rate Control for Uncertain Parameters

It is common to consider an uncertain space of the kinetic parameters instead of the exact values in the real-world catalyst applications. To illustrate the usage of the DELSA[174] to measure the importance of the parameters over a space, we hypothetically add an uncertainty range to the forward kinetic constant of each elementary step of the case III, which corresponds to an uncertain range of [-0.03 eV, 0.03 eV] on the forward activation energy. We assume the log of the kinetic parameters are uniformly distributed in the candidate space. The samples (200 in our example) are drawn using the quasi-random Sobol sequence[188] which more evenly samples the space than a uniform distribution would. According to the definition of the DELSA importance measure (Eq 7.7), the importance of a kinetic parameter k_{ij} to the net rate of sample i could be measured by:

$$S_{k_{ij}} = \frac{\left|\frac{\partial \ln r_i}{\partial \ln k_{ij}}\right|^2_{k_{im\neq ij},K_j} s_j^2}{V(\ln r_i)} \tag{7.18}$$

where s_j is the variance of the uniform distribution of k_j. Among the 17 parameters, only 3 parameters are found to be important (with the average DELSA importance measure > 0.1) over the potential space. Their distributions are shown in Figure 7.6. For this reaction example, DELSA identifies the same important steps as the DRC results (step 3, 14, 17).

However, DELSA provides more details of the distribution of the importance across the parameter space. For example, the histogram of step 14 has two peaks near 0 and 1, which means that for about half of the samples, step 14 has no effect on the net rate (with 0 importance measure), but for the other half of the samples, step 14 is the most important parameter (with 1 importance measure). High-quality distributions rely on more samples in the potential space, which could be expensive for finite differences and difficult to ensure the results are accurate (e.g., rounding and truncating error). Using the automatic differentiation, DELSA could be performed with higher accuracy and lower computational costs.

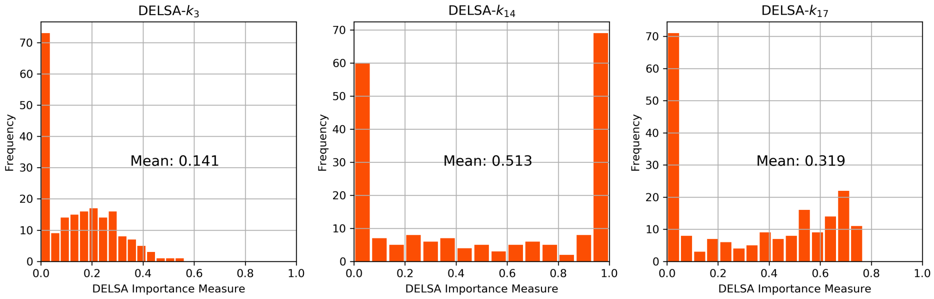

Figure 7.6: DELSA results for 3 important steps (3, 14 and 17) in the propylene oxidation reaction.[173]

7.4 Conclusion

The degree of rate control is a versatile concept in heterogeneous catalysis. We discussed the application of automatic differentiation (AD) in the calculation of the derivatives needed to evaluate the DRC. We also compared the AD method to the commonly used finite difference (FD) method. Compared to FD, AD provides a faster and more accurate solution. There is no need to choose the optimal perturbation size in AD which is a critical step in FD to obtain a reliable result. In three cases with various complexity, we demonstrated the correctness of the AD method. In addition, we illustrated that AD could be used to perform the DELSA method, which is a hybrid local-global sensitivity analysis method to measure the importance of parameters over an uncertain space. With AD, DELSA could be conducted accurately and efficiently.

8 Conclusions

This dissertation aims to present how machine learning and automatic differentiation can be used to accelerate the simulation and computation of important molecular and catalytic properties. Here, I would like to revisit the challenges that have been addressed in the previous chapters and make an outlook on the future directions along with these works to finalize this dissertation.

8.1 Machine Learning for Prediction, Exploration and Searching

A major advantage of the machine learning potential is the much faster inference of the potential energy surface than DFT with comparable accuracy. This feature enabled us to run large-scale MC simulations with an affordable time cost, which requires tens of thousands of inferences on the potential energies of large atomistic systems of more than 1000 atoms. We applied neural networks and the MC simulations to investigate the surface segregation behavior of a ternary CuPdAu alloy on a large slab. The large size of the slabs provided more details of the surface segregation by considering enriched local environments which were usually hard to be included via small unit cells with periodic conditions. With an extra machine learning model for the surface relaxation energies, the effect of the surface relaxation was integrated into our simulations, which made the simulation assumptions closer to the real experimental conditions.

Simulations under reaction conditions are more appealing to computational catalysis researchers. However, many more factors need to be considered during the simulations under reaction conditions. For example, the sampling space of the configurations with adsorbates on the surfaces, and the energetic model including the interactions between the adsorbates and the surfaces. We simulated the surface segregation and aggregation phenomenon on a AgPd slab with acrolein molecules. Some assumptions like sparse distribution of the adsorption sites and only one adsorbed species on the surface were made to limit the sampling space during simulations. For the interaction between the adsorbates and the surfaces, we used an additional ML model to predict the adsorption energies.

Obtaining the adsorption energies over different adsorption sites is not cheap. It requires a process to find the relaxed geometrical structures from initial guesses, which consists of iterative calculations of the forces for the structures along with the optimization path. We utilized neural networks as a surrogate model to accelerate the optimization process where an ensemble method was adopted to guide the exploration of the potential energy space from the initial structure to the relaxed structure. As a measure of the uncertainty of predictions, the neural network ensemble is useful to point out the information-rich samples in unknown space.

Parallel to the development of the machine learning potential, more and more molecular data is being generated every day. Searching for relevant information in the constantly growing databases could be a common event for various purposes. In Chapter 6, we demonstrated that the deep learning models could be leveraged to effectively retrieve molecular data from large databases. For example, searching for similar molecules or substructures to a query molecular structure. We presented the similarity search from different aspects, such as the geometrical features, energetic relevance, and even the density of states. A potential application of this method is to search for cheap materials with similar catalytic properties.

8.2 Automatic Differentiation for Fast and Accurate Derivatives

Automatic differentiation is one of the foundations of the recent development of deep learning models. In addition, many scientific and engineering areas could also see benefits from the development of the AD tools. We presented an example of using AD to evaluate the DRCs in complex reaction networks, which was usually performed using numerical methods like the finite difference. Unlike the finite difference, there is no need to choose the perturbation size using AD to evaluate the derivatives, which not only avoids the truncation error but also improves the speed of the whole evaluation process.

8.3 Potential Directions for Future Works

There are still lots of directions that need to be investigated. For the simulations under reaction conditions, we made several assumptions to reduce the sampling space. More effort is needed to enrich the modeling by considering more intermediate species on the surface, taking interactions among the adsorbates into account, and pushing the simulation concentrations beyond the dilute limits. In addition, our previous MC simulations were limited to systems with a few elements due to the scaling issue of the original ACSF descriptors.[18,19] This issue could be alleviated using recent GNN models and systems with more elements are able to be simulated.[21,23] Increasing scale of the models is one of the main trends to deal with more training data. Ensemble-based uncertainty estimation in the active learning framework becomes more expensive to models with many more parameters. Reliable and cheap uncertainty estimation methods or query strategies would be favorable to explore the chemical space actively and efficiently.

Furthermore, it would be exciting to see a working framework that combines the DRC evaluation, property prediction, and surface simulation to discover promising catalysts. DRC evaluation helps to find the rate-limiting steps of a reaction system. It also provides a direction to improve the target reaction rate like increasing or decreasing the activation energy of the rate-limiting steps. Then, a predicting model could be used to search for potential active sites with satisfying energy barriers. Finally, segregation and aggregation simulations are useful to find out the bulk compositions that are more likely to produce these desired active sites under the reaction conditions. Of course, each of these steps is not trivial. But with the development of machine learning models, uncertainty quantification methods, simulation algorithms, and computing resources, an automated and high-precision materials discovery system might become a reality.

References

[1] Lichen Liu and Avelino Corma. Metal catalysts for heterogeneous catalysis: From single atoms to nanoclusters and nanoparticles. *Chemical Reviews*, 118(10):4981–5079, 2018. doi: 10.1021/acs.chemrev.7b00776. URL https://doi.org/10.1021/acs.chemrev.7b00776.

[2] Ryan T. Hannagan, Georgios Giannakakis, Maria Flytzani-Stephanopoulos, and E. Charles H. Sykes. Single-atom alloy catalysis. *Chemical Reviews*, 120(21):acs.chemrev.0c00078, 2020. doi: 10.1021/acs.chemrev.0c00078. URL https://doi.org/10.1021/acs.chemrev.0c00078.

[3] Feilong Xing, Jaewan Jeon, Takashi Toyao, Ken ichi Shimizu, and Shinya Furukawa. A Cu-Pd single-atom alloy catalyst for highly efficient no reduction. *Chemical Science*, 10(36):8292–8298, 2019. doi: 10.1039/c9sc03172c. URL https://doi.org/10.1039/c9sc03172c.

[4] Xi Zhang, Guoqing Cui, Haisong Feng, Lifang Chen, Hui Wang, Bin Wang, Xin Zhang, Lirong Zheng, Song Hong, and Min Wei. Platinum-copper single atom alloy catalysts with high performance towards glycerol hydrogenolysis. *Nature Communications*, 10(1):5812, 2019. doi: 10.1038/s41467-019-13685-2. URL https://doi.org/10.1038/s41467-019-13685-2.

[5] Xing Wei, Xiao-Feng Yang, Ai-Qin Wang, Lin Li, Xiao-Yan Liu, Tao Zhang, Chung-Yuan Mou, and Jun Li. Bimetallic Au-Pd alloy catalysts for N_2O decomposition: Effects of surface structures on catalytic activity. *The Journal of Physical Chemistry C*, 116(10):6222–6232, 2012. doi: 10.1021/jp210555s. URL https://doi.org/10.1021/jp210555s.

[6] Wenhong Yang, Timothy Tizhe Fidelis, and Wen-Hua Sun. Machine learning in catalysis, from proposal to practicing. *ACS Omega*, 5(1): 83–88, 2019. doi: 10.1021/acsomega.9b03673. URL https://doi.org/10.1021/acsomega.9b03673.

[7] Lukáš Grajciar, Christopher J. Heard, Anton A. Bondarenko, Mikhail V. Polynski, Jittima Meeprasert, Evgeny A. Pidko, and Petr Nachtigall. Towards operando computational modeling in heterogeneous catalysis. *Chemical Society Reviews*, 47(22):8307–8348, 2018. doi: 10.1039/c8cs00398j. URL https://doi.org/10.1039/c8cs00398j.

[8] Jin-Woong Lee, Chaewon Park, Byung Do Lee, Joonseo Park, Nam Hoon Goo, and Kee-Sun Sohn. A machine-learning-based alloy design platform that enables both forward and inverse predictions for thermo-mechanically controlled processed (TMCP) steel alloys. *Scientific Reports*, 11(1):11012, 2021. doi: 10.1038/s41598-021-90237-z. URL https://doi.org/10.1038/s41598-021-90237-z.

[9] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Aini Palizhati, Anuroop Sriram, Brandon Wood, Junwoong Yoon, Devi Parikh, C. Lawrence Zitnick, and Zachary Ulissi. Open catalyst 2020 (OC20) dataset and community challenges. *ACS Catalysis*, 11(10):6059–6072, 2021. doi: 10.1021/acscatal.0c04525. URL https://doi.org/10.1021/acscatal.0c04525.

[10] Stefano Curtarolo, Wahyu Setyawan, Gus L.W. Hart, Michal Jahnatek, Roman V. Chepulskii, Richard H. Taylor, Shidong Wang, Junkai Xue, Kesong Yang, Ohad Levy, Michael J. Mehl, Harold T. Stokes, Denis O. Demchenko, and Dane Morgan. AFLOW: an automatic framework for high-throughput materials discovery. *Computational Materials Science*, 58(nil):218–226, 2012. doi: 10.1016/j.commatsci.2012.02.005. URL https://doi.org/10.1016/j.commatsci.2012.02.005.

[11] Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. Commentary: the materials project: a materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002, 2013. doi: 10.1063/1.4812323. URL https://doi.org/10.1063/1.4812323.

[12] K.W. Jacobsen, P. Stoltze, and J.K. Nørskov. A semi-empirical effective medium theory for metals and alloys. *Surface Science*, 366(2):394–402, 1996. doi: 10.1016/0039-6028(96)00816-3. URL https://doi.org/10.1016/0039-6028(96)00816-3.

[13] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997. ISBN 978-0-07-042807-2.

[14] Alex Sherstinsky. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404(nil):132306, 2020. doi: 10.1016/j.physd.2019.132306. URL https://doi.org/10.1016/j.physd.2019.132306.

[15] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9(4):611–629, 2018. doi: 10.1007/s13244-018-0639-9. URL https://doi.org/10.1007/s13244-018-0639-9.

[16] Errol G. Lewars. *The Concept of the Potential Energy Surface*, pages 9–49. Computational Chemistry. Springer International Publishing, 2016. doi: 10.1007/978-3-319-30916-3_2. URL https://doi.org/10.1007/978-3-319-30916-3_2.

[17] Thomas B. Blank, Steven D. Brown, August W. Calhoun, and Douglas J. Doren. Neural network models of potential energy surfaces. *The Journal of Chemical Physics*, 103(10):4129–4137, 1995. doi: 10.1063/1.469597. URL https://doi.org/10.1063/1.469597.

[18] Jörg Behler and Michele Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical Review Letters*, 98(14):146401, 2007. doi: 10.1103/physrevlett.98.146401. URL https://doi.org/10.1103/physrevlett.98.146401.

[19] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *The Journal of Chemical Physics*, 134(7):074106, 2011. doi: 10.1063/1.3553717. URL https://doi.org/10.1063/1.3553717.

[20] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. URL http://arxiv.org/abs/1609.04747.

[21] Tian Xie and Jeffrey C. Grossman. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Physical Review Letters*, 120(14):145301, 2018. doi: 10.1103/physrevlett.120.145301. URL https://doi.org/10.1103/physrevlett.120.145301.

[22] Cheol Woo Park, Mordechai Kornbluth, Jonathan Vandermause, Chris Wolverton, Boris Kozinsky, and Jonathan P. Mailoa. Accurate and scalable graph neural network force field and molecular dynamics with direct force architecture. *npj Computational Materials*, 7(1):73, 2021. doi: 10.1038/s41524-021-00543-3. URL https://doi.org/10.1038/s41524-021-00543-3.

[23] Johannes Klicpera, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. *CoRR*, 2021. URL http://arxiv.org/abs/2106.08903v8.

[24] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, 2016. URL http://arxiv.org/abs/1603.04467v2.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: an imperative style, high-performance deep learning library. *CoRR*, 2019. URL http://arxiv.org/abs/1912.01703v1.

[26] Yunqi Shao, Matti Hellström, Pavlin D. Mitev, Lisanne Knijff, and Chao Zhang. Pinn: a python library for building atomic neural networks of molecules and materials. *Journal of Chemical Information and Modeling*, 60(3):1184–1193, 2020. doi: 10.1021/acs.jcim.9b00994. URL https://doi.org/10.1021/acs.jcim.9b00994.

[27] K. T. Schütt, P. Kessel, M. Gastegger, K. A. Nicoli, A. Tkatchenko, and K.-R. Müller. Schnetpack: a deep learning toolbox for atomistic systems. *Journal of Chemical Theory and Computation*, 15(1):448–455, 2018. doi: 10.1021/acs.jctc.8b00908. URL https://doi.org/10.1021/acs.jctc.8b00908.

[28] Alireza Khorshidi and Andrew A. Peterson. Amp : a modular approach to machine learning in atomistic simulations. *Computer Physics Communications*, 207:310–324, 2016. doi: 10.1016/j.cpc.2016.05.010. URL https://doi.org/10.1016/j.cpc.2016.05.010.

[29] Jörg Behler. Constructing high-dimensional neural network potentials: a tutorial review. *International Journal of Quantum Chemistry*, 115(16): 1032–1050, 2015. doi: 10.1002/qua.24890. URL https://doi.org/10.1002/qua.24890.

[30] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3):nil, 2008. doi: 10.1214/009053607000000677. URL https://doi.org/10.1214/009053607000000677.

[31] Albert P. Bartók, Mike C. Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: the accuracy of quantum mechanics, without the electrons. *Physical Review Letters*, 104(13):136403, 2010. doi: 10.1103/physrevlett.104.136403. URL https://doi.org/10.1103/physrevlett.104.136403.

[32] Albert P. Bartók and Gábor Csányi. Gaussian approximation potentials: a brief tutorial introduction. *International Journal of Quantum Chemistry*, 115(16):1051–1057, 2015. doi: 10.1002/qua.24927. URL https://doi.org/10.1002/qua.24927.

[33] Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In *Advances in Neural Information Processing Systems 8*, pages 514–520. MIT press, 1996.

[34] Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of Chemical Physics*, 145(17):170901, 2016. doi: 10.1063/1.4966192. URL https://doi.org/10.1063/1.4966192.

[35] Albert P. Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B*, 87(18):184115, 2013. doi: 10.1103/physrevb.87.184115. URL https://doi.org/10.1103/physrevb.87.184115.

[36] Yilin Yang, Zhitao Guo, Andrew J. Gellman, and John R. Kitchin. Simulating segregation in a ternary Cu-Pd-Au alloy with density functional theory, machine learning, and Monte Carlo simulations. *The Journal of Physical Chemistry C*, 126(4):1800–1808, 2022. doi: 10.1021/acs.jpcc.1c09647. URL https://doi.org/10.1021/acs.jpcc.1c09647.

[37] J.C. Bertolini, J.L. Rousset, P. Miegge, J. Massardier, B. Tardy, Y. Samson, B.C. Khanra, and C. Creemers. Surface segregation study of a dilute Pd_1Fe_{99} alloy by LEIS and XPS. *Surface Science*, 281(1-2):102–110, 1993. doi: 10.1016/0039-6028(93)90859-i. URL https://doi.org/10.1016/0039-6028(93)90859-i.

[38] L. C. A. van den Oetelaar, O. W. Nooij, S. Oerlemans, A. W. Denier van der Gon, H. H. Brongersma, L. Lefferts, A. G. Roosenbrand, and J. A. R. van Veen. Surface segregation in supported Pd-Pt nanoclusters and alloys. *The Journal of Physical Chemistry B*, 102(18):3445–3455, 1998. doi: 10.1021/jp973395q. URL https://doi.org/10.1021/jp973395q.

[39] Deepika Priyadarshini, Petro Kondratyuk, Yoosuf N. Picard, Bryan D. Morreale, Andrew J. Gellman, and James B. Miller. High-throughput characterization of surface segregation in Cu_xPd_{1-x} alloys. *The Journal of Physical Chemistry C*, 115(20):10155–10163, 2011. doi: 10.1021/jp201793d. URL https://doi.org/10.1021/jp201793d.

[40] Ana M. Tarditi, Carolina Imhoff, Fernando Braun, James B. Miller, Andrew J. Gellman, and Laura Cornaglia. PdCuAu ternary alloy membranes: Hydrogen permeation properties in the presence of H_2S. *Journal of Membrane Science*, 479:246–255, 2015. doi: 10.1016/j.memsci.2014.12.030. URL https://doi.org/10.1016/j.memsci.2014.12.030.

[41] Matthew A. Payne, James B. Miller, Martin E. Oliveros, Geronimo Perez, Cristol P. Gouvea, Bráulio S. Archanjo, Carlos A. Achete, and

Andrew J. Gellman. Assessment of a high-throughput methodology for the study of alloy oxidation using $Al_xFe_yNi_{1-x-y}$ composition gradient thin films. *ACS Combinatorial Science*, 18(7):425–436, 2016. doi: 10.1021/acscombsci.6b00030. URL https://doi.org/10.1021/acscombsci.6b00030.

[42] John R. Kitchin and Andrew J. Gellman. High-throughput methods using composition and structure spread libraries. *AIChE Journal*, 62 (11):3826–3835, 2016. doi: 10.1002/aic.15294. URL https://doi.org/10.1002/aic.15294.

[43] Elena V. Levchenko, Alexander V. Evteev, Rafal Kozubski, Irina V. Belova, and Graeme E. Murch. Molecular dynamics simulation of surface segregation in a (110) B_2-NiAl thin film. *Phys. Chem. Chem. Phys.*, 13 (3):1214–1221, 2011. doi: 10.1039/c0cp00376j. URL https://doi.org/10.1039/c0cp00376j.

[44] Zhang Bangwei, Edmund Taglauer, Shu Xiaolin, Hu Wangyu, and Deng Huiqiu. Simulation calculations of surface segregation for Au-Cu alloys using an analytic embedded atom model. *physica status solidi (a)*, 202 (14):2686–2699, 2005. doi: 10.1002/pssa.200520092. URL https://doi.org/10.1002/pssa.200520092.

[45] Ho-Cheng Tsai, Ted H. Yu, Yao Sha, Boris V. Merinov, Pu-Wei Wu, San-Yuan Chen, and William A. Goddard. Density functional theory study of Pt_3M alloy surface segregation with adsorbed O/OH and Pt_3Os as catalysts for oxygen reduction reaction. *The Journal of Physical Chemistry C*, 118(46):26703–26712, 2014. doi: 10.1021/jp507103c. URL https://doi.org/10.1021/jp507103c.

[46] P. Wynblatt and A. Landa. Computer simulation of surface segregation in ternary alloys. *Computational Materials Science*, 15(2):250–263, 1999. doi: 10.1016/s0927-0256(99)00008-7. URL https://doi.org/10.1016/s0927-0256(99)00008-7.

[47] Susan Meñez Aspera, Ryan Lacdao Arevalo, Hiroshi Nakanishi, and Hideaki Kasai. First principles study of surface stability and segregation of PdRuRh ternary metal alloy system. *Surface Science*, 671:51–59, 2018. doi: 10.1016/j.susc.2018.02.005. URL https://doi.org/10.1016/j.susc.2018.02.005.

[48] Andrew A. Peterson, Rune Christensen, and Alireza Khorshidi. Addressing uncertainty in atomistic machine learning. *Physical Chemistry Chemical Physics*, 19(18):10978–10985, 2017. doi: 10.1039/c7cp00375g. URL https://doi.org/10.1039/c7cp00375g.

[49] Jacob R. Boes and John R. Kitchin. Modeling segregation on AuPd(111) surfaces with density functional theory and monte carlo simulations.

The Journal of Physical Chemistry C, 121(6):3479–3487, 2017. doi: 10.1021/acs.jpcc.6b12752. URL https://doi.org/10.1021/acs.jpcc.6b12752.

[50] Berk Onat, Ekin D. Cubuk, Brad D. Malone, and Efthimios Kaxiras. Implanted neural network potentials: Application to Li-Si alloys. *Physical Review B*, 97(9):094106, 2018. doi: 10.1103/physrevb.97.094106. URL https://doi.org/10.1103/physrevb.97.094106.

[51] A.M. Tarditi, M.L. Bosko, and L.M. Cornaglia. *Electroless Plating of Pd Binary and Ternary Alloys and Surface Characteristics for Application in Hydrogen Separation*, pages 1–24. Comprehensive Materials Finishing. Elsevier, 2017. doi: 10.1016/b978-0-12-803581-8.09166-9. URL https://doi.org/10.1016/b978-0-12-803581-8.09166-9.

[52] Nongnuch Artrith and Alexie M. Kolpak. Understanding the composition and activity of electrocatalytic nanoalloys in aqueous solvents: a combination of DFT and accurate neural network potentials. *Nano Letters*, 14(5):2670–2676, 2014. doi: 10.1021/nl5005674. URL https://doi.org/10.1021/nl5005674.

[53] G. Kresse and J. Hafner. Ab initio molecular dynamics for liquid metals. *Physical Review B*, 47(1):558–561, 1993. doi: 10.1103/physrevb.47.558. URL https://doi.org/10.1103/physrevb.47.558.

[54] G. Kresse and J. Furthmüller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Computational Materials Science*, 6(1):15–50, 1996. doi: 10.1016/0927-0256(96)00008-0. URL https://doi.org/10.1016/0927-0256(96)00008-0.

[55] P. E. Blöchl. Projector augmented-wave method. *Physical Review B*, 50(24):17953–17979, 1994. doi: 10.1103/physrevb.50.17953. URL https://doi.org/10.1103/physrevb.50.17953.

[56] G. Kresse and D. Joubert. From ultrasoft pseudopotentials to the projector augmented-wave method. *Physical Review B*, 59(3):1758–1775, 1999. doi: 10.1103/physrevb.59.1758. URL https://doi.org/10.1103/physrevb.59.1758.

[57] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. *Physical Review Letters*, 77(18):3865–3868, 1996. doi: 10.1103/physrevlett.77.3865. URL https://doi.org/10.1103/physrevlett.77.3865.

[58] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple [phys. rev. lett. 77, 3865 (1996)]. *Physical Review Letters*, 78(7):1396–1396, 1997. doi: 10.1103/

physrevlett.78.1396. URL https://doi.org/10.1103/physrevlett.78.1396.

[59] Hendrik J. Monkhorst and James D. Pack. Special points for brillouin-zone integrations. *Physical Review B*, 13(12):5188–5192, 1976. doi: 10.1103/physrevb.13.5188. URL https://doi.org/10.1103/physrevb.13.5188.

[60] Chunrong Yin, Zhitao Guo, and Andrew J. Gellman. Surface segregation across ternary alloy composition space: $Cu_{x}Au_{y}Pd_{1\text{-}x\text{-}y}$. *The Journal of Physical Chemistry C*, 124(19):10605–10614, 2020. doi: 10.1021/acs.jpcc.0c02058. URL https://doi.org/10.1021/acs.jpcc.0c02058.

[61] Nongnuch Artrith and Jörg Behler. High-dimensional neural network potentials for metal surfaces: a prototype study for copper. *Physical Review B*, 85(4):045439, 2012. doi: 10.1103/physrevb.85.045439. URL https://doi.org/10.1103/physrevb.85.045439.

[62] S.R. Bahn and K.W. Jacobsen. An object-oriented scripting interface to a legacy electronic structure code. *Computing in Science & Engineering*, 4(3):56–66, 2002. doi: 10.1109/5992.998641. URL https://doi.org/10.1109/5992.998641.

[63] Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer, Cory Hargus, Eric D Hermes, Paul C Jennings, Peter Bjerre Jensen, James Kermode, John R Kitchin, Esben Leonhard Kolsbjerg, Joseph Kubal, Kristen Kaasbjerg, Steen Lysgaard, Jón Bergmann Maronsson, Tristan Maxson, Thomas Olsen, Lars Pastewka, Andrew Peterson, Carsten Rostgaard, Jakob Schiøtz, Ole Schütt, Mikkel Strange, Kristian S Thygesen, Tejs Vegge, Lasse Vilhelmsen, Michael Walter, Zhenhua Zeng, and Karsten W Jacobsen. The atomic simulation environment-a python library for working with atoms. *Journal of Physics: Condensed Matter*, 29(27):273002, 2017. doi: 10.1088/1361-648x/aa680e. URL https://doi.org/10.1088/1361-648x/aa680e.

[64] L. Vegard. Die konstitution der mischkristalle und die raumfllung der atome. *Zeitschrift fr Physik*, 5(1):17–26, 1921. doi: 10.1007/bf01349680. URL https://doi.org/10.1007/bf01349680.

[65] Pernilla Tanner. Ordering and segregation in the PdAuCu system: Bulk vs. surface. Master's thesis, Chalmers University of Technology, 2019.

[66] A. R. Denton and N. W. Ashcroft. Vegard's law. *Physical Review A*, 43(6):3161–3164, 1991. doi: 10.1103/physreva.43.3161. URL https://doi.org/10.1103/physreva.43.3161.

[67] Feiyang Geng, Jacob R. Boes, and John R. Kitchin. First-principles study of the Cu-Pd phase diagram. *Calphad*, 56:224–229, 2017. doi: 10.1016/j.calphad.2017.01.009. URL https://doi.org/10.1016/j.calphad.2017.01.009.

[68] Lukas Pielsticker, Ioannis Zegkinoglou, Zhong-Kang Han, Juan J. Navarro, Sebastian Kunze, Osman Karslıoğlu, Sergey V. Levchenko, and Beatriz Roldan Cuenya. Crystallographic orientation dependence of surface segregation and alloying on PdCu catalysts for CO_2 hydrogenation. *The Journal of Physical Chemistry Letters*, 12(10):2570–2575, 2021. doi: 10.1021/acs.jpclett.1c00179. URL https://doi.org/10.1021/acs.jpclett.1c00179.

[69] S.V. Meschel and O.J. Kleppa. Thermochemistry of some binary alloys of noble metals (Cu, Ag, Au) and transition metals by high temperature direct synthesis calorimetry. *Journal of Alloys and Compounds*, 350(1-2):205–212, 2003. doi: 10.1016/s0925-8388(02)00983-0. URL https://doi.org/10.1016/s0925-8388(02)00983-0.

[70] Marcel H. F. Sluiter, C. Colinet, and Alain Pasturel. Ab initio calculation of the phase stability in Au-Pd and Ag-Pt alloys. *Physical Review B*, 73(17):174204, 2006. doi: 10.1103/physrevb.73.174204. URL https://doi.org/10.1103/physrevb.73.174204.

[71] Mei Li, Zhenmin Du, Cuiping Guo, and Changrong Li. A thermodynamic modeling of the Cu-Pd system. *Calphad*, 32(2):439–446, 2008. doi: 10.1016/j.calphad.2008.04.004. URL https://doi.org/10.1016/j.calphad.2008.04.004.

[72] Yongsheng Zhang, Georg Kresse, and C. Wolverton. Nonlocal first-principles calculations in Cu-Au and other intermetallic alloys. *Physical Review Letters*, 112(7):075502, 2014. doi: 10.1103/physrevlett.112.075502. URL https://doi.org/10.1103/physrevlett.112.075502.

[73] Yilin Yang, Omar A. Jiménez-Negrón, and John R. Kitchin. Machine-learning accelerated geometry optimization in molecular simulation. *The Journal of Chemical Physics*, 154(23):234704, 2021. doi: 10.1063/5.0049665. URL https://doi.org/10.1063/5.0049665.

[74] Zheng Li, Siwen Wang, Wei Shan Chin, Luke E. Achenie, and Hongliang Xin. High-throughput screening of bimetallic catalysts enabled by machine learning. *Journal of Materials Chemistry A*, 5(46):24131–24138, 2017. doi: 10.1039/c7ta01812f. URL https://doi.org/10.1039/c7ta01812f.

[75] Chongyi Ling, Yixin Ouyang, Qiang Li, Xiaowan Bai, Xin Mao, Aijun Du, and Jinlan Wang. A general two-step strategy-based high-throughput screening of single atom catalysts for nitrogen fixation.

Small Methods, 3(9):1800376, 2018. doi: 10.1002/smtd.201800376. URL https://doi.org/10.1002/smtd.201800376.

[76] Jiawei Zhang, Peijun Hu, and Haifeng Wang. Amorphous catalysis: Machine learning driven high-throughput screening of superior active site for hydrogen evolution reaction. *The Journal of Physical Chemistry C*, 124(19):10483–10494, 2020. doi: 10.1021/acs.jpcc.0c00406. URL https://doi.org/10.1021/acs.jpcc.0c00406.

[77] Jun Gu, Ya-Wen Zhang, and Franklin (Feng) Tao. Shape control of bimetallic nanocatalysts through well-designed colloidal chemistry approaches. *Chemical Society Reviews*, 41(24):8050, 2012. doi: 10.1039/c2cs35184f. URL https://doi.org/10.1039/c2cs35184f.

[78] Zachary W. Ulissi, Michael T. Tang, Jianping Xiao, Xinyan Liu, Daniel A. Torelli, Mohammadreza Karamad, Kyle Cummins, Christopher Hahn, Nathan S. Lewis, Thomas F. Jaramillo, Karen Chan, and Jens K. Nørskov. Machine-learning methods enable exhaustive searches for active bimetallic facets and reveal active site motifs for CO_2 reduction. *ACS Catalysis*, 7(10):6600–6608, 2017. doi: 10.1021/acscatal.7b01648. URL https://doi.org/10.1021/acscatal.7b01648.

[79] Robert A. Hoyt, Matthew M. Montemore, Ioanna Fampiou, Wei Chen, Georgios Tritsaris, and Efthimios Kaxiras. Machine learning prediction of H adsorption energies on Ag alloys. *Journal of Chemical Information and Modeling*, 59(4):1357–1365, 2019. doi: 10.1021/acs.jcim.8b00657. URL https://doi.org/10.1021/acs.jcim.8b00657.

[80] Andrew A. Peterson. Acceleration of saddle-point searches with machine learning. *The Journal of Chemical Physics*, 145(7):074106, 2016. doi: 10.1063/1.4960708. URL https://doi.org/10.1063/1.4960708.

[81] José A. Garrido Torres, Paul C. Jennings, Martin H. Hansen, Jacob R. Boes, and Thomas Bligaard. Low-scaling algorithm for nudged elastic band calculations using a surrogate machine learning model. *Physical Review Letters*, 122(15):156001, 2019. doi: 10.1103/physrevlett.122.156001. URL https://doi.org/10.1103/physrevlett.122.156001.

[82] Olli-Pekka Koistinen, Freyja B. Dagbjartsdóttir, Vilhjálmur Ásgeirsson, Aki Vehtari, and Hannes Jónsson. Nudged elastic band calculations accelerated with gaussian process regression. *The Journal of Chemical Physics*, 147(15):152720, 2017. doi: 10.1063/1.4986787. URL https://doi.org/10.1063/1.4986787.

[83] Estefanía Garijo del Río, Jens Jørgen Mortensen, and Karsten Wedel Jacobsen. Local Bayesian optimizer for atomic structures. *Physical Review B*, 100(10):104103, 2019. doi: 10.1103/physrevb.100.104103. URL https://doi.org/10.1103/physrevb.100.104103.

[84] Estefanía Garijo del Río, Sami Kaappa, José A. Garrido Torres, Thomas Bligaard, and Karsten Wedel Jacobsen. Machine learning with bond information for local structure optimizations in surface science. *The Journal of Chemical Physics*, 153(23):234116, 2020. doi: 10.1063/5.0033778. URL https://doi.org/10.1063/5.0033778.

[85] Jonathan Vandermause, Steven B. Torrisi, Simon Batzner, Yu Xie, Lixin Sun, Alexie M. Kolpak, and Boris Kozinsky. On-the-fly active learning of interpretable bayesian force fields for atomistic rare events. *npj Computational Materials*, 6(1):20, 2020. doi: 10.1038/s41524-020-0283-z. URL https://doi.org/10.1038/s41524-020-0283-z.

[86] Muhammed Shuaibi, Saurabh Sivakumar, Rui Qi Chen, and Zachary W Ulissi. Enabling robust offline active learning for machine learning potentials using simple physics-based priors. *Machine Learning: Science and Technology*, 2(2):025007, 2021. doi: 10.1088/2632-2153/abcc44. URL https://doi.org/10.1088/2632-2153/abcc44.

[87] Qunchao Tong, Lantian Xue, Jian Lv, Yanchao Wang, and Yanming Ma. Accelerating CALYPSO structure prediction by data-driven learning of a potential energy surface. *Faraday Discussions*, 211:31–43, 2018. doi: 10.1039/c8fd00055g. URL https://doi.org/10.1039/c8fd00055g.

[88] Ryosuke Jinnouchi, Kazutoshi Miwa, Ferenc Karsai, Georg Kresse, and Ryoji Asahi. On-the-fly active learning of interatomic potentials for large-scale atomistic simulations. *The Journal of Physical Chemistry Letters*, 11(17):6946–6955, 2020. doi: 10.1021/acs.jpclett.0c01061. URL https://doi.org/10.1021/acs.jpclett.0c01061.

[89] Mingjie Liu and John R. Kitchin. Singlenn: Modified behler-parrinello neural network with shared weights for atomistic simulations with transferability. *The Journal of Physical Chemistry C*, 124(32):17811–17818, 2020. doi: 10.1021/acs.jpcc.0c04225. URL https://doi.org/10.1021/acs.jpcc.0c04225.

[90] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/62dad6e273d32235ae02b7d321578ee8-Paper.pdf.

[91] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *CoRR*, 2016. URL http://arxiv.org/abs/1612.01474v3.

[92] Matthew Olson, Abraham Wyner, and Richard Berk. Modern neural networks generalize on small data sets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/fface8385abbf94b4593a0ed53a0c70f-Paper.pdf.

[93] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989. doi: 10.1007/bf01589116. URL https://doi.org/10.1007/bf01589116.

[94] On the determination of molecular fields.-I. from the variation of the viscosity of a gas with temperature. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 106(738):441–462, 1924. doi: 10.1098/rspa.1924.0081. URL https://doi.org/10.1098/rspa.1924.0081.

[95] T. L. Jacobsen, M. S. Jørgensen, and B. Hammer. On-the-fly machine learning of atomic potential in density functional theory structure optimization. *Physical Review Letters*, 120(2):026102, 2018. doi: 10.1103/physrevlett.120.026102. URL https://doi.org/10.1103/physrevlett.120.026102.

[96] Christoph Schran, Fabien Brieuc, and Dominik Marx. Transferability of machine learning potentials: Protonated water neural network potential applied to the protonated water hexamer. *The Journal of Chemical Physics*, 154(5):051101, 2021. doi: 10.1063/5.0035438. URL https://doi.org/10.1063/5.0035438.

[97] Graeme Henkelman, Blas P. Uberuaga, and Hannes Jónsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of Chemical Physics*, 113(22):9901–9904, 2000. doi: 10.1063/1.1329672. URL https://doi.org/10.1063/1.1329672.

[98] Yilin Yang. NN ensemble relaxer. https://github.com/yilinyang1/NN-ensemble-relaxer.

[99] Kyuhyun Lee, Dongsun Yoo, Wonseok Jeong, and Seungwu Han. Simple-nn: an efficient package for training and executing neural-network interatomic potentials. *Computer Physics Communications*, 242:95–103, 2019. doi: 10.1016/j.cpc.2019.04.014. URL https://doi.org/10.1016/j.cpc.2019.04.014.

[100] Péter Pulay. Convergence acceleration of iterative sequences. the case of scf iteration. *Chemical Physics Letters*, 73(2):393–398, 1980.

doi: 10.1016/0009-2614(80)80396-4. URL https://doi.org/10.1016/0009-2614(80)80396-4.

[101] Mingjie Liu, Yilin Yang, and John R. Kitchin. Semi-grand canonical Monte Carlo simulation of the acrolein induced surface segregation and aggregation of AgPd with machine learning surrogate models. *The Journal of Chemical Physics*, 154(13):134701, 2021. doi: 10.1063/5.0046440. URL https://doi.org/10.1063/5.0046440.

[102] Payoli Aich, Haojuan Wei, Bridget Basan, A. Jeremy Kropf, Neil M. Schweitzer, Christopher L. Marshall, Jeffrey T. Miller, and Randall Meyer. Single-atom alloy Pd-Ag catalyst for selective hydrogenation of acrolein. *The Journal of Physical Chemistry C*, 119(32):18140–18148, 2015. doi: 10.1021/acs.jpcc.5b01357. URL https://doi.org/10.1021/acs.jpcc.5b01357.

[103] Kaili Wang and Bo Yang. Theoretical understanding on the selectivity of acrolein hydrogenation over silver surfaces: the non-Horiuti-Polanyi mechanism is the key. *Catalysis Science & Technology*, 7(18):4024–4033, 2017. doi: 10.1039/c7cy01500c. URL https://doi.org/10.1039/c7cy01500c.

[104] Karl-Heinz Dostert, Casey P. O'Brien, Francesca Mirabella, Francisco Ivars-Barceló, Smadar Attia, Evan Spadafora, Swetlana Schauermann, and Hans-Joachim Freund. Selective partial hydrogenation of acrolein on Pd: A mechanistic study. *ACS Catalysis*, 7(8):5523–5533, 2017. doi: 10.1021/acscatal.7b01875. URL https://doi.org/10.1021/acscatal.7b01875.

[105] Luis E. Murillo, Amit M. Goda, and Jingguang G. Chen. Selective hydrogenation of the CO bond in acrolein through the architecture of bimetallic surface structures. *Journal of the American Chemical Society*, 129(22):7101–7105, 2007. doi: 10.1021/ja070264m. URL https://doi.org/10.1021/ja070264m.

[106] P. Gallezot and D. Richard. Selective hydrogenation of α,β-unsaturated aldehydes. *Catalysis Reviews*, 40(1-2):81–126, 1998. doi: 10.1080/01614949808007106. URL https://doi.org/10.1080/01614949808007106.

[107] Peter Claus, Herbert Hofmeister, and Christian Mohr. Identification of active sites and influence of real structure of gold catalysts in the selective hydrogenation of acrolein to allyl alcohol. *Gold Bulletin*, 37 (3-4):181–186, 2004. doi: 10.1007/bf03215211. URL https://doi.org/10.1007/bf03215211.

[108] Chuan-Ming Wang, Kang-Nian Fan, and Zhi-Pan Liu. Oxide-supported single gold catalyst for selective hydrogenation of acrolein predicted from

first principles. *Journal of Catalysis*, 266(2):343–350, 2009. doi: 10.1016/j.jcat.2009.06.023. URL https://doi.org/10.1016/j.jcat.2009.06.023.

[109] Bernard Coq, Franois Figuras, Claude Moreau, Patrice Moreau, and Mayukh Warawdekar. Hydrogenation of substituted acrolein over alumina supported ruthenium catalysts. *Catalysis Letters*, 22(3):189–195, 1993. doi: 10.1007/bf00810365. URL https://doi.org/10.1007/bf00810365.

[110] Mark Muir, David L. Molina, Arephin Islam, Mohammed K. Abdel-Rahman, and Michael Trenary. Selective hydrogenation of acrolein on a Pd/Ag(111) single-atom alloy surface. *The Journal of Physical Chemistry C*, 124(44):24271–24278, 2020. doi: 10.1021/acs.jpcc.0c08094. URL https://doi.org/10.1021/acs.jpcc.0c08094.

[111] D. Loffreda, Y. Jugnet, F. Delbecq, J. C. Bertolini, and P. Sautet. Coverage dependent adsorption of acrolein on Pt(111) from a combination of first principle theory and hreels study. *The Journal of Physical Chemistry B*, 108(26):9085–9093, 2004. doi: 10.1021/jp037639k. URL https://doi.org/10.1021/jp037639k.

[112] S. Pirillo, I. López-Corral, E. Germán, and A. Juan. The adsorption of acrolein on a Pt (111): a study of chemical bonding and electronic structure. *Applied Surface Science*, 263(nil):79–85, 2012. doi: 10.1016/j.apsusc.2012.08.125. URL https://doi.org/10.1016/j.apsusc.2012.08.125.

[113] Karl-Heinz Dostert, Casey P. O'Brien, Francesca Mirabella, Francisco Ivars-Barceló, and Swetlana Schauermann. Adsorption of acrolein, propanal, and allyl alcohol on Pd(111): a combined infrared reflection-absorption spectroscopy and temperature programmed desorption study. *Physical Chemistry Chemical Physics*, 18(20):13960–13973, 2016. doi: 10.1039/c6cp00877a. URL https://doi.org/10.1039/c6cp00877a.

[114] Mark Muir, David L. Molina, Arephin Islam, Mohammed K. Abdel-Rahman, and Michael Trenary. Adsorption properties of acrolein, propanal, 2-propenol, and 1-propanol on Ag(111). *Physical Chemistry Chemical Physics*, 22(43):25011–25020, 2020. doi: 10.1039/d0cp04634e. URL https://doi.org/10.1039/d0cp04634e.

[115] G. Kresse and J. Hafner. Ab initio molecular-dynamics simulation of the liquid-metal-amorphous-semiconductor transition in germanium. *Physical Review B*, 49(20):14251–14269, 1994. doi: 10.1103/physrevb.49.14251. URL https://doi.org/10.1103/physrevb.49.14251.

[116] G. Kresse and J. Furthmüller. Efficient iterative schemes forab initiototal-energy calculations using a plane-wave basis set. *Physical*

Review B, 54(16):11169–11186, 1996. doi: 10.1103/physrevb.54.11169. URL https://doi.org/10.1103/physrevb.54.11169.

[117] Geun Ho Gu, Juhwan Noh, Sungwon Kim, Seoin Back, Zachary Ulissi, and Yousung Jung. Practical deep-learning representation for fast heterogeneous catalyst screening. *The Journal of Physical Chemistry Letters*, 11(9):3185–3191, 2020. doi: 10.1021/acs.jpclett.0c00634. URL https://doi.org/10.1021/acs.jpclett.0c00634.

[118] Evelyn Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, 57(3):238, 1989. doi: 10.2307/1403797. URL https://doi.org/10.2307/1403797.

[119] O.M. Løvvik and Susanne M. Opalka. Reversed surface segregation in palladium-silver alloys due to hydrogen adsorption. *Surface Science*, 602(17):2840–2844, 2008. doi: 10.1016/j.susc.2008.07.016. URL https://doi.org/10.1016/j.susc.2008.07.016.

[120] O.M. Løvvik. Surface segregation in palladium based alloys from density-functional calculations. *Surface Science*, 583(1):100–106, 2005. doi: 10.1016/j.susc.2005.03.028. URL https://doi.org/10.1016/j.susc.2005.03.028.

[121] M. Ropo, K. Kokko, L. Vitos, and J. Kollár. Segregation at the PdAg(111) surface:electronic structure calculations. *Physical Review B*, 71(4):045411, 2005. doi: 10.1103/physrevb.71.045411. URL https://doi.org/10.1103/physrevb.71.045411.

[122] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. Pubchem in 2021: New data content and improved web interfaces. *Nucleic Acids Research*, 49(D1):D1388–D1395, 2020. doi: 10.1093/nar/gkaa971. URL https://doi.org/10.1093/nar/gkaa971.

[123] Dagmar Stumpfe and Jürgen Bajorath. Similarity searching. *WIREs Computational Molecular Science*, 1(2):260–282, 2011. doi: 10.1002/wcms.23. URL https://doi.org/10.1002/wcms.23.

[124] Adrià Cereto-Massagué, María José Ojeda, Cristina Valls, Miquel Mulero, Santiago Garcia-Vallvé, and Gerard Pujadas. Molecular fingerprint similarity search in virtual screening. *Methods*, 71(nil):58–63, 2015. doi: 10.1016/j.ymeth.2014.08.005. URL https://doi.org/10.1016/j.ymeth.2014.08.005.

[125] Kirstin Moffat, Valerie J. Gillet, Martin Whittle, Gianpaolo Bravi, and Andrew R. Leach. A comparison of field-based similarity

searching methods: CatShape, FBSS, and ROCS. *Journal of Chemical Information and Modeling*, 48(4):719–729, 2008. doi: 10.1021/ci700130j. URL https://doi.org/10.1021/ci700130j.

[126] Oliver Laufkötter, Tomoyuki Miyao, and Jürgen Bajorath. Large-scale comparison of alternative similarity search strategies with varying chemical information contents. *ACS Omega*, 4(12):15304–15311, 2019. doi: 10.1021/acsomega.9b02470. URL https://doi.org/10.1021/acsomega.9b02470.

[127] Michael A. Skinnider, Chris A. Dejong, Brian C. Franczak, Paul D. McNicholas, and Nathan A. Magarvey. Comparative analysis of chemical similarity methods for modular natural products with a hypothetical structure enumeration algorithm. *Journal of Cheminformatics*, 9(1):46, 2017. doi: 10.1186/s13321-017-0234-y. URL https://doi.org/10.1186/s13321-017-0234-y.

[128] Julian Schwartz, Mahendra Awale, and Jean-Louis Reymond. SMIfp (SMILES fingerprint) chemical space for virtual screening and visualization of large databases of organic molecules. *Journal of Chemical Information and Modeling*, 53(8):1979–1989, 2013. doi: 10.1021/ci400206h. URL https://doi.org/10.1021/ci400206h.

[129] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

[130] David Weininger, Arthur Weininger, and Joseph L. Weininger. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of Chemical Information and Computer Sciences*, 29(2):97–101, 1989. doi: 10.1021/ci00062a008. URL https://doi.org/10.1021/ci00062a008.

[131] Joseph L. Durant, Burton A. Leland, Douglas R. Henry, and James G. Nourse. Reoptimization of MDL keys for use in drug discovery. *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280, 2002. doi: 10.1021/ci010132r. URL https://doi.org/10.1021/ci010132r.

[132] John M. Barnard and Geoff M. Downs. Chemical fragment generation and clustering software. *Journal of Chemical Information and Computer Sciences*, 37(1):141–142, 1997. doi: 10.1021/ci960090k. URL https://doi.org/10.1021/ci960090k.

[133] Kunihiko Fukushima. Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980. doi: 10.1007/bf00344251. URL https://doi.org/10.1007/bf00344251.

[134] Kunihiko Fukushima. Artificial vision by multi-layered neural networks: Neocognitron and its advances. *Neural Networks*, 37(nil):103–119, 2013. doi: 10.1016/j.neunet.2012.09.016. URL https://doi.org/10.1016/j.neunet.2012.09.016.

[135] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[136] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707, 2016. doi: 10.1109/taslp.2016.2520371. URL https://doi.org/10.1109/taslp.2016.2520371.

[137] Aun Irtaza, M. Arfan Jaffar, Eisa Aleisa, and Tae-Sun Choi. Embedding neural networks for semantic association in content based image retrieval. *Multimedia Tools and Applications*, 72(2):1911–1931, 2013. doi: 10.1007/s11042-013-1489-6. URL https://doi.org/10.1007/s11042-013-1489-6.

[138] Artem Babenko Yandex and Victor Lempitsky. Aggregating local deep features for image retrieval. In *2015 IEEE International Conference on Computer Vision (ICCV)*, page nil, 12 2015. doi: 10.1109/iccv.2015.150. URL https://doi.org/10.1109/iccv.2015.150.

[139] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. *CoRR*, 2018. URL http://arxiv.org/abs/1806.09823v1.

[140] Mengzhao Wang, Xiaoliang Xu, Qiang Yue, and Yuxiang Wang. A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. *Proceedings of the VLDB Endowment*, 14(11):1964–1978, 2021. doi: 10.14778/3476249.3476255. URL https://doi.org/10.14778/3476249.3476255.

[141] Qiang Huang, Jianlin Feng, Yikai Zhang, Qiong Fang, and Wilfred Ng. Query-aware locality-sensitive hashing for approximate nearest neighbor search. *Proceedings of the VLDB Endowment*, 9(1):1–12, 2015.

[142] Qi Chen, Haidong Wang, Mingqin Li, Gang Ren, Scarlett Li, Jeffery Zhu, Jason Li, Chuanjie Liu, Lintao Zhang, and Jingdong Wang. *SPTAG: A library for fast approximate nearest neighbor search*, 2018. URL https://github.com/Microsoft/SPTAG.

[143] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. *CoRR*, 2019. URL http://arxiv.org/abs/1908.10396v5.

[144] Jeff Johnson, Matthijs Douze, and Herve Jegou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021. doi: 10.1109/tbdata.2019.2921572. URL https://doi.org/10.1109/tbdata.2019.2921572.

[145] Sivic and Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, page nil, - 2003. doi: 10.1109/iccv.2003.1238663. URL https://doi.org/10.1109/iccv.2003.1238663.

[146] H Jégou, M Douze, and C Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/tpami.2010.57. URL https://doi.org/10.1109/tpami.2010.57.

[147] Lars Ruddigkeit, Ruud van Deursen, Lorenz C. Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *Journal of Chemical Information and Modeling*, 52(11):2864–2875, 2012. doi: 10.1021/ci300415d. URL https://doi.org/10.1021/ci300415d.

[148] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):140022, 2014. doi: 10.1038/sdata.2014.22. URL https://doi.org/10.1038/sdata.2014.22.

[149] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t. URL https://doi.org/10.1021/ci100050t.

[150] H. L. Morgan. The generation of a unique machine description for chemical structures-a technique developed at Chemical Abstracts Service. *Journal of Chemical Documentation*, 5(2):107–113, 1965. doi: 10.1021/c160017a018. URL https://doi.org/10.1021/c160017a018.

[151] Dávid Bajusz, Anita Rácz, and Károly Héberger. Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of Cheminformatics*, 7(1):20, 2015. doi: 10.1186/s13321-015-0069-3. URL https://doi.org/10.1186/s13321-015-0069-3.

[152] Yilin Yang, Siddarth K. Achar, and John R. Kitchin. Evaluation of the degree of rate control via automatic differentiation. *AIChE Journal*,

nil(nil):nil, 2022. doi: 10.1002/aic.17653. URL https://doi.org/10.1002/aic.17653.

[153] Carsten Stegelmann, Anders Andreasen, and Charles T. Campbell. Degree of rate control: How much the energies of intermediates and transition states control rates. *Journal of the American Chemical Society*, 131(23):8077–8082, 2009. doi: 10.1021/ja9000097. URL https://doi.org/10.1021/ja9000097.

[154] Charles T. Campbell. The degree of rate control: A powerful tool for catalysis research. *ACS Catalysis*, 7(4):2770–2779, 2017. doi: 10.1021/acscatal.7b00115. URL https://doi.org/10.1021/acscatal.7b00115.

[155] C. Stegelmann, N.C. Schiødt, Charles T. Campbell, and P. Stoltze. Microkinetic modeling of ethylene oxidation over silver. *Journal of Catalysis*, 221(2):630–649, 2004. doi: 10.1016/j.jcat.2003.10.004. URL https://doi.org/10.1016/j.jcat.2003.10.004.

[156] Lars C. Grabow, Amit A. Gokhale, Steven T. Evans, James A. Dumesic, and Manos Mavrikakis. Mechanism of the water gas shift reaction on Pt: First principles, experiments, and microkinetic modeling. *The Journal of Physical Chemistry C*, 112(12):4608–4617, 2008. doi: 10.1021/jp7099702. URL https://doi.org/10.1021/jp7099702.

[157] Christopher A. Wolcott, Andrew J. Medford, Felix Studt, and Charles T. Campbell. Degree of rate control approach to computational catalyst screening. *Journal of Catalysis*, 330:197–207, 2015. doi: 10.1016/j.jcat.2015.07.015. URL https://doi.org/10.1016/j.jcat.2015.07.015.

[158] Mikkel Jørgensen and Henrik Grönbeck. Connection between macroscopic kinetic measurables and the degree of rate control. *Catalysis Science & Technology*, 7(18):4034–4040, 2017. doi: 10.1039/c7cy01246b. URL https://doi.org/10.1039/c7cy01246b.

[159] Zhongtian Mao and Charles T. Campbell. Apparent activation energies in complex reaction mechanisms: A simple relationship via degrees of rate control. *ACS Catalysis*, 9(10):9465–9473, 2019. doi: 10.1021/acscatal.9b02761. URL https://doi.org/10.1021/acscatal.9b02761.

[160] Talin Avanesian, Gabriel S. Gusmão, and Phillip Christopher. Mechanism of CO_2 reduction by H_2 on Ru(0001) and general selectivity descriptors for late-transition metal catalysts. *Journal of Catalysis*, 343:86–96, 2016. doi: 10.1016/j.jcat.2016.03.016. URL https://doi.org/10.1016/j.jcat.2016.03.016.

[161] Charles T Campbell. Finding the rate-determining step in a mechanism. *Journal of Catalysis*, 204(2):520–524, 2001. doi: 10.1006/jcat.2001.3396. URL https://doi.org/10.1006/jcat.2001.3396.

[162] Brandon L. Foley and Aditya Bhan. Degrees of rate control at non(pseudo)steady-state conditions. *ACS Catalysis*, 10(4):2556–2564, 2020. doi: 10.1021/acscatal.9b04910. URL https://doi.org/10.1021/acscatal.9b04910.

[163] Huijie Tian and Srinivas Rangarajan. Computing a global degree of rate control for catalytic systems. *ACS Catalysis*, 10(22):13535–13542, 2020. doi: 10.1021/acscatal.0c03150. URL https://doi.org/10.1021/acscatal.0c03150.

[164] Bjarne Kreitz, C. Franklin Goldsmith, Richard West, Emily Mazeau, Katrin Blondal, Gregor Wehinger, Thomas Turek, and Khachik Sargsyan. Quantifying the impact of parametric uncertainty on automatic mechanism generation for CO_2 hydrogenation on Ni(111), Apr 2021. URL https://chemrxiv.org/articles/preprint/Quantifying_the_Impact_of_Parametric_Uncertainty_on_Automatic_Mechanism_Generation_for_CO2_Hydrogenation_on_Ni_111_/14376899/1.

[165] Yang Cao, Shengtai Li, Linda Petzold, and Radu Serban. Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution. *SIAM Journal on Scientific Computing*, 24(3):1076–1089, 2003. doi: 10.1137/s1064827501380630. URL https://doi.org/10.1137/s1064827501380630.

[166] Alan C. Hindmarsh, Peter N. Brown, Keith E. Grant, Steven L. Lee, Radu Serban, Dan E. Shumaker, and Carol S. Woodward. Sundials. *ACM Transactions on Mathematical Software*, 31(3):363–396, 2005. doi: 10.1145/1089014.1089020. URL https://doi.org/10.1145/1089014.1089020.

[167] Hong Zhang and Adrian Sandu. FATODE: a library for forward, adjoint, and tangent linear integration of ODES. *SIAM Journal on Scientific Computing*, 36(5):C504–C523, 2014. doi: 10.1137/130912335. URL https://doi.org/10.1137/130912335.

[168] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *CoRR*, abs/1806.07366, 2018. URL http://arxiv.org/abs/1806.07366.

[169] Adrian Sandu, Dacian N. Daescu, and Gregory R. Carmichael. Direct and adjoint sensitivity analysis of chemical kinetic systems with KPP: Part I-theory and software tools. *Atmospheric Environment*, 37(36):

5083–5096, 2003. doi: 10.1016/j.atmosenv.2003.08.019. URL https://doi.org/10.1016/j.atmosenv.2003.08.019.

[170] L.B. Rall. *Automatic Differentiation: Techniques and Applications*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1981. doi: 10.1007/3-540-10861-0. URL https://doi.org/10.1007/3-540-10861-0.

[171] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *CoRR*, 2015. URL http://arxiv.org/abs/1502.05767v4.

[172] Ali Hussain Motagamwala and James A. Dumesic. Analysis of reaction schemes using maximum rates of constituent steps. *Proceedings of the National Academy of Sciences*, 113(21):E2879–E2888, 2016. doi: 10.1073/pnas.1605742113. URL https://doi.org/10.1073/pnas.1605742113.

[173] Yang-Yang Song and Gui-Chang Wang. A DFT study and microkinetic simulation of propylene partial oxidation on CuO (111) and CuO (100) surfaces. *The Journal of Physical Chemistry C*, 120(48):27430–27442, 2016. doi: 10.1021/acs.jpcc.6b09621. URL https://doi.org/10.1021/acs.jpcc.6b09621.

[174] O. Rakovec, M. C. Hill, M. P. Clark, A. H. Weerts, A. J. Teuling, and R. Uijlenhoet. Distributed evaluation of local sensitivity analysis (DELSA), with application to hydrologic models. *Water Resources Research*, 50(1):409–426, 2014. doi: 10.1002/2013wr014063. URL https://doi.org/10.1002/2013wr014063.

[175] Yingbo Ma, Vaibhav Dixit, Mike Innes, Xingjian Guo, and Christopher Rackauckas. A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. *CoRR*, 2018. URL http://arxiv.org/abs/1812.01892v2.

[176] Peter Eberhard and Christian Bischof. Automatic differentiation of numerical integration algorithms. *Mathematics of Computation*, 68 (226):717–732, 1999. doi: 10.1090/s0025-5718-99-01027-3. URL https://doi.org/10.1090/s0025-5718-99-01027-3.

[177] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[178] Jarrett Revels, Miles Lubin, and Theodore Papamarkou. Forward-mode automatic differentiation in Julia. *CoRR*, 2016. URL http://arxiv.org/abs/1607.07892v1.

[179] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.

[180] Christopher Rackauckas and Qing Nie. DifferentialEquations.jl–a performant and feature-rich ecosystem for solving differential equations in Julia. *Journal of Open Research Software*, 5(1), 2017.

[181] Valentin Sulzer, Scott G Marquis, Robert Timms, Martin Robinson, and S Jon Chapman. Python battery mathematical modelling (PyBaMM). *ECSarXiv. February*, 7, 2020.

[182] I.M Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1-3):271–280, 2001. doi: 10.1016/s0378-4754(00)00270-6. URL https://doi.org/10.1016/s0378-4754(00)00270-6.

[183] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297, 2002. doi: 10.1016/s0010-4655(02)00280-1. URL https://doi.org/10.1016/s0010-4655(02)00280-1.

[184] Mary C. Hill and Claire R. Tiedeman. *Effective Groundwater Model Calibration: With Analysis of Data, Sensitivities, Predictions, and Uncertainty*. John Wiley Sons, Inc., 2005. ISBN 978-0-471-77636-9. doi: 10.1002/0470041080. URL https://onlinelibrary.wiley.com/doi/book/10.1002/0470041080.

[185] George P. Petropoulos and Prashant K. Srivastava. *Sensitivity Analysis in Earth Observation Modelling*. Elsevier, 2016. ISBN 978-0-128-03011-0. URL https://www.elsevier.com/books/sensitivity-analysis-in-earth-observation-modelling/petropoulos/978-0-12-803011-0.

[186] I.M. Sobol' and S. Kucherenko. A new derivative based importance criterion for groups of variables and its link with the global sensitivity indices. *Computer Physics Communications*, 181(7):1212–1217, 2010. doi: 10.1016/j.cpc.2010.03.006. URL https://doi.org/10.1016/j.cpc.2010.03.006.

[187] Charles T. Campbell and Zhongtian Mao. Analysis and prediction of reaction kinetics using the degree of rate control. *Journal of Catalysis*, 404(nil):647–660, 2021. doi: 10.1016/j.jcat.2021.10.002. URL https://doi.org/10.1016/j.jcat.2021.10.002.

[188] I.M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. doi: 10.1016/0041-5553(67)90144-9. URL https://doi.org/10.1016/0041-5553(67)90144-9.

A Density of the Training Data Sampling

To quantitatively measure the density of the training data sampling in Chapter 3, we also calculated the maximum and average distances between the fingerprints of the sampled points on the triangular plane in the 3D fingerprint space. Take the subsurface atoms in the slabs with lattice constant of 4.08 (shown in Figure 3.1) as the example, the corresponding equilateral triangle has side length of about 12.8 units, the distribution of the distances of points to their nearest neighbors is shown in Figure A.1. The maximum distance is 0.26 unit while the average distance is about 0.04 unit, which is quite small compared to the side length of 12.8 units.

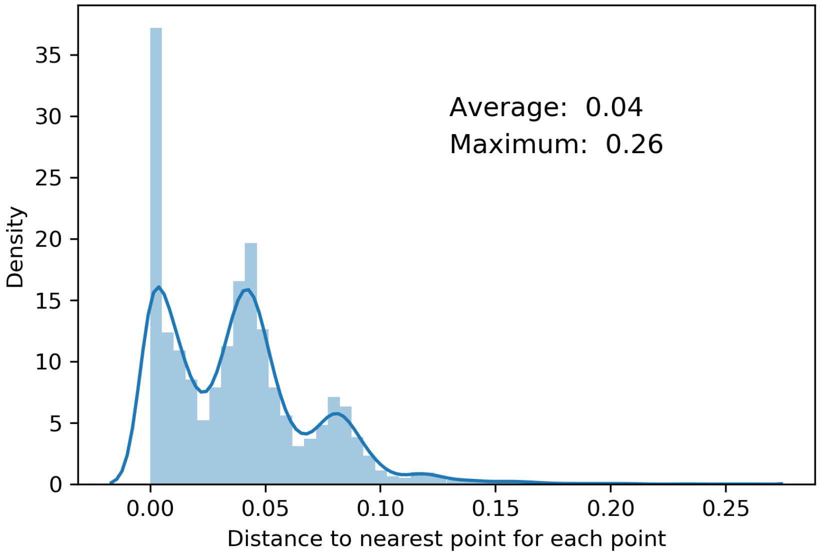

Figure A.1: Distribution of the distances to the nearest neighbors for the points in the fingerprint space.

B ACSF Parameters

In this section, we attach the parameters for the ACSFs used in Chapter 3 and Chapter 5.

Table B.1: The ACSF parameters for Chapter 3.

Type	η	R_s $(\mathring{A})$	R_c $(\mathring{A})$
G2	0.05	0.0	6.0
G2	4.0	0.0	6.0
G2	20.0	0.0	6.0

Table B.2: The ACSF parameters for Chapter 5.

Type	η	R_s $(\mathring{A})$	γ	ζ	R_c $(\mathring{A})$
G2	0.05	0.0	-	-	6.5
G2	4.0	0.0	-	-	6.5
G2	20.0	0.0	-	-	6.5
G4	0.05	-	1.0	1.0	6.5
G4	0.05	-	-1.0	1.0	6.5
G2	0.05	-	1.0	4.0	6.5
G2	0.05	-	-1.0	4.0	6.5

C AgPd Slab Configurations with Acrolein for DFT Calculation

Here, we show the 16 relaxed AgPd configurations with acrolein calculated by DFT mentioned in section Section 5.3.1.

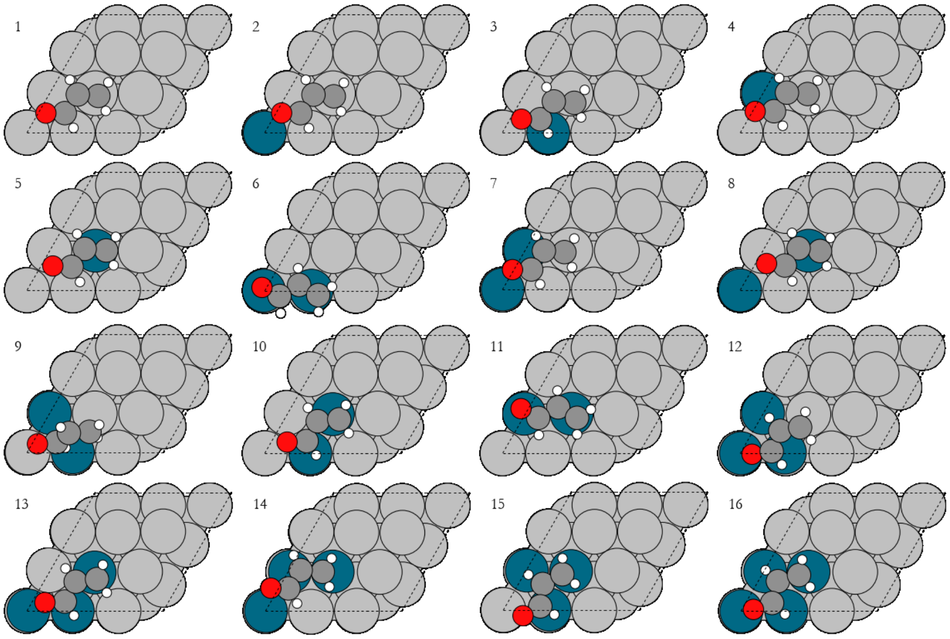

Figure C.1: Relaxed AgPd configurations with acrolein

D Gaussian Process Regression

In this section, we show the details of the Gaussian process regression used in Section 4.2.1. We adapt the Gaussian Process Regression (GPR) method from previous literature[81,82] as one of the comparisons in this work. The GPR model uses the positions of the atoms as the feature $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]$, and the model is trained on the corresponding energies ($\mathbf{e}$) and the first order derivative which is the negative forces in this application. Thus, $\mathbf{y} = [\mathbf{e}, -\mathbf{f}_1, ..., -\mathbf{f}_N]$

Therefore, the prediction function could be sampled from the Gaussian Process defined by a prior mean and a kernel function:

$$f(x) \sim \mathbb{GP}\left(\boldsymbol{\mu}, k\left(\boldsymbol{x}, \boldsymbol{x'}\right)\right) \tag{D.1}$$

where μ is the prior for the energies and forces and it is set as zero in our work. Given a training set $\mathbb{D}$, the predicted mean and variance are

$$\mathbb{E}\left[f(\boldsymbol{x}|\mathbb{D})\right] = \boldsymbol{k}(\boldsymbol{x})\left[\boldsymbol{K}(\boldsymbol{x} + \sigma_n^2 \boldsymbol{I})\right]^{-1} \boldsymbol{y} \tag{D.2}$$

and

$$\mathbb{V}\left[f(\boldsymbol{x}|\mathbb{D})\right] = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}(\boldsymbol{x})^T \left[\boldsymbol{K}(\boldsymbol{x} + \sigma_n^2 \boldsymbol{I})\right]^{-1} \boldsymbol{k}(\boldsymbol{x}) \tag{D.3}$$

where σ_n is the noise of the data.

The kernel function could be partitioned into:

$$\boldsymbol{K}(\boldsymbol{x}) = \begin{pmatrix} \boldsymbol{K}_{ee}(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{K}_{ef}(\boldsymbol{x}, \boldsymbol{x}) \\ \boldsymbol{K}_{ef}(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{K}_{ff}(\boldsymbol{x}, \boldsymbol{x}) \end{pmatrix} \tag{D.4}$$

The squared exponential kernel is used in our implementation. Thus, the formula for these kernel function are:

$$k_{ee}(\boldsymbol{x}, \boldsymbol{x'}) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d - x'_d)^2}{l_d^2}\right) \tag{D.5}$$

$$k_{fe}(\boldsymbol{x}, \boldsymbol{x'}) = -\frac{\sigma_f^2 (x_d - x'_d)}{l_d^2} \exp\left(-\frac{1}{2}\sum_{j=1}^{D}\frac{(x_j - x'_j)^2}{l_j^2}\right) \tag{D.6}$$

$$k_{ff}(\boldsymbol{x}, \boldsymbol{x'}) = \frac{-\sigma_f^2}{l_{d1}^2}\left(\delta_{d_1 d_2} - \frac{(x_{d1} - x'_{d1})(x_{d2} - x'_{d2})}{l_{d1}^2}\right) \exp\left(-\frac{1}{2}\sum_{j=1}^{D}\frac{(x_j - x'_j)^2}{l_j^2}\right) \tag{D.7}$$

where σ_f is fixed as 1.0, and the bandwidth l is optimized isotropically.

E Hyperparamters for GemNet model, Faiss searching and Flare GP model

Here, we attach the hyperparamters of the GemNet model and FAISS searching used in Chapter 6.

Table E.1: Hyperparameters for the GemNet model used in Chapter 6. More details about the hyperparameters can be found in the OC20 GitHub repository.

Hyperparameter	Value	Hyperparameter	Value
num_spherical	7	num_radial	128
num_blocks	3	emb_size_atom	128
emb_size_edge	128	emb_size_trip	64
emb_size_rbf	16	emb_size_cbf	16
emb_size_bil_trip	64	num_before_skip	1
num_after_skip	2	num_concat	1
num_atom	3	cutoff	6.0
max_neighbors	50	rbf	gaussian
envelope	polynomial	cbf	spherical_harmonics
extensive	true	oft_graph	false
output_init	heOrthogonal	activation	silu
regress_forces	true	direct_forces	true

Table E.2: Hyperparameters for the Faiss IndexIVFPQ method. More details about the hyperparameters can be found in Faiss wiki.

Hyperparameter	Value
Coarse Quantizer	IndexFlatL2
d (dimension)	128
nlists (number of centroids in coarse quantizer)	7000
m (number of subvectors for division)	128
nbits (encoding size for a subvector)	8

Table E.3: Hyperparameters for the Flare GPR model.

Hyperparameter	Value
cutoff radius	3.7
descriptors	two body and three body
kernel	square kernel
length scale	0.5
energy noise	0.005
force noise	0.005
stress noise	0.1

F Supplementary Examples for ANN Search

F.1 ANN Search Examples for Molecular Substructures

Here we attach other examples for the neural network embedding-based similar molecules search. The color-element relation in the following figures is: red for O, blue for N, white for H, and grey for C.

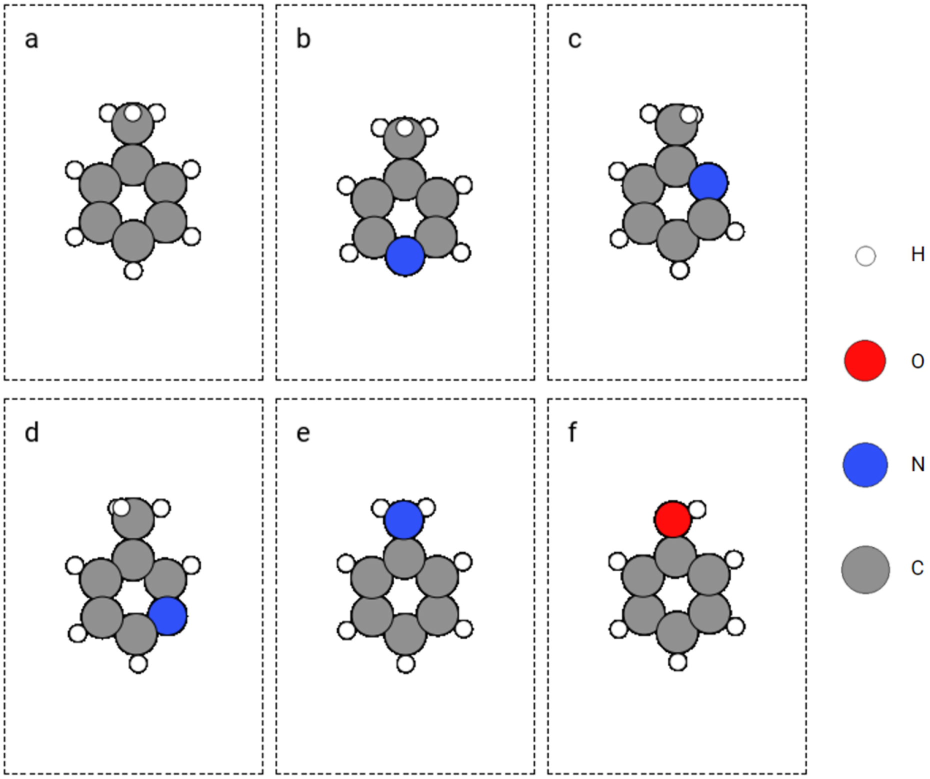

Figure F.1: Similar molecules (b to f) retrieved from querying toluene molecule (a).

F.2 ANN Search Example for Similar Bulk Environment

Here, we present another example using GemNet embedding and ANN to search for similar bulk environment in the Materials Project database. The query and searched atoms are shown in Figure F.5. The query palladium atom (atom 2 in Figure F.5 a) and two other palladium atoms (atom 3 and 4) form a hollow site and there is a zinc atom (atom 0) on this hollow site. The searched palladium atoms are all similar to the query palladium atom in terms of the atom arrangement and element type.

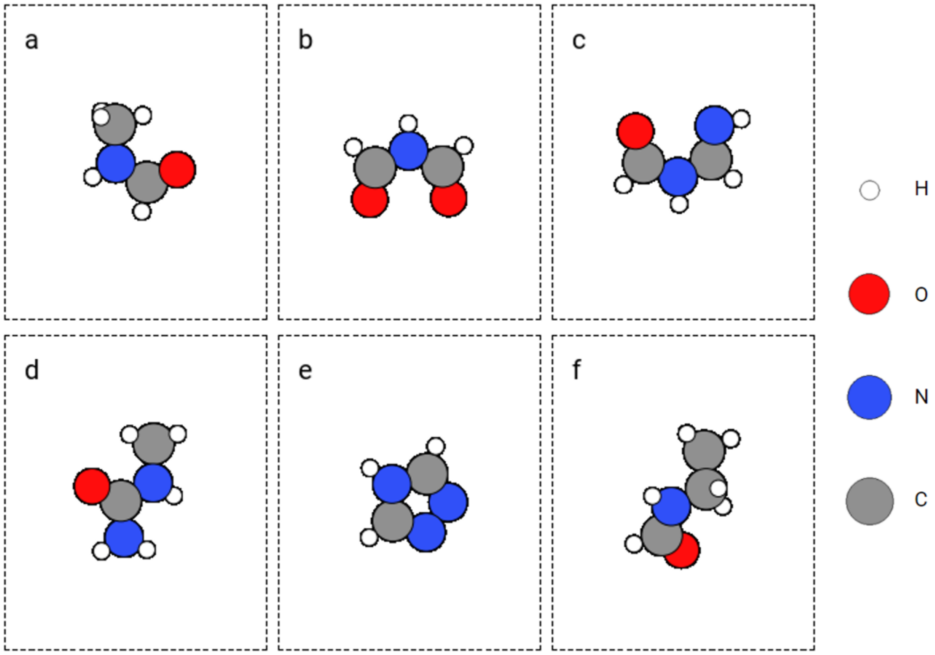

Figure F.2: Similar -N(H)- substructure (b to f) retrieved from querying -N(H)- substructure (a).

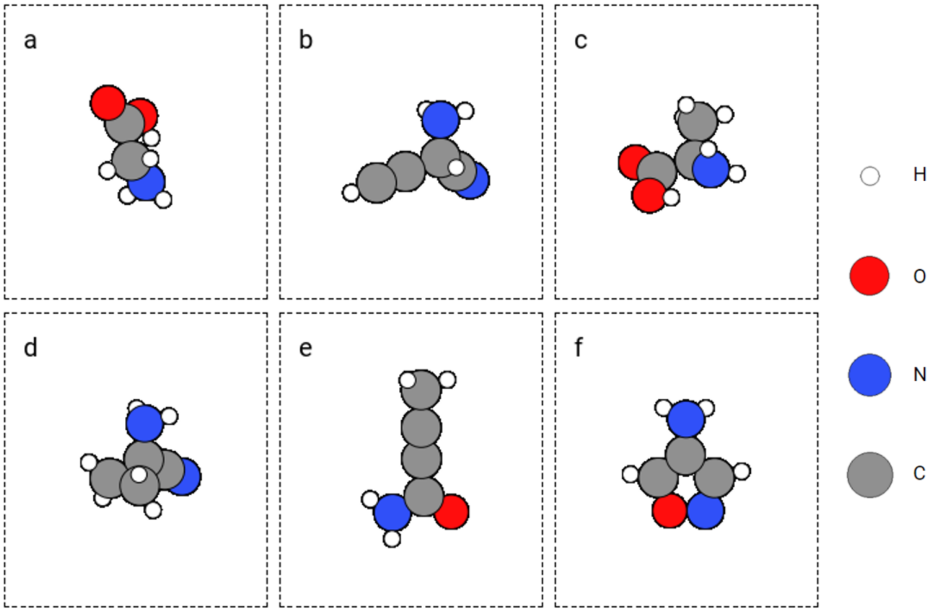

Figure F.3: Similar -NH_2 substructure (b to f) retrieved from querying -NH_2 substructure (a).

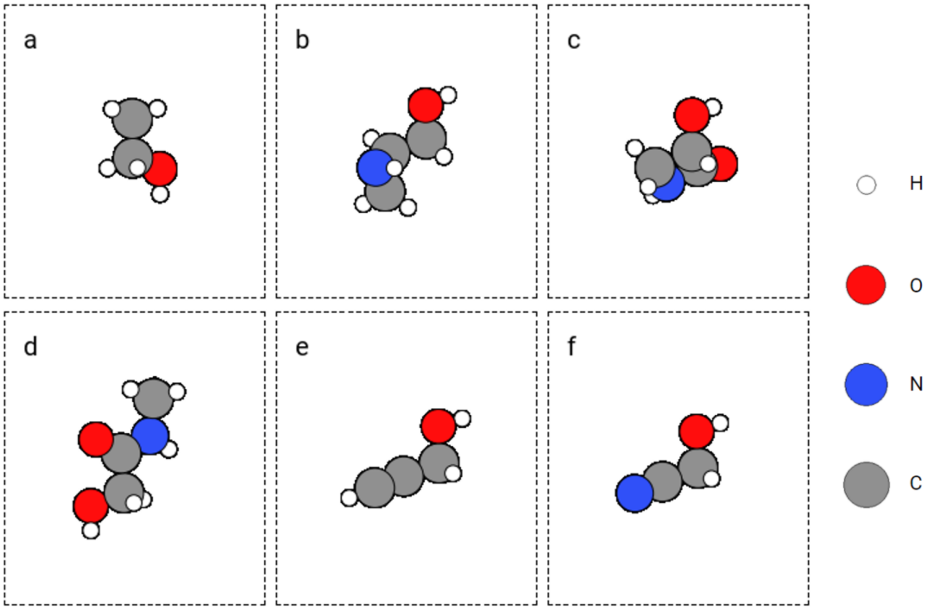

Figure F.4: Similar -OH substructure (b to f) retrieved from querying -OH substructure (a).

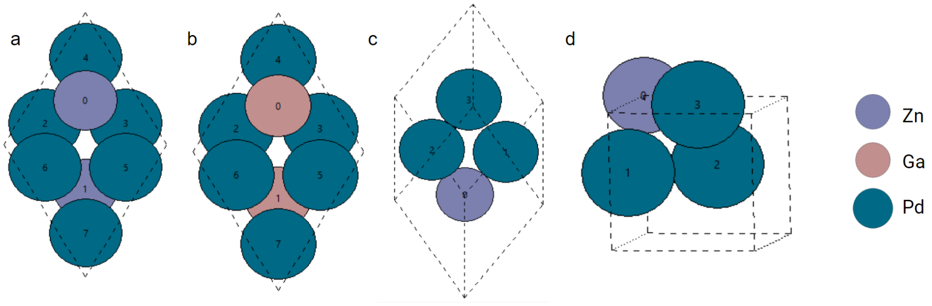

Figure F.5: Top 10 nearest atoms to a palladium query atom in the Materials Project dataset. Atom 2 in figure a is the query atom. Atom 2 to 7 in figure b, atom 1, 2 in figure c, and atom 1, 3 in figure d are the searched atoms.

F.3 Supporting Configurations for the OC20 Case

Here, we attached supporting configurations for the examples shown in Section 6.3.3, like the randomly selected configurations, zoomed-in local configurations for the query and searched acetylene adsorption system.

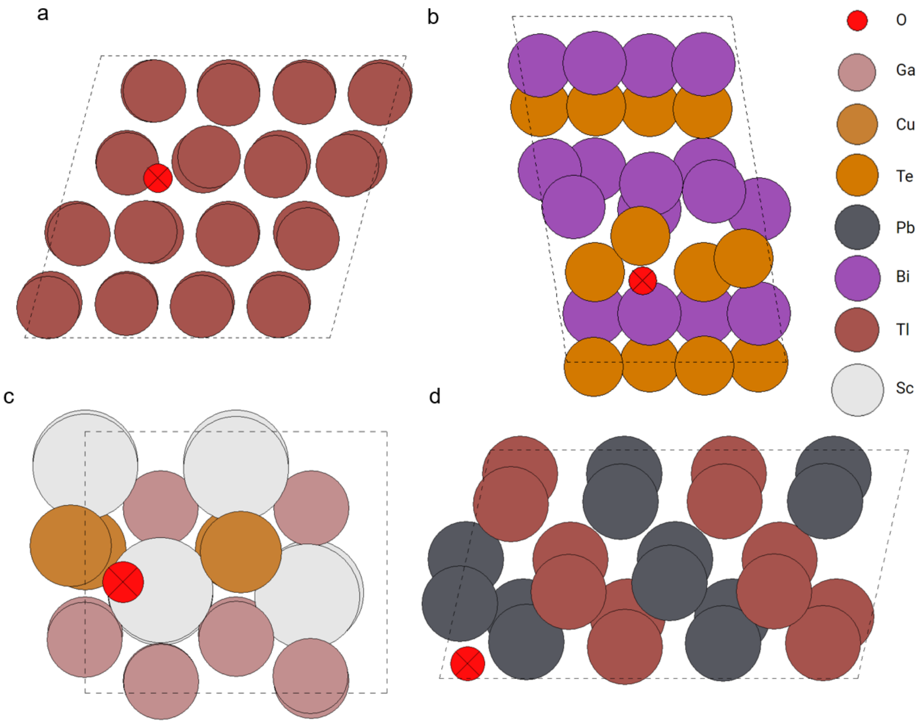

Figure F.6: Four randomly selected oxygen atoms in the OC20 dataset.

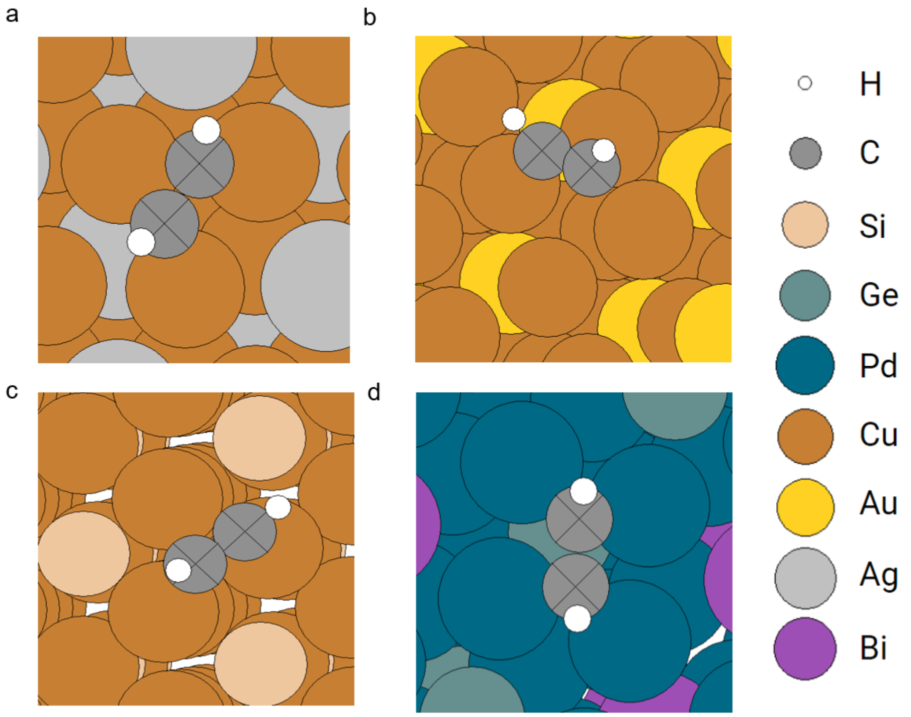

Figure F.7: Zoomed-in local configurations for the acetylene search example.

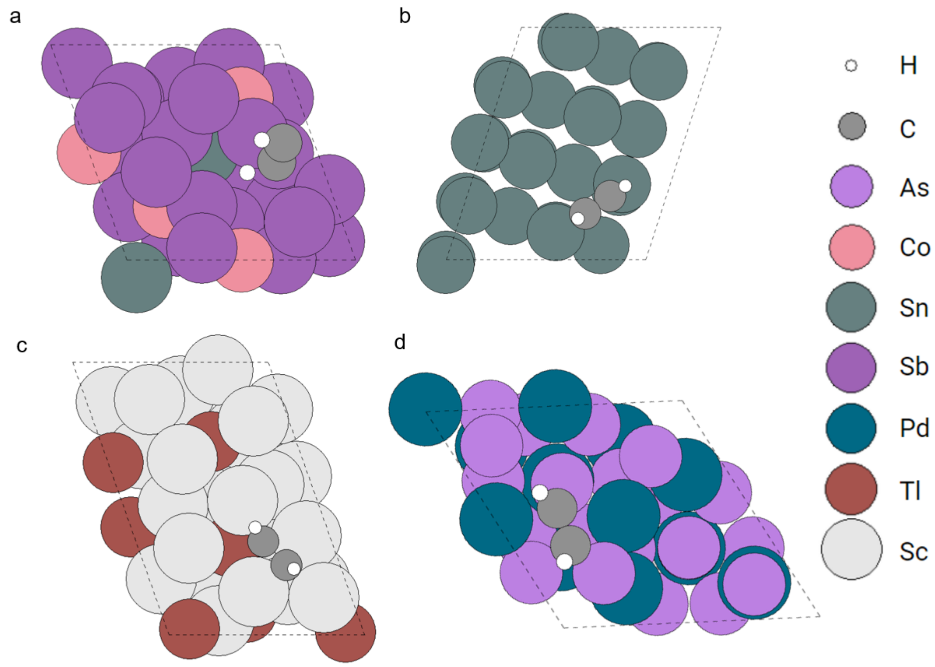

Figure F.8: Four randomly selected acetylene adsorption systems in the OC20 dataset.