CS 229 Final Report: Learning Catalysis, One Piece At A Time

Michael Tang, Philip Hwang, Robert Sandberg

December 16, 2016

1 Introduction

The CO_2 reduction reaction has garnered widespread attention in the scientific community due to its potential to both produce renewable fuels and mitigate the impact of anthropogenic CO_2 emissions. Polycrystalline copper (Cu) is the only known transition metal to produce hydrocarbons at reasonable efficiency but at a extremely large overpotential/energy cost[1]. Experimental and theoretical studies have found that the CO hydrogenation step of the reaction pathway to produce formate is the rate limiting step[2]. This is due to the high activation barrier for this reaction step, which is dependent on the CO binding energy. Therefore, the CO binding energy is the parameter that is routinely screened when searching for new active catalysts. Density Functional Theory (DFT) is the main computational tool used to screen potential candidate materials. In this paper, we aim to accelerate the material screening process with machine learning algorithms by creating a model that can accurately estimate CO binding energy on any given material surface. The inputs of our models will be features that describe the adsorbate-catalyst atom-network, and we used linear regression, k-neighbor regression, neural networks, kernel ridge regression, support vector regression, and Gaussian processes to predict the binding energy (in eV) of each atomic network.

2 Relevant works

In recent years, machine learning has become a tool used by several research groups to study catalysis, and in particular, the CO_2 reduction reaction. One recent study used a feedforward artificial neural network to construct a nonlinear mapping between their feature vector and CO adsorption energy. The primary features were electronic in nature: d-band filling, center, width, and skewness, as well as local Pauling electronegativity. This resulted in an improved error (0.1eV) compared with a two-level interaction model (0.3eV)[3]. This work was then extended by the same group to include geometric features in search of Cu 100-terminated bimetallics, aimed at producing C_2 species, obtaining similar error (0.1eV) as the previous model[4]. Zeolite catalysts have also been investigated for CO_2 reduction. A Bayesian regularized feed-forward neural network was used to identify quantitative relationships between structural characteristics and simulated adsorption properties. Top candidates were identified based on these quantitative relationships, with most having a cavity size of 6 angstroms[5]. Machine learning has also been used to help generate surface Pourbaix diagrams for catalysis[6]. Machine Learning has also been applied to create hybrid DFT atomic functionals that is properly optimized for chemisorption calculations[10].

In the Norskov group, Gaussian Process (GP) regression was used to predict CO binding energy on various facets of the Nickel-Gallium bimetallic system. The study used the coordination of surface atoms as its main set of features; however, these features do not account for the electronic structure of elemental constituents. By introducing a new set of features which include electronic structure information, we expect to expand the current model to encompass a wider variety of systems.

3 Data

Data was collected from calculations using the Quantum Espresso DFT code with the Atomic Simulation Environment (ASE)[13]. The output from DFT contains all atomic structure information, such as atomic positions and energies. Bulk atomistic structures were taken from the Materials Project and used to create surface structures[12]. Each surface was then partitioned into various molecular motifs via Voronoi decomposition, each representing a unique bonding network of 16 neighboring atoms with a CO molecule. Our current dataset has 960 examples spanning more than 80 materials systems.

4 Feature Extraction

One of the main challenges of our project was devising a way to vectorize atomic structure data from DFT to construct the dataset. Unlike atomic calculations of biomolecules, atomic structures of heterogeneous catalysts are periodic and do not have a terminal length. Thus, we needed to determine a threshold length where the sampled atoms are sufficient in describing the bonding nature of CO on any atomic surface.

4.1 The Quadtree Model

We hypothesize that the CO binding energy is a function of the attributes of its neighboring atoms such that:

$$\Delta E_{ads} = f(atom_1, atom_2, ..., atom_n)$$

where each $atom_i$ has its own a set of identifying attributes. We devised a feature vector based on a linearized layered quadtree model. A quadtree model is a tree model where each node has four child nodes. The

layered tree-model allows us to describe a network of neighboring atoms with a finite number of vector elements. We start with the zeroth layer, which represents the CO adsorbate on the surface. Each surface has, at most, 4 coordinated metal atoms to CO, so we select the 4 closest atoms to CO as the tree network's first layer. We then select four more descending neighbors of the atoms in the first layer as the second layer. We stop at 2 'layers' because 3rd order neighbors generally have very weak interactions with the adsorbate. In short, our feature vector is derived from features from each atom in each layer.

4.2 Geometric Features

Geometric features were added to describe the spatial information of the atomic network. These features include the bond distances between atoms in the first layer and CO, which helps distinguish different site types (on top, bridge, 3-fold, 4-fold). In addition to bond distances, we include the polar coordinates of the 4 neighboring atoms to gain angular information of those atoms relative to the adsorbate.

4.3 Electronic Features

Electronic features help describe each motif in terms of its electron density, a distinction that geometric features ignore. The electronic features include the element's group, quantum angular momentum 'l', valence orbital fill factor, and electron population, all taken from DFT calculations of the material's bulk structure.

4.4 Dimensionality reduction

Dimensionality reduction was performed manually based on literature suggestions[7, 8]. We removed atomic radii, shielding, and Pauling electronegativity features because they did not provide enough information about electron interactions involved between elements; DFT data provide a clearer picture of the changes that occur in electron interactions between different elements.

After assigning features to each atom, the feature vector was further reduced by taking means and sums of appropriate quantities. This allows for the creation of aggreagated data for each layer. We can then describe the interaction between the first and second layer more compactly, to help reduce the risk of overfitting.

In total, each example has 56 features. Table 1 provides a full description of the resulting feature vector in our study.

Table 1: Full Feature Table by Vector Index

I	Feature	I	Feature
1	Atom 1 Element Group	29	ΣAll Electron-states in 1st Layer
2	Atom 1 Polar Angle	30	ΣFilled Electron-states in 1st Layer
3	Atom 1 Azimuth Angle	31	1st Layer S-shell Fill Factor
4	Atom 1 Bond Distance	32	1st Layer P-shell Fill Factor
5	Atom 1 Coordination	33	1st Layer D-shell Fill Factor
6	Atom 1 Quantum Ang. Mom.	34	1st Layer F-shell Fill Factor
7	Atom 2 Element Group	35	% of 1st layer electrons in S-shell
8	Atom 2 Polar Angle	36	% of 1st layer electrons in P-shell
9	Atom 2 Azimuth Angle	37	% of 1st layer electrons in D-shell
10	Atom 2 Bond Distance	38	% of 1st layer electrons in F-shell
11	Atom 2 Coordination	39	Avg. Element Group in 2nd Layer
12	Atom 2 Quantum Ang. Mom.	40	Avg. Bond Distances in 2nd Layer
13	Atom 3 Element Group	41	Avg. Coordination in 2nd Layer
14	Atom 3 Polar Angle	42	Avg. Quant. Ang. Mom. in 2nd Layer
15	Atom 3 Azimuth Angle	43	ΣS-shell Electrons in 2nd Layer
16	Atom 3 Bond Distance	44	ΣP-shell Electrons in 2nd Layer
17	Atom 3 Coordination	45	ΣD-shell Electrons in 2nd Layer
18	Atom 3 Quantum Ang. Mom.	46	ΣF-shell Electrons in 2nd Layer
19	Atom 4 Element Group	47	ΣAll Electron-states in 2nd Layer
20	Atom 4 Polar Angle	48	ΣFilled Electron-states in 2nd Layer
21	Atom 4 Azimuth Angle	49	2nd Layer S-shell Fill Factor
22	Atom 4 Bond Distance	50	2nd Layer P-shell Fill Factor
23	Atom 4 Coordination	51	2nd Layer D-shell Fill Factor
24	Atom 4 Quantum Ang. Mom.	52	2nd Layer F-shell Fill Factor
25	ΣS-shell Electrons in 1st Layer	53	% of 2nd layer electrons in S-shell
26	ΣP-shell Electrons in 1st Layer	54	% of 2nd layer electrons in P-shell
27	ΣD-shell Electrons in 1st Layer	55	% of 2nd layer electrons in D-shell
28	ΣF-shell Electrons in 1st Layer	56	% of 2nd layer electrons in F-shell

5 Methodology

We made use of three classes of models from the sklearn Python package[11]: Gaussian Process (GP), kernel-based methods, and Neural Networks (NN).

5.1 Data Preprocessing

Since some training examples were generated by "straining" some data points (much like spatial operations on images for image recognition problems), we cautiously split our dataset to ensure that examples in the test set were independent of those in the training set. We also normalized the data set prior to fitting using kernel-based models and the Gaussian Process model. Interestingly, we found that standardizing the data for a neural network provides a drastic improvement in performance.

5.2 Gaussian Process

We modeled our training set with GP regression, which is a form of a kernel-based Bayesian regression algorithm.

For predictions, GP generates a prior distribution over functions $f(\cdot)$ based on some covariance function $k(\cdot,\cdot)$ such that

$$f(\cdot) \sim GP(0, k(\cdot,\cdot)) \tag{1}$$

and attempts to use these functions to fit the sample data by optimizing these functions with the maximum log-likelihood equation given below:

$$L = -\frac{1}{2}(log\,|K| + y^T * K^{-1} * y) \tag{2}$$

For the Gaussian Process, we used the Gaussian Kernel. The elements of the kernel can be calculated from the sample data given in the equation below.

$$k(x,y) = exp(-\frac{\|x-y\|^2}{2\sigma^2}) \tag{3}$$

where we can tune parameters in A^{-1} to adjust for overfitting with the Gaussian Kernel. We choose to use GP regression because it is reasonable to assume that our dataset comes from a multivariate distribution; the data contains several adsorbate configurations on several surfaces for a few dozen material systems. GP attempts to predict new data based on the most probable covariance function from the training data and uses it to predict the binding energy on a new motif. Furthermore, since GP

regression is kernel-based, we hypothesize that it can take advantage of our tree-based data structure. We used a constant GP regression model; the prior distribution $f(\cdot)$ has a constant mean while taking the Gaussian kernel to be the covariance function.

5.3 Support Vector Regression

We used support vector regression, primarily for its ability to use kernel methods to model non-linear behavior in the binding energy. We attempt to find the hypothesis $h_{w,b}(x) = w^T x + b$, where our regularized optimization problem is

$$\min_{w,b} \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*),$$

s.t.

$$y^{(i)} - w^T x^{(i)} - b \leq \epsilon + \xi_i, i = 1, \ldots, m$$
$$w^T x^{(i)} + b - y^{(i)} \leq \epsilon + \xi_i^*, i = 1, \ldots, m$$
$$\xi_i, \xi^* \geq 0,$$

where $\epsilon > 0$ is fixed, $x^{(i)}$ are the inputs, and $y^{(i)}$ are the labels for binding energy. We used cross-validation for our SVR to select appropriate values for C to avoid underfitting or overfitting. We used a Laplacian Kernel, which is similar to the Gaussian Kernel but the square of the norm is removed and the kernel is less sensitive to changes in the σ parameter. The elements of the kernel can be calculated from the sample data given in the equation below.

$$k(x, y) = exp(-\frac{\|x - y\|}{\sigma})$$

5.4 Kernel Ridge Regression

Kernel Ridge Regression is the kernelized version of Ridge Regression where we apply the kernel trick to the loss function:

$$J = \frac{1}{m}\sum_{i=1}^{m}(L(\sum_{j=1}^{m}\alpha_j * K(x^{(i)}, y^{(i)}), y^{(i)}) + \frac{\lambda}{2}\sum_{i,j}\alpha_i\alpha_j * K(x^{(i)}, y^{(i)})$$

then optimize for α values. We used the same kernel that we used for SVR in the KRR model.

5.5 k-Neighbor Regression

We used k-Neighbor Regression ($k = 2$) as a baseline to test the notion that new motifs should regress based on "similar" motifs from the training set. Rather than computing with a kernel, new data is simply regressed based on the Manhattan distance of k neighboring points:

$$Dis(x_a, x_b) = \sum_{i=1}^{n}|x_{a,i} - x_{b,i}|$$

where x_a and x_b are points in our input set.

5.6 Fully Connected Neural Networks

We used the Multi-layer Perceptron as our neural network model for the study. In this algorithm, our features are put into linear combination and fed into hidden nodes, where they are passed into tanh activation functions. This process propagates forward through the network until a linear combination is finally given to the output layer. In each hidden node, a linear combination of inputs is passed into a tanh activation function. The weights are optimized via backpropagation and gradient descent on the l2 squared-error cost function:

$$\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{k}w_j^2,$$

where y_i is the prediction, $\hat{y}_i$ is the label, and w_j are the weights.

5.7 Linear Regression

As another benchmark, we applied a linear regression model to our dataset. To do so, we solve the normal equations for linear regression:

$$\theta = (X^T X)^{-1} X^T y$$

5.8 Cross Validation

We split our dataset to create separate training and test data, where 90% is for training and 10% is for testing purposes. The training set was further split into 25 samples for cross-validation purposes, using Shuffle-splitting for kernel-based methods and K-fold for the Neural Network. For Gaussian Processes, we found that the most appropriate regularization parameter σ for the RBF kernel function was 0.1. For SVR, we found that the most appropriate cost parameter was C=3. For both Support Vector Regression and Kernel Ridge Regression, we found the optimal gamma value was 0.001. For our neural network approach, we did an extensive hyperparameter search for hidden layer dimensions, activation function, and l2 penalty term, and obtained a network with two hidden layers of dimensions 29 and 8 for tanh activation function and $\lambda = 15$ l2 penalty term, respectively.

5.9 Error Metric

We used root-mean-squared-error (RMSE) as our testing metric to compare our models with DFT calculated energies:

$$\text{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}.$$

where m is the number of examples, and $(y_i - \hat{y}_i)$ is the difference between the predicted CO binding energy and the CO binding energy calculated from DFT.

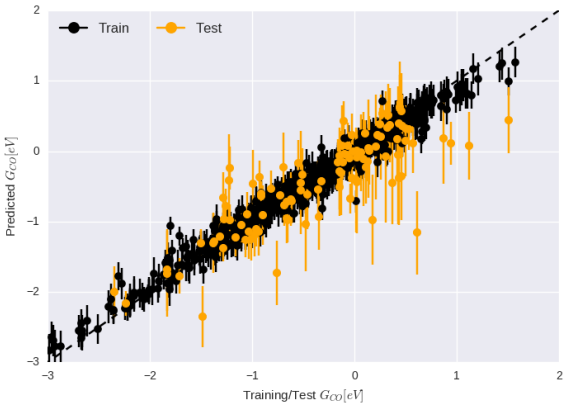

Figure 1: Gaussian Process Parity Plot

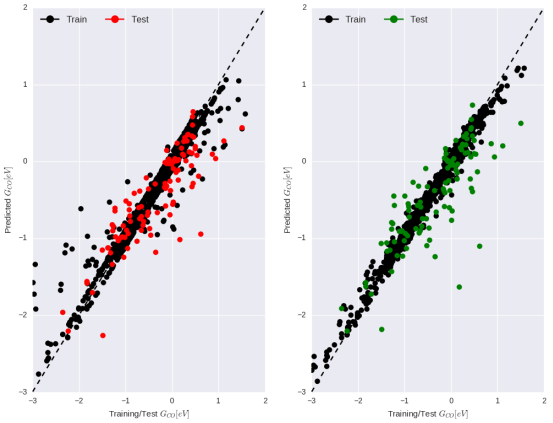

Figure 2: Left: SVR Parity Plot. Right: KRR Parity Plot

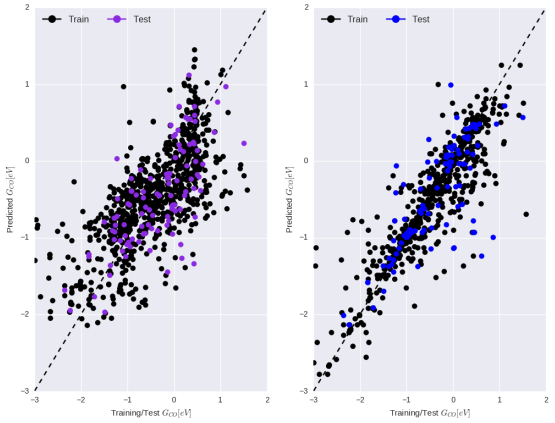

Figure 3: Left: Linear Regr. Parity Plot. Right: K-Neighbor Regr. Parity Plot

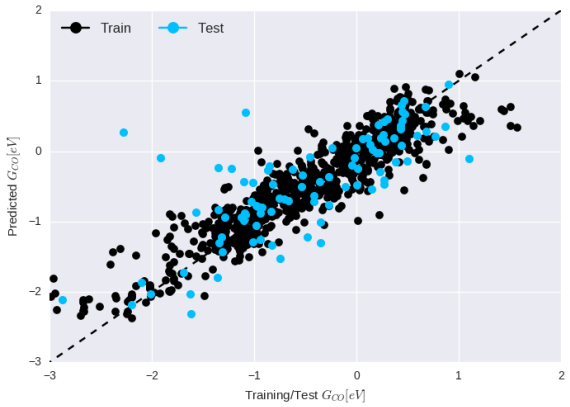

Figure 4: Neural Network Parity Plot

6 Results And Discussion

Parity plots reflect the model performance. From both linear regression and k-neighbor regression, we can see that the predicted test points were scattered. This helps confirm that predicting CO binding energy is not as simple as a linear relation nor is it an entirely local phenomena. From the SVR and KRR parity plots, we can see the models are seemingly overfit; their training points are very close to actual values, yet their test points are slightly scattered. Both the gaussian process and neural networks exhibited the best test error.

We can compare the RMSE values from the test set in Table 2 to the RMSE values for chemisorption in Figure 5. Our models exhibit an average RMSE of 0.4eV. Given the size of the dataset, this is quite good. The mBEEF[10] functional, which is a ML-based DFT functional, has an accuracy of 0.2eV. This functional was trained on a chemisorption dataset that was generated from quantum physics calculations which give much better accuracy than DFT.

6.1 Learning Curve

We can plot the learning curve for every method to qualitatively gauge whether the errors for these methods can be improved with more sample data. We plot the training error with the cross validation error for each model, splitting 50 samples from the training set. As shown in Figure 6, only our linear regression model appears to have plateau at just around 850 samples. All other methods have yet to plateau. We may still be able to improve the performance for these models by adding more data to the data set.

6.2 Feature Importance

We scored the usefulness of each feature by training each model such that we remove one feature at a time. Figure 7 allows us to see how removing features affects the score of the model. As shown, all models weighed some electronic features in the 2nd layer as important for predicting CO binding energy. In particular, both SVR and KRR seem to weigh electronic features more heavily than geometric features. GP and k-neighbor regression

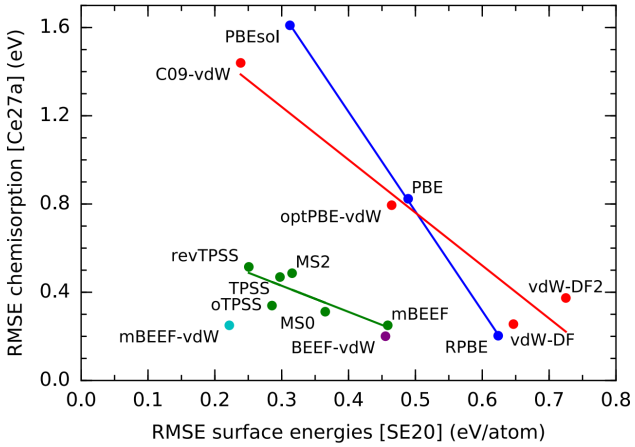

Figure 5: DFT Functional Error [10]

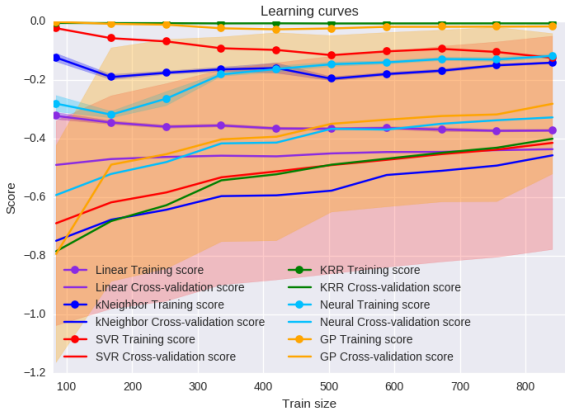

Figure 6: Learning Curves for each model

Model	**Train** [eV]	**Test** [eV]
Gaussian Proc.	0.178	0.432
Supp. Vec. Regr	0.094	0.495
Kernel Ridge Regr.	0.084	0.496
Neural Networks	0.354	0.371
k Neigh. Regr	0.394	0.690
Linear Regr.	0.607	0.667

Table 2: Errors by Model.

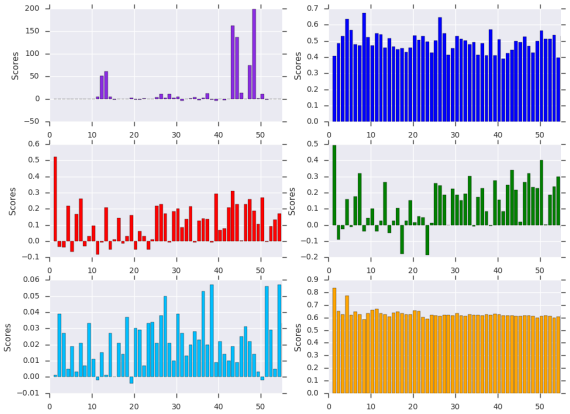

Figure 7: Feature Scores for each model. Purple: Linear Regression; Blue: k-Neighbors Regression; Red: SVR; Green: KRR; Aqua: Neural Networks; Yellow: Gaussian Process

appear to actively use both geometric and electronic features. Note that our neural network model did not make good use of geometric features, which hints at some areas of improvement in the future.

6.3 Outliers

Another probable cause for any increase in test error might not be inherent to the models used, but rather that the features are unable to describe other chemical phenomenon. Our feature set hinges on the idea of predicting CO binding energy based on a neighboring atom network. For instance, if the CO molecule prefers not to bind to the surface and instead desorbs, our feature set can only describe the repulsive interaction between the surface and the CO molecule with arbitrary distance and no electronic component.

7 Conclusions And Future Work

We have demonstrated promising results, particularly for our kernel-based methods and neural networks approach. These results may ultimately be able to reduce the computational power necessary to screen for CO_2 reduction catalysts.

Our highest performing method was the neural network, yielding a test error of .371 eV. This, perhaps, is due to the ability of deep neural networks to model higher level abstractions better than our other methods. Our kernel based methods also performed substantially well. We believe this is primarily due to the ability of kernels to map our inputs into a higher dimensional space and to model non-linearities within the data set. Our worst performing model was k-neighbor regression, as it benchmarked a .690 eV error. We believe this is due to the high-dimensionality nature of our data, the same reason for why linear regression performed poorly.

There are many routes for future work. For one, we can try refining our feature set in describing the bonding nature of adsorbates to the surface. Our current featurization does not describe the effects of very strong or weak binding catalysts, where many non-trivial effects may arise. Another route would be to reexamine our deep learning approach. Namely, with more computing resources we could try expanding the data set size and restructure our features to apply more sophisticated architectures such as convolutional neural networks, which have been successful in structure-based high-throughput drug discovery[9]. We believe convolutional neural networks might be more suitable for this problem primarily due to the spatial dependencies of the geometric features, which were not well modeled by the fully connected network.

References

[1] Hori et al. Electrochemical CO2 Reduction on Metal Electrode. *Modern Aspects of Electrochemistry* 2008.

[2] Peterson et al. How copper catalyzes the electroreduction of carbon dioxide into hydrocarbon fuels. *Energy Environ. Sci.* 2010.

[3] Ma et al. Machine-Learning-Agumented Chemisorption Model for CO_2 Electroreduction Catalyst Screening. *J. Phys Chem. Lett.* 2015.

[4] Li et al. Feature engineering of machine-learning chemisorption models for catalyst design. *Catal. Today* 2017.

[5] Thornton et al. Towards computational design of zeolite calysts for CO_2 reduction. *RSC Adv.* 2015.

[6] Ulissi et al. Automated Discovery and Construction of Surface Phase Diagrams Using Machine Learning. *J. Phys Chem. Lett.* 2016.

[7] Wolverton et al. How to choose fingerprints for computational chemistry. *Energy Environ. Sci.* 2010.

[8] Balachandran et al. Materials Prediction via Classification Learning. *Energy Environ. Sci.* 2010.

[9] Wallach et al. AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery. *arXiv Preprint* 2015.

[10] Lundgaard et al. mBEEF-vdW: Robust fitting of error estimation density functionals *Phys. Rev. B 93* 2016.

[11] Pedregosa et al. Scikit-learn: Machine Learning in Python *Journal of Machine Learning Research* 2011.

[12] Jain et al. The Materials Project: A materials genome approach to accelerating materials innovation.*APL Materials* 2013.

[13] Bahn et al. An object-oriented scripting interface to a legacy electronic structure code. *Comput. Sci. Eng.* 2002.