Predicting the convergence behavior of electronic structure calculations

BACHELORARBEIT

zur Erlangung des akademischen Grades
Bachelor of Science
(B. Sc.)
im Fach Physik

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät
Institut für Physik
Humboldt-Universität zu Berlin

von
Frau Sopio Bregadze
geboren am 27. Oktober in 2000

Gutachter:

1. *Prof. Dr. Claudia Draxl*
2. *Prof. Dr. Christoph Koch*

Contents

Abstract

Highly precise density functional theory calculations are computationally expensive. This thesis investigates how different combinations of calculation parameters affect the calculations' precision and computational cost. For density functional theory data of 69 materials, comprising binary semiconductors and metals, I use machine-learning (ML) methods to predict the precision for given computational parameters. This work helps to build a framework to support users to recommend the selection of settings that give them a desired numerical precision of a material property with the lowest computational cost.

1 Introduction

Density functional theory (DFT) is one of the most common computational methods to simulate crystalline materials. Each numerical solution is calculated using a set of computational parameters (e.g. the basis set size), with respect to which the result needs to be converged. Convergence means that a material property does not change significantly when using parameters requiring a more expensive calculation. For instance, in Ref. [**Carbogno**; **2022**] the basis set size is increased until the total energy deviates less than $10E-4eV/atom$ at which point the authors consider the calculation to be converged. In this thesis, I aim to estimate the level of convergence of a calculation using machine-learning methods. In particular, I use a dataset from Ref. [**Carbogno**; **2022**] obtained with density-functional theory (DFT) using the software package FHI-aims. This dataset, which is available in the NOMAD database [**NOMAD**], contains calculations for 69 binary materials with in total, 15,313 calculations.

I concentrate on how different computational settings can affect the convergence of the electronic density of states (DOS). In Ref. [**Speckhard**; **2023**] the authors trained a ML model to predict how far the total energy is from the converged DFT value based on the basis set size. Here, I approach a more complicated problem, since the DOS is not a simple scalar but a 2-dimensional vector. I make use of a density of states fingerprint and similarity metric from the literature [**Kuban**; **2022**] to compare how similar two DOSs are. I train a machine learning model to predict how similar a DOS is to the well converged DOS counterpart. The goal is to find a model which can accurately predict the convergence level, based on the input parameters of the DFT calculation. This will help users avoiding unnecessarily computationally expensive numerical settings.

I have trained machine-learning models such as Gaussian regression, kernel ridge regression, decision trees, random forests and boosted trees, so that the resulting models can receive a set of computational parameters and estimate the numerical precision of a DOS (i.e. proximity to convergence) based on this input. First, the DOS data were represented as fingerprints. This is outlined in Section 3. Then, the degree of convergence was assigned a similarity S. The similarity metric is described in Section 3. The similarities depend on the computational parameters used in the calculations.

Initially, I trained and tested ML models on a dataset containing calculations for a single material, MoC. The five best-performing models were picked to be then trained and tested on 68 materials remaining in the original dataset, where each material was treated as a distinct dataset. The performance of the models was analysed again and the two best performing models were later trained and tested on the entire dataset including all materials. Due to the low accuracy of the resulting predictions, the data were split into subsets according to specific material properties. Specifically, the splits were based on whether the material is a metal or which space group it belongs to. These subsets of materials were evaluated based on how accurately their convergence levels could be predicted. The importance of various computational parameters for determining the level of convergence was also analysed.

In this thesis, I first describe DFT. Then, I introduce the DOS fingerprints. It is followed by a description of machine-learning methods that are used in this thesis and how their performance is evaluated. Lastly, I present and discuss the results.

2 Density Functional Theory

The most established method for computing material properties is Density Functional Theory (DFT). In this approach, the many-body problem of interacting electrons is reduced to an effective single-body problem. This can be justified by the Hohenberg-Kohn theorem [**Hohenberg**; **Kohn**; **1964**], which states that the energy E of a system of interacting electrons can be expressed by its

density alone. The total energy $E[n]$ is thus given by the sum of the unknown density functional $F[n]$ and the contribution of the external potential $v_{ext}(\vec{r})$:

$$E[n] = F[n] + \int d\vec{r} v_{ext}(\vec{r}) n(\vec{r}) \tag{1}$$

In 1965, Kohn and Scham constructed a fictitious system of non-interacting electrons with the identical electron density as the exact many electron system [**Kohn**; **Scham**; **1965**]. This equation reduces the Schrödinger equation for the N-particle system to a single-particle equation:

$$\left(-\frac{\hbar^2}{2m_i}\Delta - v_{eff}(\vec{r})\right)\phi_i = \epsilon_i \phi_i \tag{2}$$

Here, ϕ_i and ϵ_i represent the single-particle orbitals and corresponding energies, respectively. The effective potential is the sum of three components: the external potential v_{ext}, which corresponds to the Coulomb potential of the nuclei, the Hartree potential v_H, resulting from classical Coulomb interaction (computed using the Poisson equation), and the exchange-correlation potential v_{XC}, which has no closed-form expression:

$$v_{eff}(\vec{r}) = v_{ext} + v_H + v_{XC} \tag{3}$$

The exchange-correlation potential is defined by:

$$v_{XC} = \frac{\delta E_{XC}}{\delta n(\vec{r})} \tag{4}$$

The exact expression for the exchange-correlation energy E_{XC} remains unknown and thus needs to be approximated. There are many levels to calculate the exchange-correlation energy, one of them is the local-density approximation (LDA). This method assumes that E_{XC} equals the exchange-correlation energy of a homogenous electron gas and thus is solely dependent on the density:

$$E_{XC}^{LDA}(\rho) = C \int \rho^{\frac{4}{3}}(r) d^3r \tag{5}$$

Another popular approximation is the generalized gradient approximations (GGA), which is more accurate for a variety of properties. In this approach, E_{XC} is dependent on both the density and its gradient according to:

$$E_{XC}^{GGA}(\rho) = \int \epsilon_{XC}(\rho, \nabla\rho)\rho(r) d^3r \tag{6}$$

Different exchange-correlation functionals can be displayed on the Jacob's ladder of DFT [**Perdew; 2001**]. Its purpose is to visually illustrate that various exchange-correlation functionals can be grouped together based on the required input. More accurate functionals require more computational resources to solve the Kohn-Sham equations.

Once an appropriate approximation has been chosen, an iterative process begins in which an initial electron density $n(\vec{r})$ is used to calculate the Hartree potential v_H (using the Poisson equation), to then compute the effective potential with the Eqs. 3 and 4, which is then used to solve Eq. 2. The resulting Kohn-Sham wave functions are subsequently employed to compute the new electron density:

$$n(\vec{r}) = \sum_{i=1}^{N} |\phi_i(\vec{r})|^2 \tag{7}$$

The procedure is repeated until the changes in the density are smaller than a desired limit, i.e. convergence is achieved.

From the Kohn-Sham eigenvalues the electronic density of states (DOS) can be derived. It is defined as the number of electronic states, $N(E)\delta E$, per unit-cell volume V

$$D(E) = \frac{N(E)\delta E}{V} \tag{8}$$

whose energies are between E and $E + \delta E$. An example of a DOS is shown in Fig. 1.

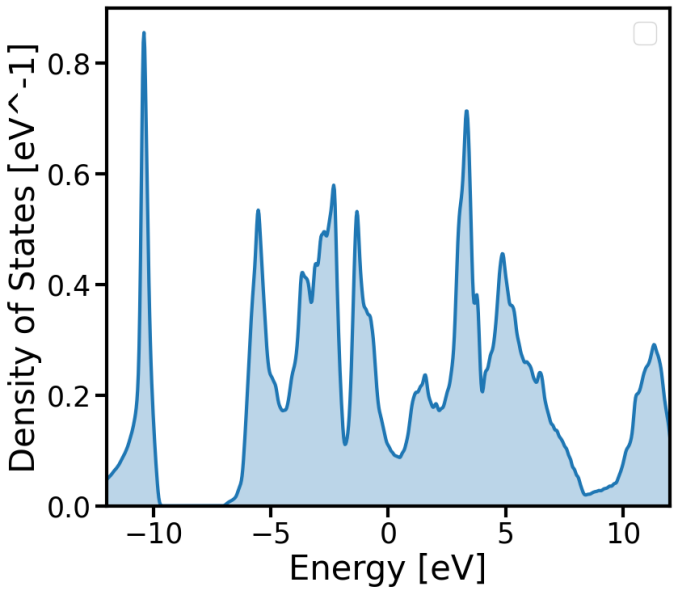

Figure 1: Density of states of MoC as a function of energy.

Many solids exhibit a band gap, which means that no electronic states exist in a region around the Fermi energy. If this gap is of the order of a few eV, the solid is classified as a semiconductor. If the gap is larger, the solid is classified as an insulator. Metals don't exhibit such gap and can conduct electricity.

3 Density of States Fingerprints

Instead of directly working with the density of states (DOS), the fingerprint proposed in [**Kuban**; **2022**] is employed in this work. This is achieved in the following way and illustrated in Fig. 2: The energy axis is divided into N intervals with variable widths of $\Delta\epsilon_i$ (see Eq. 9). A reference energy ϵ_{ref} is introduced to focus on the spectrum around this particular energy. The intervals are more dense around this value.

$$\epsilon_{i+1} - \epsilon_i = \Delta\epsilon_i = n(\epsilon_i, W, N)\epsilon_{min} \tag{9}$$

The density of states $\rho(\epsilon)$ is then integrated within each of these intervals:

$$\rho_i = \int_{\epsilon_i}^{\epsilon_{i+1}} \rho(\epsilon) d\epsilon \tag{10}$$

On the other axis, each column is discretised using the subsequent intervals:

$$\Delta\rho_i = n(\epsilon_i, W_H, N_H)\rho_{min} \tag{11}$$

where

$$n(\epsilon_i, W, N) = \lfloor g(\epsilon_i, W)N + 1 \rfloor \in [1, N] \tag{12}$$

and

$$g(\epsilon, W) = 1 - exp\left(\frac{-\epsilon^2}{2W^2}\right) \tag{13}$$

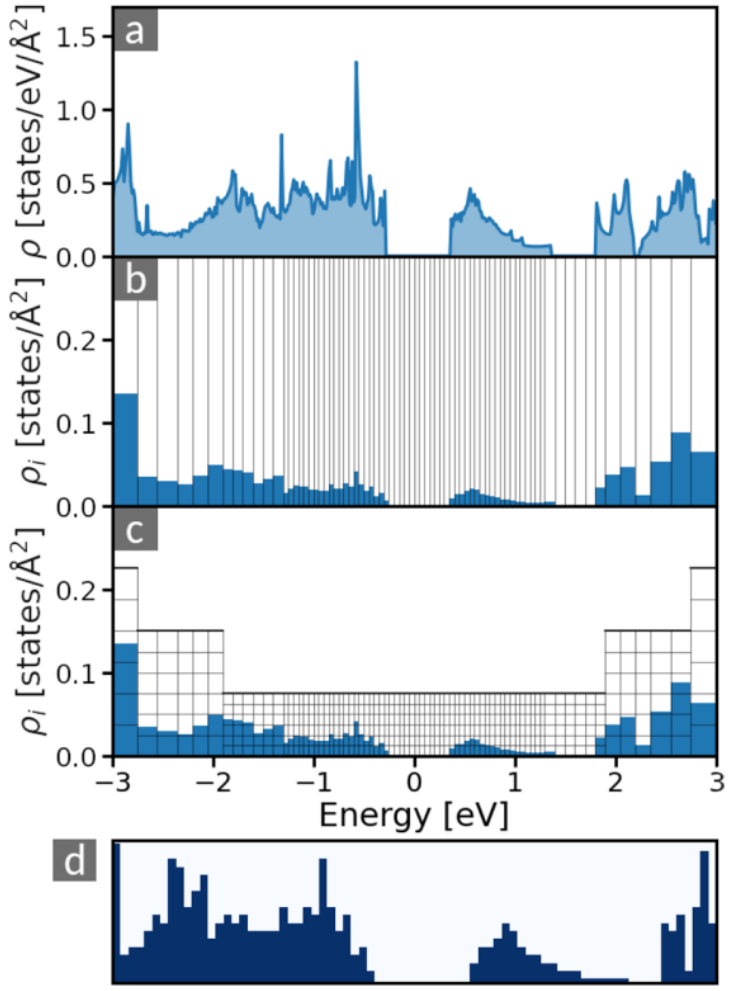

Figure 2: Construction of a DOS fingerprint. (a) Original DOS; (b) graph after the energy axis was discretised and the DOS numerically integrated over the intervals; (c) graph after both axes being discretised; (d) final fingerprint. Figure from Ref. [**Kuban**; **2022**].

This approach offers two primary advantages. Firstly, the fingerprint comprises less numerical information, thereby consuming less memory and yielding faster computations. Secondly, for practical purposes, characteristics like the magnitude of the band gap or the density of states in the vicinity of the Fermi energy are more crucial than the entire spectra. Due to this, I utilised the fingerprint with a non-uniform energy axis discretisation in order to concentrate on the required energy range.

There are several more parameters that can be adjusted to create different fingerprints of the same graph. I made slight adjustments to the default settings of these parameters to optimize the number of unfilled grid cells and the amount of information left outside of the grid. The

parameters employed are: reference energy ϵ_{ref} of -2 eV; $\Delta\epsilon_i$ between 0.26 eV and 3.11 eV; $\Delta\rho_i$ between 0.05 and 1.06; a value of 7 for W; and a pixel count of 251.

To compare two given fingerprints a similarity metric has to be utilised. In this thesis the Tanimoto metric [**Tanimoto**; **1958**] is used, where the similarity of two DOS's, f_i and f_j, is:

$$S(f_i, f_j) = \frac{f_i f_j}{\mid f_i \mid^2 + \mid f_j \mid^2 - f_i f_j} \tag{14}$$

4 Machine-Learning Methods

Machine learning, as a field, develops algorithms that allow for data-driven decision making [**El Bouchefry**; **2020**]. Machine learning algorithms fall into two main categories: supervised and unsupervised learning. In unsupervised learning, a computer program is provided with unlabeled data and attempts to identify patterns within it. In the case of supervised learning, the computer receives input data with corresponding output and aims to find a function that determines how the output depends on the input. The entire dataset is usually divided into the training and the test data. The data given to the machine learning model to improve it's performance is referred to as the training set. When evaluating the accuracy of a function in supervised learning, the algorithm is provided with data that is has not yet seen before for output prediction, and comparisons are made between these predicted outputs and the desired ones. This unseen data is called the test set.

In this section, various machine learning models are introduced. I begin with linear models, proceed to non-linear models, and then focus on tree-based algorithms.

4.1 Linear Models

Linear models of machine learning aim to fit the following equation:

$$\mathbf{f(x)} = \mathbf{wx} \tag{15}$$

where $\mathbf{w} = (w_0, w_1, w_2, ...w_p)^T$ are the weights that the algorithm determines. $\mathbf{x} = (\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_n})^T$ are the input values, with each $\mathbf{x_i} = (1, x_{i1}, x_{i2}, ..., x_{ip})$, $\mathbf{f(x)} = (\mathbf{f(x_1)}, \mathbf{f(x_2)}, ...\mathbf{f(x_n)})^T$ being a linear function. One type of a linear model is least-squares regression, which is a process of fitting Eq. 15 by minimizing the squared distance between the predicted and true values. The cost function (i.e. the function that this model aims to minimize) can be expressed as

$$\mathbf{j(w)} = \frac{1}{2n}\sum_{i=0}^{n}((w_0 + w_1\mathbf{x_{i1}} + ...w_p\mathbf{x_{ip}}) - y_i)^2, \tag{16}$$

where $\mathbf{y} = (y_1, y_2, ...y_n)^T$ are the true target values.

Sensitivity to outliers in the data is a downside of this method. Ridge regression improves upon this as it aims to fit a linear model using the following cost function:

$$\mathbf{j(w)} = \frac{1}{2n}\sum_{i=0}^{n}((w_0 + w_1\mathbf{x_{i1}} + ...w_p\mathbf{x_{ip}}) - y_i)^2 + \frac{\alpha}{2n}\sum_{j=0}^{n} w_j^2. \tag{17}$$

Eq. 17 consists of the squared distance and a regularization parameter which penalises large values of the weights. α is a hyperparameter that can be tuned to improve the fit. A different regularisation scheme is used by LASSO regression with the cost function

$$\mathbf{j(w)} = \frac{1}{2n}\sum_{i=0}^{n}((w_0 + w_1\mathbf{x_{i1}} + ...w_p\mathbf{x_{ip}}) - y_i)^2 + \frac{\alpha}{2n}\sum_{j=0}^{n} |w_j|. \tag{18}$$

4.2 Non-linear Models

Examples of non-linear models are Gaussian process regression (Gaussian regression) and kernel ridge regression. The Gaussian process regression algorithm is similar to linear regression but uses Gaussian kernel functions to improve the model's ability to express non-linear functions. In this model, it is assumed that the target values $\mathbf{y}$ have the form $\mathbf{y} = \mathbf{f}(\mathbf{x}) + \mathbf{e}$ where $\mathbf{e}$ is independent noise. $\mathbf{f}(\mathbf{x}) = (\mathbf{f}(\mathbf{x_1}), \mathbf{f}(\mathbf{x_2}), ...\mathbf{f}(\mathbf{x_n}))^T$ is called the kernel function. The probability of observing y is assumed to have the normal (Gaussian) distribution:

$$p(y, \mathbf{f}(\mathbf{x})) = N(y, \mathbf{f}(\mathbf{x}), \sigma). \tag{19}$$

The kernel function used for this problem is

$$K(x, y) = e^{-\gamma|x-y|^2}. \tag{20}$$

An alternative method for fitting a kernel function is the kernel ridge regression method. This method employs a similar loss function to ridge regression (see Eq. 17), but instead of fitting a linear function, it produces a fit for a given kernel function. Two hyperparameters can be tuned when using the kernel function from Eq. 20, i.e., the regularisation strength α and the coefficient γ. The choice of these hyperparameters is provided in the Appendix.

4.3 Decision Trees

Unlike Gaussian and kernel ridge regressions, tree based algorithms perform discrete partitioning of the feature space. One of the machine learning methods I have frequently used is decision trees [**Breiman**; **1984**][**Quinlan**; **1993**]. This supervised learning method is applicable to regression and classification problems. Decision trees are built by learning decision rules from specific data features to construct a tree diagram.

Decision trees follow an algorithm that aims to learn a set of if-then-else decision rules to split the data accordingly. For instance, the algorithm may split the data according to whether the Fermi energy is larger than 7.0 eV. Based on the answer, the data is split into various categories and presented in a tree graph with nodes. Each node may contain a question and has branches that lead to other nodes. The final output of a decision tree is a leaf node which does not have any further branches. Figure 3 provides on the left an example of a decision tree. Another way to visualise this algorithm is by using a multidimensional feature space where each dimension corresponds to a feature of the input data and the nodes (here, node is used to refer to a basic unit of a tree) partition this space is some way. In the example in Fig. 3, these features are labelled as X_1 and X_2. The feature space is partitioned by each node of a decision tree, as demonstrated on the right-hand side of Fig. 3.

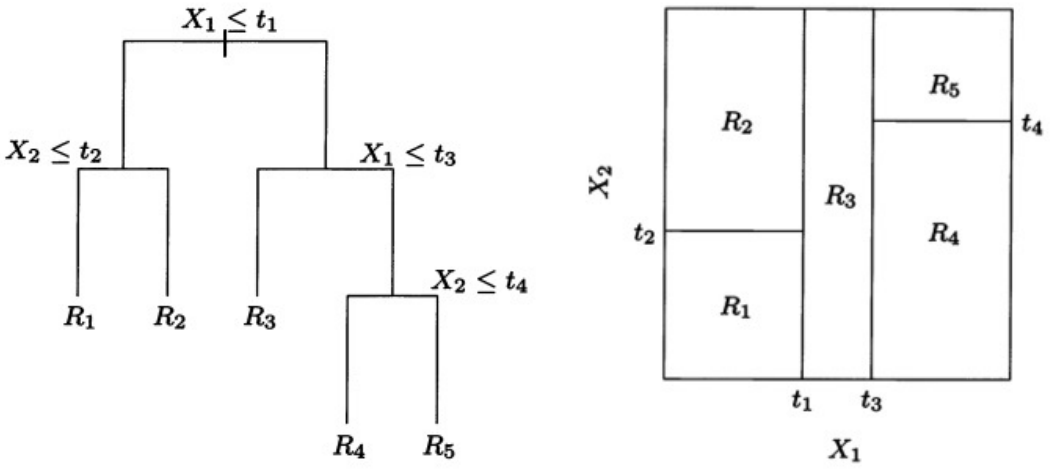

Figure 3: Left: Schematic representation of a decision tree. Right: the way the tree on the left partitions the feature space. Figure taken from Ref. [**Hastie**; **2001**].

The algorithm's performance can be improved through hyper-tuning parameters like the tree's maximum depth, minimum number of leaves, and required number of samples for node splitting.

4.4 Random Forest

The random forest algorithm trains multiple decision trees. For each tree, the algorithm selects subsets of features randomly at each split to ensure high diversity between different decision trees. In case of a continuous output, where a regression problem is being addressed, the prediction of the random forest is made by averaging the results from each tree in the forest to compute the final outcome. For classification problems, the algorithm uses a majority vote to produce a discrete output. The final output is determined by the most frequent output generated by the trees.

The algorithm's performance can be improved by hyper-tuning parameters such as the number of random subsets, the number of features in each subset, or the parameters of the trees. In my program, I chose the number of random trees to be 50 and the maximum depth of the trees to be 10. Further information about this choice is provided in the Appendix.

One benefit of a random forest model over decision trees is the reduced chance of overfitting. One downside is that this algorithm is more time-consuming to train. The algorithm combines different decision trees and this way increases the accuracy of the model and lowers the risk of overfitting.

4.5 Boosted Trees

Gradient-boosting tree models, like the random forest, utilise an ensemble of trees for predictions. However, unlike the random forest, each tree in this model aims to enhance the performance of the preceding one through a gradient-boosting method. This method can be broken down as follows: An initial tree is constructed, and the discrepancy between its forecasts and the actual values is assessed. Subsequently, another model is trained to predict the first model's errors. The subsequent model is then generated as a combination of the previous tree and the differences. Adjustable hyperparameters of boosted trees include the number of trees and the maximum tree

depth. In my program, I selected the amount of trees to be 50 and set the maximum tree depth to 10. Further details concerning these choices are provided in the Appendix.

4.6 Evaluation of Model Accuracy

To assess the effectiveness of the used machine-learning models, various different metrics can be used, such as: the mean absolute error (MAE), the maximum absolute error (Max AE), the root mean square error (RMSE), and the R^2 score s, which is a measure of how well the model fits the data and is defined by

$$s = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}. \tag{21}$$

In this equation, y_i represent the true expected values and f_i the predicted values, while $\bar{y}$ denotes the mean values of y_i. In case of a perfect fit, the score would be $s = 1$. A model that consistently predicts the average true value will obtain a score of $s = 0$, whereas models that perform worse than that will receive a negative score.

RMSE and MAE provide an overview of the difference between predicted and true values. RMSE penalises larger errors and offers a more comprehensive evaluation of the model's performance. Max AE displays the upper limit of a model's error. All three of these values should be minimised as much as possible, as the greater their values, the poorer the model is. Conversely, an ideal prediction would achieve an R^2 score of 1. A model that predicts the average output value, regardless of input, would achieve a score of 0. Any prediction that performs worse than this would result in a negative score.

Another distinction between the errors and the R^2 score is that the assessment of whether a particular error (such as RMSE or MAE) corresponds to the desired accuracy relies on the absolute values of the predictions. However, the assessment of whether a certain value of score is good does not. In general, a score above 0.9 is deemed desirable.

4.7 Feature Importances

While employing decision trees, random forest, and boosted trees, one can determine the significance of each setting in deciding the result of the machine learning algorithm. Such an evaluation can aid in identifying the crucial feature required for generating target values.

Numerous approaches exist for calculating feature importances. One approach to ranking feature importance is through the Gini importance [**Menze**; **2009**], which looks at how often a feature is used in a decision tree split and how much that split contributes to improving the target metric. Another method is permutation-based feature importance [**Breiman**; **2001**], which is computed as follows: firstly, the original R^2 score s is calculated. Subsequently, for a given feature, the values are randomly shuffled, and the score s_k of this altered data is calculated. This is repeated K number of times, with K being set to 5 in my program. The resulting importance is calculated using the formula:

$$i = s - \frac{1}{K} \sum_{k=1}^{K} s_k \tag{22}$$

Permutation importance is used to emphasize the impact of disregarding a specific feature on the accuracy of predictions, and how it would affect the similarity of corresponding fingerprints to the most converged fingerprint. The Gini importance can be inflated for features that could overfit the model even if they do not predict the output. Thus, in our case, permutation importance can be more informative in determining the significance of the given feature in the convergence of the DOS. [**Breiman**; **2001**]

5 Dataset

The dataset that I use contains DOS calculations of 69 different materials that have been generated using FHI-aims [**Carbogno**; **2022**]. FHI-aims [**Blum**; **2009**] is a software package implementing DFT using numeric atomic orbitals (NAOs) as the basis set. For each material, computational parameters were varied to study their effect on the numerical precision. Since I focus primarily on binaries in this dataset, I distinguish between the two species in the binary by the electronegativity. I label the species with the lower (higher) electronegativity by the index 1 (2). This is done to make sure that features are independent of the order of elements in the dataset.

Also the accuracy of approximations is evaluated. The features I investigate include:

- Exchange-correlation (XC) functional (see Sec. 2). The dataset contains calculations using the LDA functional, and the GGA functional in the PBE parameterization.
- k-point density: The author of the dataset used a k-point density of 2, 4, and 8 for sampling the first Brillouin zone of the reciprocal space.
- Numerical settings: The FHI-aims code provides pre-defined sets of numerical settings for adjusting the numerical precision of a calculation. These are also referred to as the representation of atomic species. These settings are available as tabular data for each atomic species and range from "light", meaning least precise, over "tight", to "really tight", which refers to the most precise settings. The choice of the numerical settings will define several parameters. The confinement potential is a potential that is zero until a certain radius and then ramps up to infinity at another radius value. It works to force the charge density in certain atomic centered basis functions to zero for large radii away from the atom to help speed up the calculation. Using no confinement potential will give the most precise calculation. The confinement potential offset and width are set by the numerical settings which determine where the potential starts to take effect (offset) and the $Deltar$ it needs to ramp up to infinity. The "basis dependency cutoff" adjusts the confinement potential for each basis function separately when used and is defined for each basis function. The maximum number of angular grid points per radial integration shell, and the highest angular momentum component used in the multipole expansion of the charge density are also set by the numerical settings. The light/tight and really tight presets are only suggestions and users can modify the precision by changing the respective values in the input file. In the present dataset, the author of the data used the light/tight and really tight settings.
- Basis set: To solve the Kohn-Sham equation, the wave functions are expressed as linear combinations of basis functions. These are NAOs centered at the nuclei (see also representation of atomic species above). Therefore each calculation of a binary material contains two different basis sets, one for each atom species. The basis set size can be set to be "minimal", "standard", "tier1", "tier2", "tier3" and "tier4" in FHI-aims, with "tier4" being the most precise setting. This value determines how many basis functions to include for each atom species. The "standard" basis set refers to the default basis set size when using a specific numerical setting.
- Unit-cell volume: The dataset contains calculations that were performed with the geometry found in the Springer handbook of materials. The geometries were also increased along the direciton of each lattice vector by 5% of the lattice vector magnitude. I represent the unit-cell expansion with by using the unit-cell volume as a feature.
- The relativistic treatment in terms of the zero-order regular approximation (ZORA) and the atomic ZORA, as defined in Ref. [**Blum**; **2009**] in Eqs. (53) and (55) respectively.
- Iteration limit: It represents the maximum number of iterations for the self-consistent loop of solving the Kohn-Sham equations.

For each calculation in the dataset, I compile a list of features referred to as settings, which differentiate the various calculations from one another. These settings comprise the parameters listed above. In addition, I use the electronegativity of the elements [**Connelly**; **2005**]. Instead of the k-point density, I use the number of k-points along the k-grid. This feature has three components along the reciprocal lattice vectors, referred to as K-grid x, y, and z.

Non-numerical parameters were encoded as numerical values. The PBE XC functional is represented by 1, LDA by 0. ZORA was assigned 1 and atomic ZORA was assigned 0. I represent the numerical settings by the values of the *cutoff potential offset*, *cutoff potential width* and *basis depth cutoff*. The set of basis functions of each element was represented by a one dimensional vector of 45 items, with each position in the list corresponding to a particular basis function. In total the dataset had 45 different NAOs were used, hence the use of a vector of length 45. For example, in the first position is the 1s orbital function. The quantum number was varied between 1 to 8 and the orbital type was varied between s, p, d, f, g, h. If the basis function occurred in the basis set, the corresponding entry was set to 0, otherwise 1. Additionally, I represent the basis set size with an additional feature, which is the number of basis functions in the basis set for this species. For each binary I distinguish them, as mentioned above, with the 1 and 2 suffixes based on electronegativy. I refer to this feature as the basis size 1 and 2.

The most precise calculation is referred to as the *reference calculation* of each material and for each unit-cell volume. It is identified as the calculation with the largest basis set size, highest numerical settings, and using the PBE functional. I then compared the DOS for each material and volume with the respective reference. The comparison was conducted by calculating the similarity, as outlined in Section 3.

6 Results

In the following, I report how machine learning (ML) models were used to predict the convergence of DOS calculations. The goal was to predict the level of convergence of an arbitrary calculation for a given material, represented by the similarity of its DOS to the DOS of the reference calculation. In order to achieve this, my first aim was to determine the most effective machine learning method for the given problem. This was done by systematically comparing the model performances for a single material, namely MoC. The results of this analysis are presented in Section 6.1. In Section 6.2, the ML methods that performed best on MoC were trained and tested on each of the remaining 68 materials. Subsequently, the feature importance was analysed to identify the settings with greater influence on the ML model. In Section 6.3, I expand the scope of the learning task by trying to learn the level of convergence independent of the material, by training on the full dataset. In the subsequent analysis, I divided the dataset into several subsets and analysed which ones the ML performed better on. Finally, I re-examined the feature importance.

6.1 Performance of ML Algorithms on the Example of MoC

In order to study the performance of different ML algorithms independently of the effects of the selected data, I selected a single material to systematically evaluate the model performance. I chose MoC because the calculations for the full set of parameters are available on NOMAD. Figure 4 shows the crystal structure of MoC, with lattice vectors and angles indicated. MoC has a hexagonal unit cell with space group number 187 containing two atoms. It has equilibrium lattice constants of 2.9Å in the $\vec{a}$ and $\vec{b}$ directions, and 2.83Å in the $\vec{c}$ direction. The corresponding cell volume is 20.6Å^3.

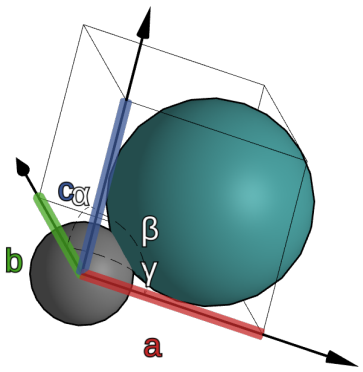

Figure 4: Crystal structure of MoC. Carbon is in grey, molybdenum in green; a, b and c are the lattice constants, α, β, and γ indicate angles between lattice vectors. Source: [**Carbogno**; **NOMAD**].

For each of the 288 calculations in the dataset, I assessed the similarities between the respective fingerprint and the fingerprint of the reference calculation. Then, I applied different machine-learning algorithms to predict similarities for given settings. Specifically, I trained machine learning models $\hat{y} = f(X)$, where X represents the settings and $\hat{y}$ represents the similarity between the fingerprint corresponding to the respective settings and the most accurate fingerprint. To achieve this, I randomly divided the dataset into two subsets, i.e. training and test set, with a ratio of 7:3.

I start the analysis with linear models, which are conceptually simple, but generally require a linear relationship between the input data and the learning target. Here, I present the results for three different approaches: least squares, ridge, and LASSO regressions. The predicted similarities resulting from these models are compared to the true values in the upper left of the figure 5. The model metrics are shown in Table 1. The score, RMSE, MAE and maximum absolute error can vary depending on how the training and test sets are split in the main dataset. For each method, I computed the mean values from 50 distinct random splits and estimated the uncertainties using standard deviations. Some correlation can be observed; however, the predictions are not entirely accurate. This is to be expected, given the complex learning task. However, the apparent correlation suggests that the predictions can be improved with non-linear ML models.

To include non-linearities in the models, I used kernel ridge regression with a Gaussian kernel (see Eq. 20), and Gaussian regression. The resulting accuracy is presented in Table 1. Both methods achieve similar performance, which is expected because of their similar approaches to modelling the data: in both cases, Gaussian distributions are used. In both cases, the test error is significantly smaller than for the linear models.

Finally, I evaluated three different, decision-tree based models: decision tree regressors, as well as two ensemble methods, the random forest algorithm, and boosted trees. The parity plot bottom right panel of Fig. 5 presents the comparison of true and predicted similarities for the decision tree model. The accuracies of all three tree-based methods are presented in Table 1. The tree based methods show similar performances, with random forests showing the lowest RMSE, reaching the level of kernel ridge regression and Gaussian process regression. The boosted tree method appears to be less suitable for predictions on a single material.

Method	RMSE	R^2 score
Least squares regression	0.030 ± 0.003	0.836 ± 0.036
Ridge regression	0.069 ± 0.156	-4.465 ± 25.983
LASSO regression	0.030 ± 0.003	0.837 ± 0.034
Gaussian regression	0.014 ± 0.003	0.964 ± 0.015
Kernel ridge regression	0.014 ± 0.004	0.965 ± 0.018
Decision trees	0.015 ± 0.007	0.953 ± 0.045
Random forest	0.014 ± 0.005	0.961 ± 0.032
Boosted trees	0.020 ± 0.003	0.925 ± 0.026

Table 1: Scores and the correlation coefficients between the predicted and the true similarities for different supervised learning methods for the example of MoC.

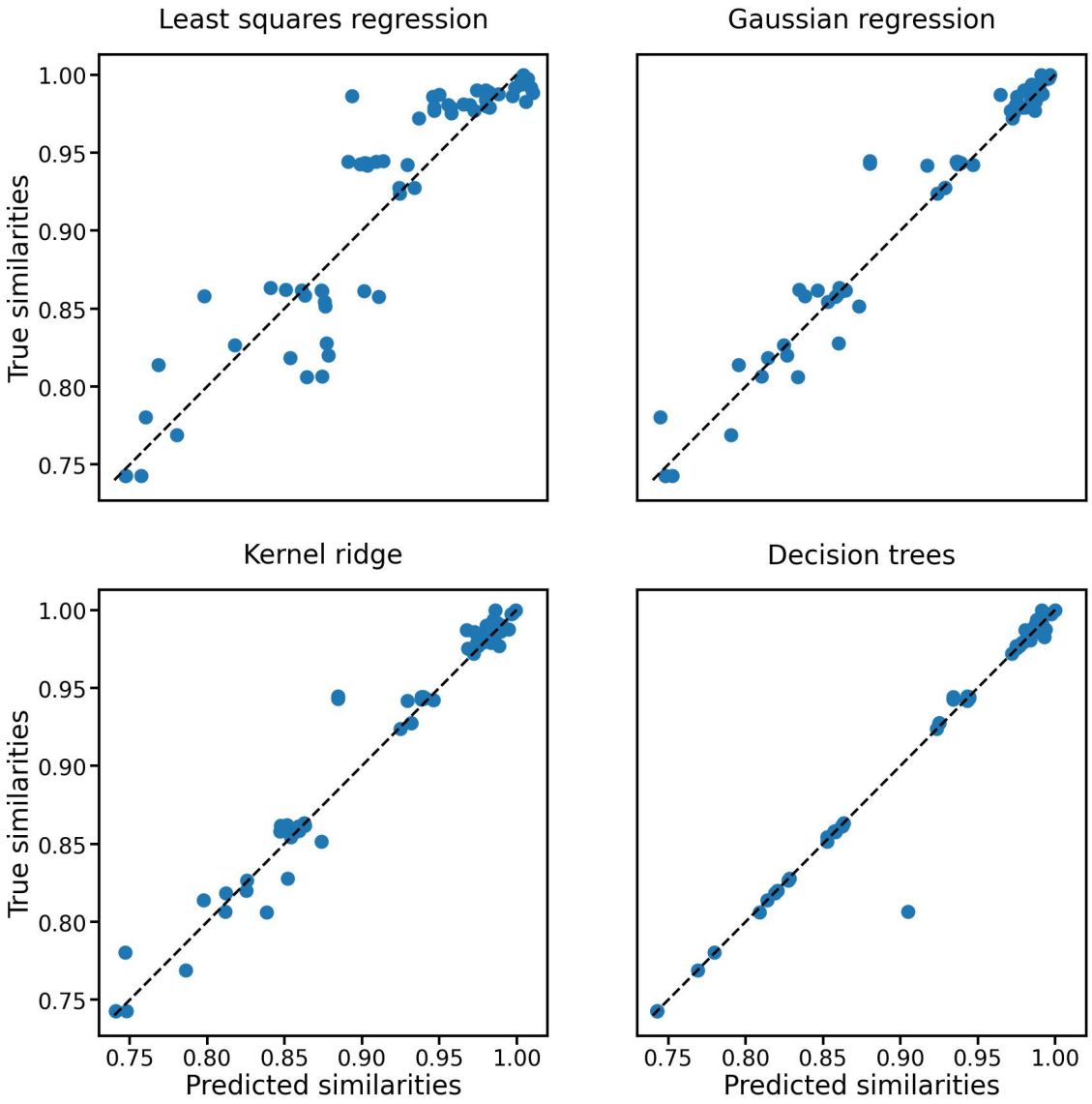

Figure 5: Results of different machine-learning methods for MoC. The x axes depict the predicted similarities, the y axes the true similarities. The dashed line indicates where true values and predictions are equal.

Comparing all employed methods, I find that neither of the linear methods is flexible enough to provide good predictions, as indicated by the comparatively high RMSE. The non-linear methods show comparable errors, with the exception of boosted trees, which show significantly worse performance.

6.2 Applicability to Different Materials

To test the generalisability of the approach, I repeated the analysis described in the previous section on the datasets of all 69 binary alloys (see Sections 5). Model training and hyper-parameter optimisation were performed in a high-throughput manner, automatised for all materials. For this task, based on the results above, I use only non-linear models. Table 2 presents a comparison of Gaussian and kernel ridge regression, decision tree, random forest and boosted trees models in terms of their RMSE, maximum absolute error, MAE and R^2 score, averaged over all 69 materials. Furthermore, I calculated the standard deviation over all materials for each model. The table reveals a clear disadvantage for Gaussian and kernel ridge regression, therefore I put the focus on the tree based models. Although the RMSE remains similar for them, the Max AE is highest for decision trees and lowest for random forests. I note that the standard deviations for all models were very high, larger than the average in most cases. This indicates a significant variation in model performance depending on the material.

Based on Table 2, the most promising model architectures are ensemble models, belonging to the random forest and boosted trees categories. This aligns with my expectations since decision tree based models are able to deal with heterogeneous data (i.e. a mix of discrete/continuous data) well since they partition the feature space into discrete regions. Furthermore, ensemble models are known to be more robust than individual decision trees, which is well represented in the Max AE values in the table.

Method	RMSE	Max AE	MAE	R^2 Score
Gaussian regression	0.056 ± 0.052	0.189 ± 0.172	0.038 ± 0.039	0.834 ± 0.186
Kernel ridge	0.054 ± 0.058	0.203 ± 0.231	0.035 ± 0.039	0.857 ± 0.196
Decision trees	0.039 ± 0.046	0.194 ± 0.197	0.014 ± 0.021	0.912 ± 0.129
Random forest	0.036 ± 0.037	0.151 ± 0.139	0.019 ± 0.023	0.932 ± 0.083
Boosted trees	0.035 ± 0.041	0.165 ± 0.175	0.016 ± 0.020	0.932 ± 0.099

Table 2: Scores and errors of predicted similarities of the DOS fingerprints with respect to the most converged fingerprint for different supervised learning methods.

Now, I examine the permutation feature importance of the settings. High (low) feature importance indicates that a feature is important (not important) for the the performance of the model. I therefore computed the feature importance for each feature and material. The minimum values for each feature are close to 0, indicating that every feature is insignificant for at least one material. Sometimes, the importance of a feature was less than zero, indicating that the accuracy of shuffled data surpassed that of the original data. If the importance is negative with a small absolute value, it implies that the feature is not relevant to the prediction, but the shuffled data happened to be more precise by chance. I also note that some features are correlated with each other, so the resulting importance for each feature is lower than it would be in case of uncorrelated features. Figure 6 presents the maximal importance of each feature across all materials. I note that the figure does not show the basis functions feature, which is composed of 90 components. This was done to increase the readability of the plot.

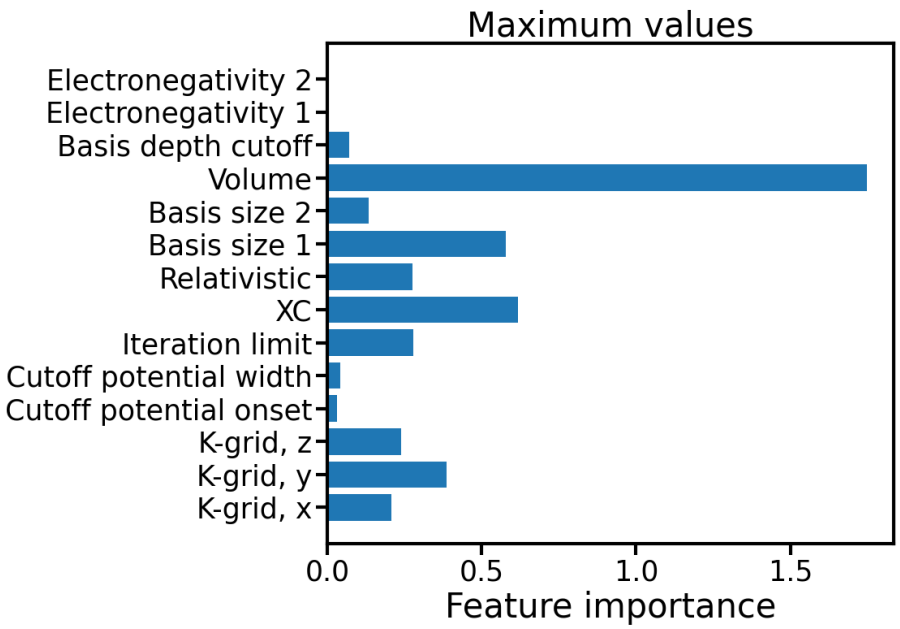

Figure 6: Maximum values of permutation feature importance across all materials for the random forest model.

The maximal features importance of the electronegativities are 0. This is expected, as every model is trained on different calculations for the same material, therefore the electronegativities are identical for all data points. The importance values for the other features varied significantly across the models for different materials, therefore no general statement can be made. However, features that have low importance for several materials can be identified. These include the basis dependent cutoff, which was insignificant for 53 materials, the cutoff potential onset, which was insignificant for 39 materials, and the cutoff potential width, which was unimportant for 35 materials. This finding is consistent with the information the the FHI-aims manual, which states that the settings of the potential cutoff have a small impact on the results of a calculation. Therefore its predictive power is low in models that describes the convergence of the DOS. The relativistic approximation was unimportant for 3 materials, the basis size 1 was insignificant for 5 materials, and the basis size 2 was insignificant for 4 materials. The iteration limit was irrelevant for 4 materials, and the K-grid was irrelevant for one material.

Considering the most important features for different materials, I found that for 48 materials, the exchange-correlation (XC) functional was most important, while the volume was most important for 9 materials. The K-grid along x direction had the highest importance among the computational parameters, turning up 5 times, followed by the K-grid along y direction, which turned up 4 times. The basis size 1 and 2 turned up once among the features with the highest importance.

Figure 6 shows that the volume has the highest feature importance, however, as described above, it is the most important feature only for 9 materials. This is likely a consequence of the fact that the feature represents the expansion of the unit cell. This expansion has a large effect on the electronic structure of some materials. Therefore, it is not surprising that for some materials it is very important.

6.3 Generalisation to the Full Dataset

After training individual models for each material, I investigated the transferability of my approach by predicting the convergence of a DOS spectrum of a material with models trained on data from the the other 68 materials. To differentiate between materials, I extended the features to include the DOS fingerprint of the calculation with the lowest numerical settings for the respective material. For the ML models, I used the random forest and boosted tree algorithms.

The models were trained on 68 materials, while the test set consisted of the calculations for the respective remaining material. In total, I trained each ML model 69 times, with a different material selected for the test set each time. The outcome of each model was then employed to assess the accuracy of a DOS calculation without knowing the most converged calculation. This approach reduces the computational complexity associated with convergence tests, while also measuring the precision of a calculation.

Table 3 presents a comparison of the MAE, (Max AE), the RMSE and the R^2 score of the mentioned machine learning models across all 69 materials, along with their standard deviations. Additionally, Fig. 7 displays a histogram of the RMSE values.

Method	RMSE	Max AE	MAE	Score
Random forest	0.107 ± 0.080	0.273 ± 0.153	0.084 ± 0.069	0.437 ± 0.517
Boosted trees	0.116 ± 0.089	0.294 ± 0.186	0.091 ± 0.074	0.303 ± 0.765

Table 3: Scores and errors of predicted similarities for different supervised learning methods.

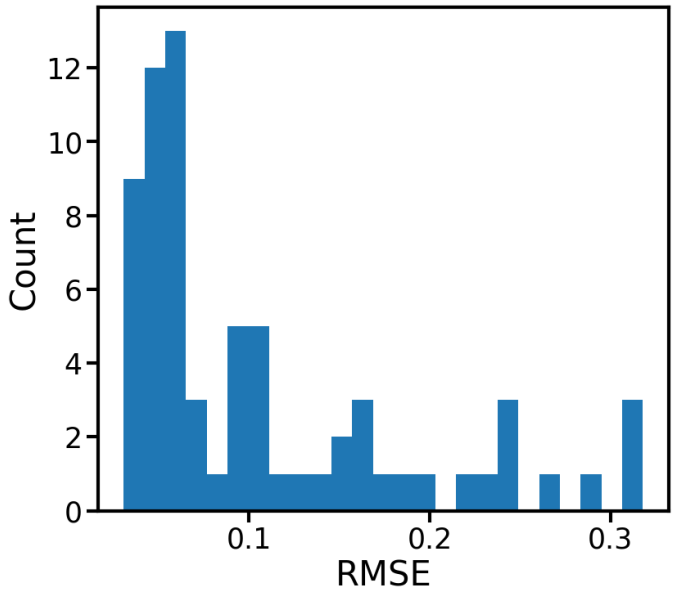

Figure 7: Histogram of RMSEs for the random forest model, representing the model performance on each of the 69 materials individually.

The vast majority of RMSEs are below 0.1, with the densest region being closest to 0. However, in 3 cases, the RMSE is above 0.3, indicating that, while random forests perform relatively well for certain materials, this is not the case for others. Based on the information presented in Table 3, it is clear that the overall accuracy of these models is very low due to the low scores and relatively high RMSE, with random forests performing marginally better than boosted trees. To enhance

the predictive power, I proceeded with dividing the materials into subsets and use separate ML models for each subset.

The convergence of the DOS is influenced by whether the substance is a metal or a semiconductor. The dataset contains 38 semiconductors and 31 metals. When fitting the metals separately using a random forest model, the RMSE value is 0.084 ± 0.054, while it is 0.142 ± 0.085 for semiconductors. Both of these values are inferior to the mean RMSE values for metals (0.071 ± 0.045) and semiconductors (0.137 ± 0.089) when the random forest algorithm was trained on the full dataset. Thus, the model performed best on metals when trained on the full dataset. The contrast in performance between metals and semiconductors is illustrated in Fig. 8, showing a histogram of the RMSE for metals (in blue) and semiconductors (in orange).

The findings demonstrate that the errors in predicting the DOS for semiconductors are significantly larger than those for metals, necessitating further categorisation of semiconductors. One characteristic worth examining is the space group. The semiconductors in the dataset exhibit the following space groups: 216, 225, 176, 221, 62, 160, 224, 186 and 194, where space groups 216 and 225 are the only ones with more than two representatives. I split the dataset into two subsets and trained one model on all semiconductors with space group 216, and one with all semiconductors with space group 225.

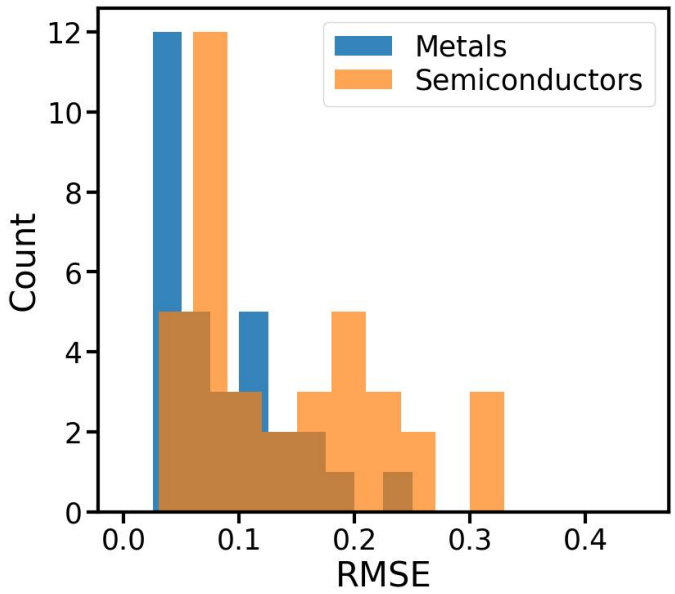

Figure 8: Histogram of RMSE for metals (blue) and semiconductors (orange).

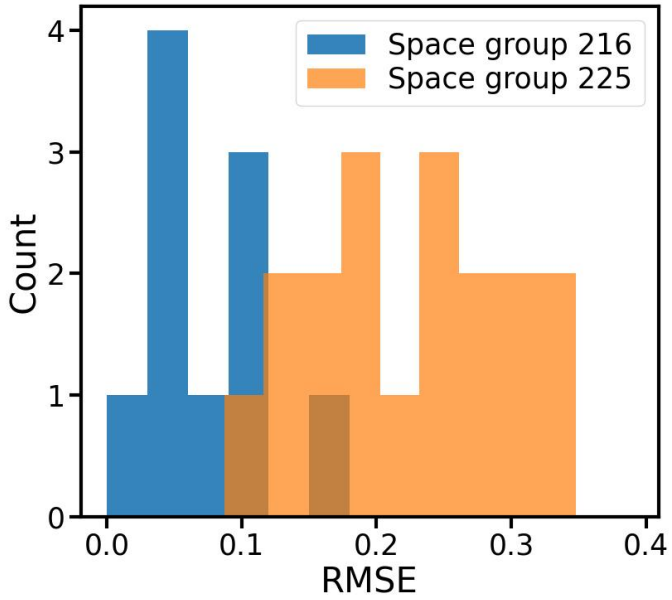

Figure 9: Histogram of RMSÉ for semiconductors of space group 216 (blue) and 225 (orange).

Fig. 9 displays the RMSE of space group 216 (in blue) and space group 225 (in orange). The graph indicates a notable contrast in prediction accuracy between the two space groups. With an average RMSE of 0.072 ± 0.041, space group 216 considerably outperforms space group 225, which as an average RMSE of 0.224 ± 0.074. Both of these values are close to the average root mean square error (RMSE) obtained when training a random forest on the entire dataset: 0.072 ± 0.035 and 0.191 ± 0.094, respectively.

For the boosted tree model I see similar behaviour to the random forest model. For the metals the RMSE is 0.075 ± 0.052, for all semiconductors the RMSE is 0.151 ± 0.099, for semiconductors of space group 216 RMSE is 0.075 ± 0.032 and for semiconductors of space group 225 RMSE is 0.218 ± 0.101. This shows that the boosted tree model performs better on metals than on semicondctors. Moreover, it performs better on semiconductors of space group 225 than of space group 216.

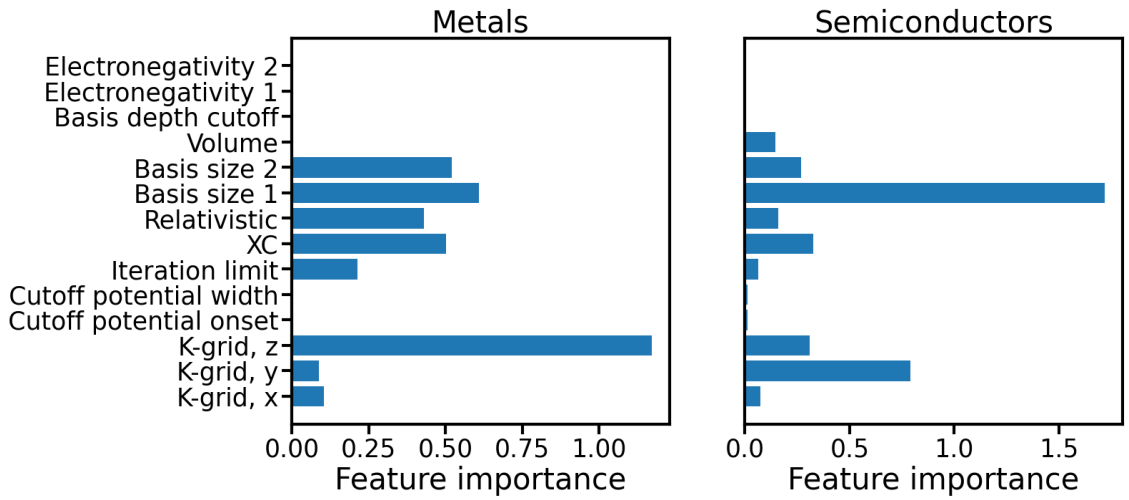

Figure 10: Maximum values of feature importance as a result of random forest models for metals (left) and semiconductors (right).

Finally, I examine the feature importances of the random forest for both metals and semiconductors, as illustrated in Fig. 10. The importance of the electronegativity was consistently assigned a value of 0 for both atomic species, indicating that the random forest does not take into account the composition of the material. Instead, different materials are distinguished using other indicators. In 23 out of 31 cases, the K-grid had the greatest importance for metals, while for semiconductors, the number of basis functions for the first element was most important in 33 out of 38 cases. This contrasts with the importances for individual materials, where, for instance, the exchange correlation functional and the volume were typically more significant.

7 Discussion and Conclusions

To determine the degree of similarity between the electronic structure of a particular DFT calculation and that of the most precise DFT calculation, one can use the DOS fingerprints as defined in Section 3. These similarities can be predicted by machine learning algorithms based on the numerical settings employed. For this purpose, I have found tree-based algorithms (decision trees, random forest and boosted trees) to be more effective than other algorithms, such as least squares, ridge, LASSO, kernel ridge or Gaussian regression. Tree based algorithms are known to deal well with mixed integer/floating point data. This is in line with the expectations because tree-based algorithms perform discrete feature partitions and the given features (i.e. settings) are composed of discrete numbers, rather than continuous parameters.

When using decision trees, random forest and boosted trees for each material individually (i.e. the data for the training and test sets were all of the same material), these algorithms demonstrated strong performance. For the decision trees, the fit was good, as shown by the average R^2 score, being 0.912 ± 0.129 and the average RMSE between the predicted and true values being 0.039 ± 0.046. For the random forest model, the fit was very good, as shown by the R^2 score being 0.932 ± 0.083 and the RMSE being 0.036 ± 0.037. For the boosted trees, the fit was also very good as shown by the R^2 score being 0.932 ± 0.099 and the RMSE being 0.035 ± 0.041.

The model's performance was, however, suboptimal when trained and tested on all 69 materials in the dataset. I found the accuracy of predictions to be material-dependent, which is a common problem in materials science. Additionally, mixing the data of materials with equilibrium and expanded unit cell volumes may had a strong influence on the model performance. On

one hand, this increases data set size, on the other hand, this may make the modelling task more challenging, since the model needs to include the effects of volume expansion on the DOS. As was shown in Fig. 6, the volume expansion is used heavily by some single material models. This can be one reason why the multi-material model performance is lower.

The random forest model shows a better performance for metals (RMSE = 0.071 ± 0.045) than for semiconductors (RMSE = 0.137 ± 0.089). A possible explanation for this is that the DOS convergence can vary depending on the band gap, which is obviously 0 for metals but has a wide range for semiconductors. I note that with 69 materials I have only a small sample of the possible material space and not all types of materials are equally represented in the training dataset.

I have observed that the model also depends on the crystal structure. For example, it performed better for semiconductors of the space group 216 than for those of space group 225. One possible explanation is that among the materials with space group 225, there is a significant number of semiconductors with large band gaps (> 3 eV), as well as semiconductors with small band gaps. This variety may cause these data points to behave differently from each other despite sharing electronic and geometrical properties. The same can be said for the boosted trees model, which performs better for metals than semiconductors and exhibits the same behaviour as the random forest model for the semiconductors with respect to the two space groups. This further suggests that despite my best modelling efforts, I may be working with too few data to model the entire material space.

It can be inferred that training and testing the model on specific subsets of materials does not necessarily result in significantly better performance, as shown for the case of being trained and tested only on metals or only on semiconductors. This result implies that extending the training data with more materials, even if these do not belong to the same material class, can lead to better model performance.

In addition to the predictions, also the feature importance for the tree based machine learning models was analysed. For model that are trained on one material only, the XC functional has the highest feature importance. For models trained on different materials, I found that for metals the k-grid affected the convergence the most. For semiconductors, it was the number of basis functions of the less electronegative element.

Overall, the best-performing models were boosted trees and random forests. These models produced more accurate predictions for metals than for semiconductors, as well as being more accurate for semiconductors with space group 216 than 225. Training the models on the entire dataset was more effective than training on the subsets that the target materials belong to.

This work could be extended by utilizing other machine learning models such as neural networks. For the purpose of grouping materials with similar behaviour, unsupervised machine learning methods could be used. Also other material properties could be into account. Most importantly, many more materials should be used to go beyond the limiting exploration of binary materials. For that, however, more high quality data are needed.

Appendix

Selection of Hyperparameters

I have adjusted the maximum depth of the tree in the decision trees, the maximum depth of trees for the random forests and the number of trees used in boosted trees. However, for all three models, altering these hyperparameters appeared to have only negligible effect on the prediction. Fig. 11 illustrates the impact of varying hyperparameters on the predictive performance of four distinct models, measured by the RMSE for C_4 Ta_4, CMo, Cd_4 O_4 and S_4Sn_4.

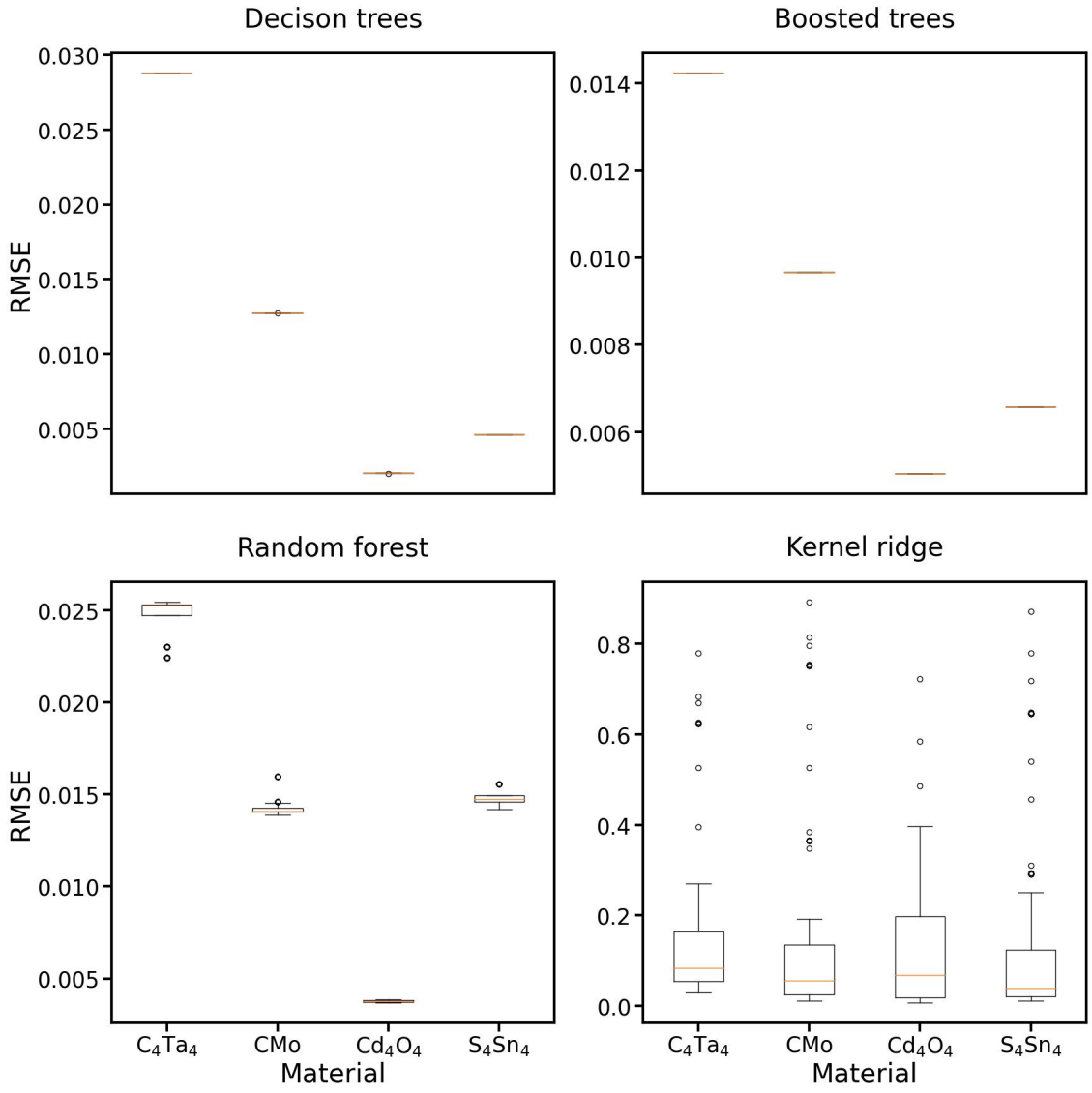

Figure 11: Box plots for how the RMSE of model predictions changes upon changing the hyperparameters for four different materials. The upper left (right) plot shows the RMSE for decision trees (boosted trees), the lower left (right) shows the RMSE for a random forest (kernel ridge regression.

Based on these findings, I selected the hyperparameters requiring minimal computational

time, i.e., a maximum depth of 10 and 50 trees. Additionally, I optimised the hyperparameters for kernel ridge regression, including the regularisation strength α and the γ coefficient of the kernel function (Eq. 20). In this case, altering these parameters had a significant impact on the outcome. Figure 11 demonstrates how changing the hyperparameters affects the RMSE of the prediction for four distinct materials. The most common hyperparameters ($\alpha = 10^{-5}$ and $\gamma = 10^{-3}$) were employed.

References

[Carbogno, 2022] Carbogno, C., Thygesen, K.S., Bieniek, B. et al. Numerical quality control for DFT-based materials databases. *npj Comput Mater* 8, 69 (2022). https://doi.org/10.1038/s41524-022-00744-4

[Speckhard, 2023] Speckhard, D. T., Carbogno, C., Ghiringhelli, L., Lubeck, S., Scheffler, M., Draxl, C. (2023). Extrapolation to complete basis-set limit in density-functional theory by quantile random-forest models. *arXiv preprint* arXiv:2303.14760. https://arxiv.org/pdf/2303.14760.pdf

[NOMAD] NOMAD. Available at: https://nomad-lab.eu/nomad-lab/ (Accessed: 12 January 2024).

[Hohenberg, Kohn, 1964] P. Hohenberg and W. Kohn. "Inhomogeneous electron gas" *Physical Review* 136(3B):B 864 – B 871, (1964. 4.1).

[Kohn, Scham, 1965] W.Kohn and L.J.Sham. "Self-consistent equations including exchange and correlation effects." *Phys. Rev.* 140:A1133, 4.1 (1965).

[Perdew; 2001] John P. Perdew; Karla Schmidt "Jacob's ladder of density functional approximations for the exchange-correlation energy" *AIP Conf. Proc.* 577, 1–20 (2001). https://doi.org/10.1063/1.1390175

[Blum, 2009] Volker Blum, Ralf Gehrke, Felix Hanke, Paula Havu, Ville Havu, Xinguo Ren, Karsten Reuter, and Matthias Scheffler, Ab initio molecular simulations with numeric atom-centered orbitals. *Computer Physics Communications* 180, 2175-2196 (2009). http://dx.doi.org/10.1016/j.cpc.2009.06.022

[Kuban, 2022] Martin Kuban, Santiago Rigamonti, Markus Scheidgen, Claudia Draxl. "Density-of-states similarity descriptor for unsupervised learning from materials data" *Scientific Data* 9, 646 (2022). https://doi.org/10.1038/s41597-022-01754-z

[Tanimoto, 1958] Tanimoto, T. T. (1958). "An elementary mathematical theory of classification and prediction by T.T. Tanimoto". New York, : International Business Machines Corporation

[El Bouchefry, 2020] Khadija El Bouchefry PhD, Rafael S. de Souza PhD, Chapter 12 - Learning in Big Data: Introduction to Machine Learning, 225-249 (2020). https://doi.org/10.1016/B978-0-12-819154-5.00023-0

[Breiman, 1984] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA (1984).

[Quinlan, 1993] J.R. Quinlan. C4. 5: programs for machine learning. Morgan Kaufmann (1993).

[Carbogno, NOMAD] Carbogno, C. NOMAD. https://nomad-lab.eu/prod/v1/gui/search/entries/entry/id/ztTuiBKMMwOlHs_HGdmrWdSuVrtC (Accessed: 12 January 2024).

[Hastie, 2001] Trevor Hastie , Jerome Friedman , Robert Tibshirani. "The Elements of Statistical Learning" p. 266 (2001)

[Menze, 2009] Menze, B.H., Kelm, B.M., Masuch, R. et al. A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC Bioinformatics* 10, 213 (2009). https://doi.org/10.1186/1471-2105-10-213

[Breiman, 2001] Breiman, L. Random Forests. *Machine Learning* 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[Connelly, 2005] Connelly, N. G. Nomenclature of inorganic chemistry: IUPAC recommendations 2005. Royal Society of Chemistry. (2005)

[Bechtel, 2023] Bechtel, T., Speckhard, D. T., Godwin, J., Draxl, C. (2023). Band-gap regression with architecture-optimized message-passing neural networks. arXiv preprint arXiv:2309.06348.