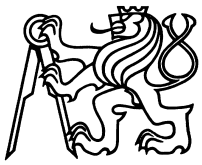
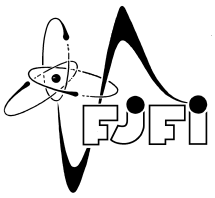


Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering



Machine learning for prediction of energy in condensed matter physics

Aplikace strojového učení k predikci energií ve fyzice pevných látek

Diploma Thesis

Author: **Bc. Jiří Chmel**

Supervisor: **doc. RNDr. Jan Vybíral, Ph.D.**

Academic year: 2021/2022

České vysoké učení technické v Praze, Fakulta jaderná a fyzikálně inženýrská

Katedra: matematiky Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student:	Bc. Jiří Chmel
Studijní program:	Aplikace přírodních věd
Studijní obor:	Aplikované matematicko-stochastické metody
Název práce (česky):	Aplikace strojového učení k predikci energií ve fyzice pevných látek
Název práce (anglicky):	Machine learning for prediction of energy in condensed matter physics

Pokyny pro vypracování:

1) Student se seznámí s metodou používanou k získávání dat o chemických sloučeninách.
2) Student se seznámí s přístupy používanými k získání vektorů popisu materiálů (tzv. deskriptory) ve fyzice pevných látek a vybrané aplikuje.
3) S využitím metod strojového učení student prozkoumá vztah mezi vlastnostmi materiálu (vazebná energie, šířka zakázaného pásu) a jeho geometrií.
4) Získané algoritmy student aplikuje na dostupné datasety z Fritz-Haberova Institutu v Berlíně a výsledky porovná s dostupnou literaturou.

Doporučená literatura:

1) L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, M. Scheffler, Big data of materials science - Critical role of the descriptor. Phys. Rev. Lett. 114, 2015, 105503.

2) L. M. Ghiringhelli, J. Vybiral, E. Ahmetchik, R. Ouyang, S. V. Levchenko, C. Draxl, M. Scheffler, Learning physical descriptors for materials science by compressed sensing. New Journal of Physics 19, 2017, 023017.

3) C. Sutton, L. M. Ghiringhelli, T. Yamamoto, Y. Lysogorskiy, L. Blumenthal, T. Hammerschmidt, J. R. Golebiowski, X. Liu, A. Ziletti, M. Scheffler, Crowd-sourcing materials-science challenges with the NOMAD 2018 Kaggle competition, Npj Comput. Mater. 5, 2019, 111.

4) C. M. Bishop, Pattern recognition and machine learning, Springer, 2006.

5) T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning: Data mining, inference, and prediction. Springer, New York, 2009.

Jméno a pracoviště vedoucího diplomové práce:

doc. RNDr. Jan Vybíral, Ph.D.
Katedra matematiky FJFI, ČVUT v Praze, Trojanova 13, 120 00 Praha 2

Jméno a pracoviště konzultanta:

Doc. Ing. Václav Šmídl, PhD.
ÚTIA AV ČR, Pod vodárenskou věží 4, 180 00 Praha 8

Datum zadání diplomové práce: 31.10.2020

Datum odevzdání diplomové práce: 3.5.2021

Doba platnosti zadání je dva roky od data zadání.

Acknowledgment:
I would like to thank doc. RNDr. Jan Vybíral, Ph.D. for his guidance and patience.

> *...and I shambled after as I've been doing all my life after people who interest me, because the only people for me are the mad ones, the ones who are mad to live, mad to talk, mad to be saved, desirous of everything at the same time, the ones who never yawn or say a commonplace thing, but burn, burn, burn like fabulous yellow roman candles exploding like spiders across the stars...*

Jack Kerouac, *On the Road*, 1957

Author's declaration:
I declare that this Diploma Thesis is entirely my own work and I have listed all the used sources in the bibliography.

Prague, May 1, 2022 Bc. Jiří Chmel

Název práce:

Aplikace strojové učení k predikci energií ve fyzice pevných látek

Autor: Bc. Jiří Chmel

Obor: Aplikované matematicko-stochastické metody

Druh práce: Diplomová práce

Vedoucí práce: doc. RNDr. Jan Vybíral, Ph.D., Katedra matematiky FJFI, ČVUT v Praze, Trojanova 13, 120 00 Praha 2

Konzultant: doc. Ing. Václav Šmídl, PhD., ÚTIA AV ČR, Pod vodárenskou věží 4, 180 00 Praha 8

Abstrakt: S rostoucím výpočetním výkonem posledních desetiletí došlo k zvětšení množství dat o krystalických materiálech vypočtených metodami density functional theory. Tyto výpočty bývají extrémně časově náročné. Spolu s nebývalým vývojem strojového učení vznikla příležitost využití těchto dat k predikci fyzikálních vlastností a tedy možnost obejít zdlouhavý numerický výpočet. Ukazuje se, že najít vhodnou transformaci vstupních dat není jednoduché a závisí především na konkrétním materiálu a predikované fyzikální veličině. Tato práce zkoumá již vyvinuté transformace vstupních dat a na jejich základě vyvíjí nové. Tyto přístupy jsou testovány na datech z Fritz Haberova Institutu v Berlíně v kombinace s mnoha metodami strojového učení.

Klíčová slova: datová analýza, density functional theory, fyzika materiálů, fyzika pevných látek, kvantová mechanika, strojové učení

Title:

Machine learning for prediction of energy in condensed matter physics

Author: Bc. Jiří Chmel

Abstract: In solid-state physics, the increase in computational capabilities leads to bigger datasets computed using density functional theory. The computations are extremely expensive in terms of needed time. Combining the extraordinary advancements of machine learning and the available data, there is an opportunity to use the data for prediction of physical properties without the need to perform the expensive density functional theory calculations. It appears that it is not easy to find a proper transformation of the data and it depends on the data and the physical quantity we want to predict. This work researches already established methods of data transformation and develops new ones based on the gathered knowledge. The new approaches are tested on datasets from Fritz Haber Institute of the Max Planck Society in Berlin combined with many machine learning methods.

Key words: condensed matter physics, data science, density functional theory, machine learning, quantum mechanics, solid-state physics

Contents

Introduction

The use of machine learning in physics has become increasingly widespread in the last decade. The combination of advancements in machine learning and the ability to collect big datasets calls for the use of machine learning tools in many areas of research and naturally, also in solid-state physics and computational chemistry. The gradual increase of computational power results in ever increasing availability of datasets calculated using numerical implementations of density functional theory (DFT) which approximates the many-body quantum mechanical description of our world and allows us to theoretically calculate physical properties. Nowadays, there is a community of researches who attempt to create machine learning models which could predict the desired physical properties in a fraction of the computational time needed to calculate them using conventional DFT implementations. This new branch of solid-state physics and computational chemistry has already found exciting applications, for example the screening of perovskites for solar cells [19], [22] and the screening of candidates for 2D ferromagnetic semiconductors, half-metals and metals [23]. The use of machine learning to some parts of computational chemistry and solid-state physics where DFT or methods based on classical physics were the only tools so far (e.g. drug design [32]) is exciting. Recently, deep learning was used in drug screening for COVID-19 treatment [30]. It is likely that machine learning will continue to be a very useful tool in this part of physics and chemistry.

The DFT data available for modeling requires the creation of features which conveniently capture the information. Soon, it became apparent that there is no general solution for the problem of feature generation. These features are called descriptors and the study of already developed descriptors was carried out in this work. We propose improvements of some descriptors, we implement them and test their capabilities. Multiple machine learning methods are used in this work, namely ordinary least squares, least absolute shrinkage and selection operator, kernel ridge regression and feedforward neural networks.

The text of this work is divided into 4 chapters and appendices. The first chapter outlines the machine learning methods used in this work. The second chapter provides a brief overview of the ideas which influenced the development of the descriptors in this work. The third chapter implements the experiment carried out in [10], [11] and [9] and provides deeper insight into the results. The fourth chapter studies the Nomad2018 Predicting Transparent Conductors Kaggle competition [35], [36] implements a selection of descriptors used in it, proposes some improvements and tests them on the data from the competition as well as more data provided by Fritz Haber Institute of the Max Planck Society in Berlin. The appendices contain details about the data and computational implementations.

Chapter 1

Methodology

The following chapter captures the mathematical background of the machine learning methods used in this work. The rigor of the utilized mathematical expressions is fine-tuned to explain the concepts and not to overwhelm the text with standard theory. The text starts with the simplest method of ordinary least squares (OLS) as the most basic and well-known statistical learning method. The OLS allows for various generalizations which extend its usability and the ones outlined in this work are the ridge regression and the LASSO. The ridge regression method is extended into kernel ridge regression. The pinnacle of this chapter is explanation of feedforward networks with emphasis on the architectures used in this work. The techniques used to transform the available datasets are provided and defined. Also, the methods of validating the created models are listed and explained.

The notation and conventions which will be used throughout this work are defined as stated below to eliminate any confusion which can easily occur [1].

Definition 1 (The $\hat{N}$ Notation)**.** The set of natural numbers $\{1, 2, \ldots, N\}$ is denoted by $\hat{N}$.

Definition 2 (Vector and Matrix Notation)**.** A vector of real numbers $\boldsymbol{x} \in \mathbb{R}^N$, $N \in \mathbb{N}$ is denoted by

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} = \left(x_1, x_2, \ldots, x_N\right)^T . \tag{1.1}$$

A matrix of real numbers $\boldsymbol{X} \in \mathbb{R}^{N \times M}$, $M, N \in \mathbb{N}$ is denoted by

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1M} \\ x_{21} & x_{22} & \ldots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & \ldots & \ldots & x_{NM} \end{pmatrix} = \left(\boldsymbol{x}_{\bullet 1}, \boldsymbol{x}_{\bullet 2}, \ldots, \boldsymbol{x}_{\bullet M}\right) = \left(\boldsymbol{x}_{1\bullet}, \boldsymbol{x}_{2\bullet}, \ldots, \boldsymbol{x}_{N\bullet}\right)^T , \tag{1.2}$$

where $\boldsymbol{x}_{\bullet j} \in \mathbb{R}^N$ for $j \in \hat{M}$ are columns of $\boldsymbol{X}$ and $\boldsymbol{x}_{i\bullet} \in \mathbb{R}^M$ for $i \in \hat{N}$ are rows of $\boldsymbol{X}$.

Definition 3 (Centered Input and Centered Matrix)**.** We define centered input of a matrix $\boldsymbol{X}$ at row i and column j as

$$x_{ij}^c = x_{ij} - \bar{x}_j = x_{ij} - \frac{1}{N} \sum_{k=1}^{N} x_{kj}, \forall i, j \in \hat{N}, \hat{M}. \tag{1.3}$$

[1] "This ambiguity is another example of a growing problem with mathematical notation: There aren't enough squiggles to go around." - Jim Blinn

Centered matrix $\boldsymbol{X}^c$ to a matrix $\boldsymbol{X}$ is the matrix which inputs have the form of (1.3).

Definition 4 (The ℓ_p Norm)**.** Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_M)^T \in \mathbb{R}^M$.

1. If $p \in [1, +\infty)$, then the ℓ_p norm of a vector $\boldsymbol{x}$ is

$$\|\boldsymbol{x}\|_p = \left(\sum_{j=1}^{M} |x_j|^p \right)^{\frac{1}{p}}. \tag{1.4}$$

2. The ℓ_0 norm ($p = 0$) of a vector $\boldsymbol{x} \in \mathbb{R}^M$ is

$$\|\boldsymbol{x}\|_0 = \#\{j : x_j \neq 0\}, \tag{1.5}$$

which counts the number of non-zero components of $\boldsymbol{x}$.

Remark 1. For $0 < p < 1$ the convexity is broken - the triangle inequality does not hold and ℓ_0 norm cannot be defined the same way as for $p \geq 1$. Instead, the triangle inequality with a constant is satisfied

$$\|\boldsymbol{x} + \boldsymbol{y}\|_p \leq C(\|\boldsymbol{x}\|_p + \|\boldsymbol{y}\|_p), \tag{1.6}$$

for $C > 0$ and vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^M$.

1.1 Regression Methods

The methods used in this work and their underlying theory is outlined in this section. As stated in the introduction of this chapter, ordinary least squares (OLS), ridge regression, kernel ridge regression (KRR), the least absolute shrinkage and selection operator (LASSO) and neural networks are described.

Let us assume we have a real setting with data points $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)$, where $\boldsymbol{x}_i \in \mathbb{R}^M$ for $i \in \hat{N}$ are regressors and $y_1, y_2, \ldots, y_N \in \mathbb{R}$ are responses. The regressors are assumed to be fixed numbers. Generally, we want to describe the dependence of the output y_i on $\boldsymbol{x}_i$ - in other words we want to model the relation $y_i = f(\boldsymbol{x}_i)$ for $i \in \hat{N}$ where the function f is the actual relationship between the regressors and responses. The goal of regression is to find the most suitable approximation of f for the given problem and evaluate the performance of such approximation.

1.1.1 Ordinary Least Squares

The linear model of ordinary least squares is the most well-known method of statistical learning. We presume the model is linear in the coefficients

$$\boldsymbol{y}_i = f(\boldsymbol{x}_i) = \langle \boldsymbol{x}_i, \boldsymbol{b} \rangle, \tag{1.7}$$

where $\boldsymbol{b} = (b_1, b_2, \ldots, b_M)^T \in \mathbb{R}^M$ is the vector of the coefficients and $\langle \cdot, \cdot \rangle$ is the scalar product of two vectors. Our goal is to obtain the coefficients $\hat{\boldsymbol{b}} = (\hat{b}_1, \hat{b}_2, \ldots, \hat{b}_M)^T \in \mathbb{R}^M$. Generally, we want to add an absolute term called *bias* (or *intercept*) to our linear regression model. We elegantly do so by adding a column of ones to the matrix $\boldsymbol{X}$ which is then $N \times (M + 1)$ dimensional and the vector of coefficients $\hat{\boldsymbol{b}}$ is $(M + 1)$ dimensional. It will be assumed (unless explicitly said otherwise) that bias is included in the model - in other words, we implicitly assume the vector of ones is already enumerated in all expressions ($M + 1 \to M$).

Now, we can write down the mathematical formulation of the model more explicitly. The linear regression model for a matrix of regressors $\boldsymbol{X} \in \mathbb{R}^{N\times M}$ and a vector of responses $\boldsymbol{y} \in \mathbb{R}^N$ has the form

$$y_i \approx \sum_{j=1}^{M} x_{ij} b_j = \langle \boldsymbol{x}_{i\bullet}, \boldsymbol{b} \rangle, i \in \hat{N}. \tag{1.8}$$

We want to find the estimate of the vector of coefficients $\boldsymbol{b} = (b_1, \ldots, b_M)^T \in \mathbb{R}^M$. We perform the minimization of quadratic loss $J_{OLS}(\boldsymbol{b}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2$ called the least squares

$$\hat{\boldsymbol{b}}_{OLS} = \underset{\boldsymbol{b}\in\mathbb{R}^M}{\operatorname{argmin}} J_{OLS}(\boldsymbol{b}) = \underset{\boldsymbol{b}\in\mathbb{R}^M}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}\|_2^2 = \underset{\boldsymbol{b}\in\mathbb{R}^M}{\operatorname{argmin}} \sum_{i=1}^{N} \Big(y_i - \langle \boldsymbol{x}_{i\bullet}, \boldsymbol{b} \rangle\Big)^2 = \underset{\boldsymbol{b}\in\mathbb{R}^M}{\operatorname{argmin}} \sum_{i=1}^{N} \Big(y_i - \sum_{k=1}^{M} x_{ik} b_k\Big)^2. \tag{1.9}$$

It is easily seen that the problem in (1.9) is convex and the solution can be found in a closed form. We take the derivative of the quadratic loss with respect to the coefficients

$$\frac{\partial J_{OLS}}{\partial b_j} = \frac{\partial}{\partial b_j} \sum_{i=1}^{N} \Big(y_i - \sum_{k=1}^{M} x_{ik} b_k\Big)^2 = -2 \sum_{i=1}^{N} x_{ij} \Big(y_i - \sum_{k=1}^{M} x_{ik} b_k\Big), \forall j \in \hat{M}. \tag{1.10}$$

The expression above can be written in a compressed form as follows

$$\frac{\partial J_{OLS}}{\partial \boldsymbol{b}} = -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}) = -2(\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b}). \tag{1.11}$$

If $\boldsymbol{X}^T\boldsymbol{X}$ is regular, the unique solution can be recovered from

$$0 = \frac{\partial J_{OLS}}{\partial \boldsymbol{b}} = \boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{b}, \tag{1.12}$$

which means the OLS approximation of the coefficients $\hat{\boldsymbol{b}}$ is given as

$$\hat{\boldsymbol{b}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{1.13}$$

The predicted values at the training inputs are then

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{b}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{H}\boldsymbol{y}. \tag{1.14}$$

The matrix $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ is called the "hat" matrix because it puts the hat on vector $\boldsymbol{y}$ and it computes the projection $\hat{\boldsymbol{y}}$ onto the hyperplane spanned by the columns of $\boldsymbol{X}$. Therefore the vector $\boldsymbol{y} - \hat{\boldsymbol{y}}$ is orthogonal to this hyperplane.

If the $\boldsymbol{X}^T\boldsymbol{X}$ matrix is singular then there is not a unique solution. Usually, this problem can be solved by localizing the linearly dependent columns of $\boldsymbol{X}$ and excluding some of them until regularity is reached. In general, the inverse of a singular matrix $\boldsymbol{X}^T\boldsymbol{X}$ is recoverable in the form of Penrose inverse which always exists and is unique [12] (str.46 goodfellow zeptat se jestli tento zdroj je ok).

1.1.2 Ridge Regression

One of the solutions to the problem of the singularity of the matrix $\boldsymbol{X}$ from the previous section is to add a regularization term into the OLS loss function as follows

$$J_{ridge}(b_0, \boldsymbol{b}) = \sum_{i=1}^{N} \Big(y_i - b_0 - \sum_{k=1}^{M} x_{ik} b_k\Big)^2 + \lambda \sum_{k=1}^{M} b_k^2 = \|\boldsymbol{y} - \boldsymbol{1}b_0 - \boldsymbol{X}\boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{b}\|_2^2,\ \lambda > 0. \tag{1.15}$$

Here, we choose not to add the intercept b_0 into the newly added term in (1.15) and we are explicit about the intercept term in the previous term. The penalization of the intercept would make the process depend on the origin chosen for the responses which basically means that making a shift of the responses by a constant would not shift the predictions by the same constant. The $\mathbf{1}$ symbol means a vector of ones, $\mathbf{1} \in \mathbb{R}^N$.

The solution of the ridge regression problem is obtained by finding the following

$$\begin{pmatrix} b_0 \\ \hat{\boldsymbol{b}}_{ridge} \end{pmatrix} = \underset{\left[\begin{smallmatrix} b_0 \\ \boldsymbol{b} \end{smallmatrix}\right] \in \mathbb{R}^{M+1}}{\operatorname{argmin}} J_{ridge}(b_0, \boldsymbol{b}) = \underset{\left[\begin{smallmatrix} b_0 \\ \boldsymbol{b} \end{smallmatrix}\right] \in \mathbb{R}^{M+1}}{\operatorname{argmin}} \left(\|\boldsymbol{y} - \mathbf{1}b_0 - \boldsymbol{X}\boldsymbol{b}\|_2^2 + \lambda \|\boldsymbol{b}\|_2^2 \right). \tag{1.16}$$

The concept of regularization basically means we impose a penalty on the size of $\boldsymbol{b}$. The parameter λ controls the strength of the regularization. We get OLS coefficients for $\lambda \to 0^+$ and $\hat{\boldsymbol{b}}_{ridge} = \mathbf{0}$ for $\lambda \to +\infty$. The solution can be found by reparametrization of (1.15) using centered inputs

$$J_{ridge}(b_0, \boldsymbol{b}) = \sum_{i=1}^{N} \left(y_i - b_0 - \sum_{k=1}^{M} \bar{x}_k b_k - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) b_k \right)^2 + \lambda \sum_{k=1}^{M} b_k^2 = \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) \tilde{b}_k \right)^2 + \lambda \sum_{k=1}^{M} \tilde{b}_k^2. \tag{1.17}$$

The new coefficients $\tilde{\boldsymbol{b}}$ satisfy following equations

$$\begin{aligned} \tilde{b}_0 &= b_0 + \sum_{k=1}^{M} \bar{x}_k b_k, \\ \tilde{b}_j &= b_j, \forall j \in \hat{M}. \end{aligned} \tag{1.18}$$

Then, the solution can be found using the very same procedure as in (1.10) or (1.11)

$$\begin{aligned} \frac{\partial J_{ridge}}{\partial \tilde{b}_0} &= \frac{\partial}{\partial \tilde{b}_0} \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) \tilde{b}_k \right)^2 + \frac{\partial}{\partial \tilde{b}_0} \lambda \sum_{k=1}^{M} \tilde{b}_k^2 = -2 \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) \tilde{b}_k \right) \\ \frac{\partial J_{ridge}}{\partial \tilde{b}_j} &= \frac{\partial}{\partial \tilde{b}_j} \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) \tilde{b}_k \right)^2 + \frac{\partial}{\partial \tilde{b}_j} \lambda \sum_{k=1}^{M} \tilde{b}_k^2 = \\ &= -2 \sum_{i=1}^{N} (x_{ij} - \bar{x}_j) \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} (x_{ik} - \bar{x}_k) \tilde{b}_k \right) + 2\lambda \tilde{b}_j, \forall j \in \hat{M}. \end{aligned} \tag{1.19}$$

The solution can be once again expressed in a compressed form for the second equation above

$$\frac{\partial J_{ridge}}{\partial \tilde{\boldsymbol{b}}} = -2 \boldsymbol{X}^{cT} (\boldsymbol{y} - \mathbf{1}\tilde{b}_0 - \boldsymbol{X}^c \tilde{\boldsymbol{b}}) + 2\lambda \boldsymbol{I} \tilde{\boldsymbol{b}}. \tag{1.20}$$

We find the solution by putting the first derivative of the loss function by the coefficients to zero

$$\begin{aligned} 0 = \frac{\partial J_{ridge}}{\partial \tilde{b}_0} &= -2 \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} x_{ik} \tilde{b}_k \right) = \sum_{i=1}^{N} \left(y_i - \tilde{b}_0 - \sum_{k=1}^{M} x_{ik} \tilde{b}_k \right) \\ 0 = \frac{\partial J_{ridge}}{\partial \tilde{\boldsymbol{b}}} &= -2 \boldsymbol{X}^{cT} (\boldsymbol{y} - \mathbf{1}\tilde{b}_0 - \boldsymbol{X}^c \tilde{\boldsymbol{b}}) + 2\lambda \tilde{\boldsymbol{b}} = -\boldsymbol{X}^{cT} (\boldsymbol{y} - \mathbf{1}\tilde{b}_0 - \boldsymbol{X}^c \tilde{\boldsymbol{b}}) + \lambda \boldsymbol{I} \tilde{\boldsymbol{b}} \end{aligned} \tag{1.21}$$

where $\boldsymbol{I}$ is the identity matrix. The solution for the intercept arises from the first equation above as

$$\tilde{b}_0 = \frac{1}{N} \sum_{i=1}^{N} y_i = \bar{y}. \tag{1.22}$$

Finally, the solution for the rest of the coefficients can be extracted as

$$\hat{\boldsymbol{b}}_{ridge} = (\boldsymbol{X}^{cT}\boldsymbol{X}^{c} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}^{cT}(\boldsymbol{y} - \mathbf{1}\tilde{b}_0) = (\boldsymbol{X}^{cT}\boldsymbol{X}^{c} + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}^{cT}(\boldsymbol{y} - \mathbf{1}\bar{y}). \tag{1.23}$$

It is important to standardize the columns of the matrix $\boldsymbol{X}$ before training to eliminate spurious behavior. The shape of the equation (1.23) shows the reason why this procedure works: the regularization stabilizes the inverse of the matrix for some value of λ. This regularization also helps to avoid overfitting. The optimal value of λ is usually chosen using cross validation.

1.1.3 Kernel Ridge Regression (KRR)

Kernel ridge regression (KRR) builds on top of ridge regression and allows modeling of nonlinear relationships between regressors and responses. We put $\boldsymbol{x}_{i\bullet} = \boldsymbol{x}_i$ to make the notation less cumbersome in the following text. The datapoints themselves are replaced with a feature vector $\boldsymbol{x}_i \to \phi(\boldsymbol{x}_i)$ where $\phi : \mathbb{R}^M \to \mathcal{F}$ is a nonlinear mapping to a higher dimensional feature space $\mathcal{F}$, $\dim(\mathcal{F}) \leq +\infty$. Now, we consider datapoints $(\phi(\boldsymbol{x}_1), y_1), \ldots, (\phi(\boldsymbol{x}_N), y_N)$ for the very same learning algorithm of ridge regression. In other words, we find ridge regression coefficients and create a linear model in feature space where datapoints $(\phi(\boldsymbol{x}_1), y_1), \ldots, (\phi(\boldsymbol{x}_N), y_N)$ are but we observe a nonlinear model in space where datapoints $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)$ are.

We define the loss function of KRR in a similar fashion as in (1.15)

$$J_{KRR}(b_0, \boldsymbol{b}) = \sum_{i=1}^{N}\Big(y_i - b_0 - \boldsymbol{b}^T\phi(\boldsymbol{x}_i)\Big)^2 + \lambda\sum_{k=1}^{M} b_k^2 = \|\boldsymbol{y} - \mathbf{1}b_0 - \Phi\boldsymbol{b}\|_2^2 + \lambda\|\boldsymbol{b}\|_2^2, \tag{1.24}$$

where $\lambda > 0$ and $\Phi = \begin{pmatrix} \phi^T(\boldsymbol{x}_1) \\ \vdots \\ \phi^T(\boldsymbol{x}_N) \end{pmatrix}$ is the mapping of matrix $\boldsymbol{X}$. Here, we choose not to explicitly note the prerequisites we utilized during the pursuit of the ridge regression solution (e.g. centering of the regressors). Setting the gradient of J_{KRR} in (1.24) equal to zero gives

$$\begin{aligned} b_0 &= \frac{1}{N}\sum_{i=1}^{N} y_i = \bar{y} = a_0 \\ \boldsymbol{b} &= -\frac{1}{\lambda}\sum_{i=1}^{N}\Big(y_i - b_0 - \boldsymbol{b}^T\phi(\boldsymbol{x}_i)\Big)\phi(\boldsymbol{x}_i) = \sum_{i=1}^{N} a_i\phi(\boldsymbol{x}_i) = \Phi^T\boldsymbol{a}, \end{aligned} \tag{1.25}$$

where we put

$$a_i = -\frac{1}{\lambda}\Big(y_i - b_0 - \boldsymbol{b}^T\phi(\boldsymbol{x}_i)\Big),\ i \in \hat{N}. \tag{1.26}$$

The result of (1.25) allows us to reformulate the loss function (1.24) in terms of $a_0, \boldsymbol{a}$ instead of $b_0, \boldsymbol{b}$

$$J_{KRR}(a_0, \boldsymbol{a}) = \|\boldsymbol{y} - \mathbf{1}a_0 - \Phi\Phi^T\boldsymbol{a}\|_2^2 + \lambda\|\Phi^T\boldsymbol{a}\|_2^2 = \|\boldsymbol{y} - \mathbf{1}a_0 - \Phi\Phi^T\boldsymbol{a}\|_2^2 + \lambda\boldsymbol{a}^T\Phi\Phi^T\boldsymbol{a}. \tag{1.27}$$

Let us examine the result. We put $K = \Phi\Phi^T$. Therefore

$$K_{ij} = (\Phi\Phi^T)_{ij} = \langle\phi(\boldsymbol{x}_i),\ \phi(\boldsymbol{x}_j)\rangle = k(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{1.28}$$

where we introduce the kernel function $k : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$. The loss function then takes very elegant form

$$J_{KRR}(\boldsymbol{a}) = \|\boldsymbol{y} - \mathbf{1}a_0 - K\boldsymbol{a}\|_2^2 + \lambda \boldsymbol{a}^T K \boldsymbol{a}, \tag{1.29}$$

in comparison with (1.24). This is the final form of the loss function for KRR with kernel K. Setting the gradient of J_{KRR} with respect to $a_0, \boldsymbol{a}$ in (1.29) to zero gives the final solution

$$\boldsymbol{a} = (K + \lambda \boldsymbol{I})^{-1}(\boldsymbol{y} - \mathbf{1}\bar{y}), \tag{1.30}$$

and the original coefficients

$$\boldsymbol{b} = \Phi^T (K + \lambda \boldsymbol{I})^{-1}(\boldsymbol{y} - \mathbf{1}\bar{y}). \tag{1.31}$$

The prediction y_{pred} for a new datapoint $\boldsymbol{x}$ can be expressed elegantly as

$$y_{pred} = b_0 + \boldsymbol{b}^T \phi(\boldsymbol{x}) = a_0 + \boldsymbol{a}^T \Phi \phi(\boldsymbol{x}) = \bar{y} + (\boldsymbol{y} - \mathbf{1}\bar{y})^T (K + \lambda \boldsymbol{I})^{-1} \Phi \phi(\boldsymbol{x}) = \bar{y} + (\boldsymbol{y} - \mathbf{1}\bar{y})^T (K + \lambda \boldsymbol{I})^{-1} \kappa(\boldsymbol{x}). \tag{1.32}$$

where $\kappa(\boldsymbol{x}) = \begin{pmatrix} \phi^T(\boldsymbol{x}_1) \\ \vdots \\ \phi^T(\boldsymbol{x}_N) \end{pmatrix} \phi(\boldsymbol{x}) = \begin{pmatrix} \langle \phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}) \rangle \\ \vdots \\ \langle \phi(\boldsymbol{x}_N), \phi(\boldsymbol{x}) \rangle \end{pmatrix} = \begin{pmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}) \\ \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}) \end{pmatrix}$. We shall see that we can avoid working with the mapping ϕ which can even fulfill $dim(\mathcal{F}) = \infty$. The object of our interest is the kernel function k and we will show it is all we need in the following section.

So far, we dealt with the kernel K and its kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ without specifying any needed properties of these mathematical objects. The following theorem explains why we can avoid working with the cumbersome mapping ϕ and justifies and explains our previous steps and operations we performed with it [26].

Theorem 1 (Mercer)**.** To guarantee that the symmetric continuous function $k(\boldsymbol{x}, \boldsymbol{y}) : C \times C \to \mathbb{R}$ on compact set $C \subset \mathbb{R}^N$ has an expansion

$$k(\boldsymbol{x}, \boldsymbol{y}) = \sum_{k=1}^{N_{\mathcal{F}}} \lambda_j \psi_j(\boldsymbol{x}) \psi_j(\boldsymbol{y}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle \tag{1.33}$$

with $\lambda_j > 0$ and $\phi : \mathbb{R}^N \to \mathcal{F}$ with $\dim(\mathcal{F}) = N_{\mathcal{F}} \leq +\infty$, it is necessary and sufficient that the function k is a kernel of a positive integral operator on $L_2(C)$:

$$\forall f \in L_2(C) : \int_C \int_C k(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{x}) f(\boldsymbol{y}) \mathrm{d}\boldsymbol{x} \mathrm{d}\boldsymbol{y} \geq 0. \tag{1.34}$$

Proof. Can be found in [24]. □

It is easy to see that a possible realization of the mapping can have the form

$$\phi(\boldsymbol{x}) = \left(\sqrt{\lambda_1} \psi_1(\boldsymbol{x}), \sqrt{\lambda_2} \psi_2(\boldsymbol{x}), \dots \right).$$

The significance of this result is that we do not need to know the shape of ϕ. This fact is often called the kernel trick. The dimensionality of ϕ is infinite for Gaussian kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{y}\|_2^2)$ which can be seen from the following decomposition

$$\begin{aligned} \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle = k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{y}\|_2^2) = \exp(-\gamma \|\boldsymbol{x}\|_2^2 + 2\gamma \langle \boldsymbol{x}, \boldsymbol{y} \rangle - \gamma \|\boldsymbol{y}\|_2^2) = \\ = \exp(-\gamma \|\boldsymbol{x}\|_2^2) \exp(2\gamma \langle \boldsymbol{x}, \boldsymbol{y} \rangle) \exp(-\gamma \|\boldsymbol{y}\|_2^2)). \end{aligned} \tag{1.35}$$

Taking the middle term and using the fact that binomial expansion $\langle \boldsymbol{x}, \boldsymbol{y} \rangle^n, n \in \mathbb{N}$ exists for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^N$ and Taylor expansion of e^x exists for all $x \in \mathbb{R}$

$$\begin{aligned}\exp(2\gamma\langle \boldsymbol{x}, \boldsymbol{y}\rangle) &= 1 + 2\gamma\langle \boldsymbol{x}, \boldsymbol{y}\rangle + \frac{(2\gamma)^2\langle \boldsymbol{x}, \boldsymbol{y}\rangle^2}{2} + \frac{(2\gamma)^3\langle \boldsymbol{x}, \boldsymbol{y}\rangle^3}{6} + \cdots = \\ &= 1 + 2\gamma(x_1y_1 + x_2y_2 + \cdots + x_Ny_N) + \cdots = \\ &= \left\langle (1, \sqrt{2\gamma}x_1, \sqrt{2\gamma}x_2, \ldots, \sqrt{2\gamma}x_N, \ldots)^T, (1, \sqrt{2\gamma}y_1, \sqrt{2\gamma}y_2, \ldots, \sqrt{2\gamma}y_N, \ldots)^T \right\rangle,\end{aligned} \tag{1.36}$$

where we do not list higher order expansion terms for visibility. The mapping function can be expressed as

$$\phi(\boldsymbol{x}) = \exp(-\gamma\|\boldsymbol{x}\|_2^2)\Big(1, \sqrt{2\gamma}x_1, \sqrt{2\gamma}x_2, \ldots, \sqrt{2\gamma}x_N, \ldots\Big)^T, \tag{1.37}$$

and the dimension of $\mathcal{F}$ is infinite because of the Taylor expansion we used.

New kernels can be constructed from already developed kernels. We list a few of the techniques in Table 1.1.

Construction Technique
$k(\boldsymbol{x}, \boldsymbol{y}) = ck_1(\boldsymbol{x}, \boldsymbol{y})$
$k(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x})k_1(\boldsymbol{x}, \boldsymbol{y})f(\boldsymbol{y})$
$k(\boldsymbol{x}, \boldsymbol{y}) = q(k_1(\boldsymbol{x}, \boldsymbol{y}))$
$k(\boldsymbol{x}, \boldsymbol{y}) = \exp(k_1(\boldsymbol{x}, \boldsymbol{y}))$
$k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}, \boldsymbol{y}) + k_2(\boldsymbol{x}, \boldsymbol{y})$
$k(\boldsymbol{x}, \boldsymbol{y}) = k_1(\boldsymbol{x}, \boldsymbol{y})k_2(\boldsymbol{x}, \boldsymbol{y})$
$k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y}$

Table 1.1: The $k_1(\boldsymbol{x}, \boldsymbol{y})$ and $k_2(\boldsymbol{x}, \boldsymbol{y})$ are valid kernels, constant $c > 0$, f is a real function defined on $\mathbb{R}^N$, q is a polynomial with nonnegative coefficients and $\boldsymbol{A}$ is a symmetric positive semidefinite matrix

We can construct the Gaussian kernel from the linear kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{y}$ which is a trivial identity. We use the second and the fourth technique in Table 1.1

$$k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma\|\boldsymbol{x}\|_2^2 + 2\gamma\langle \boldsymbol{x}, \boldsymbol{y}\rangle\gamma + \|\boldsymbol{y}\|_2^2) = \exp(-\gamma\boldsymbol{x}^T\boldsymbol{x})\exp(2\gamma\boldsymbol{x}^T\boldsymbol{y})\exp(-\gamma\boldsymbol{y}^T\boldsymbol{y}). \tag{1.38}$$

Some commonly used kernels are listed in Table 1.2.

	Kernels
Gaussian	$\exp(-\gamma\|\boldsymbol{x} - \boldsymbol{y}\|_2^2)$
Laplacian	$\exp(-\gamma\|\boldsymbol{x} - \boldsymbol{y}\|_1)$
Sigmodial,	$\tanh(\kappa(\boldsymbol{x} \cdot \boldsymbol{y}) + \theta)$
Polynomial	$(\boldsymbol{x} \cdot \boldsymbol{y} + \theta)^d$

Table 1.2: Commonly used kernels. $\gamma > 0$, $\kappa \in \mathbb{R}$, $\theta \in \mathbb{R}$, $d \in \mathbb{N}$.

We are concerned with the Gaussian and Laplacian kernels because of their form. These two kernels have the property $k(\boldsymbol{x}, \boldsymbol{y}) = k(\|\boldsymbol{x} - \boldsymbol{y}\|_p)$ where $p \geq 1$ and are called radial basis functions. This property will play an important role in the carried out experiments.

Kernel ridge regression with Gaussian or Laplacian kernel has two parameters λ and γ which have to be optimized outside of the training procedure. Such numbers are called hyperparameters and they are usually tuned using cross validation.

1.1.4 The Least Absolute Shrinkage and Selection Operator (LASSO)

The LASSO emerged as a technique to obtain low-dimensional solutions to regression problems and interestingly enough, long before the underlying theory was developed and understood thoroughly. Since its establishment as a useful method, LASSO made its way into the portfolio of virtually every machine learning engineer. However, the proper use of the method with all the constrains fulfilled is not always done as it should be. We choose to outline the theory needed to define LASSO with careful attention towards the use in the experiments carried out in this work [4].

The mathematical theory of compressed sensing is the underlying cornerstone of LASSO. We start with defining a few objects which will be useful later on.

Definition 5 (k-sparse vectors)**.** Let $k \in \mathbb{N}$ such that $k < M$. A vector $\boldsymbol{x} \in \mathbb{R}^M$ is called k-sparse if $\|x\|_0 \leq k$. The set of all k-sparse vectors is

$$\mathbb{R}^M_k = \{\boldsymbol{x} \in \mathbb{R}^M : \|x\|_0 \leq k\} \tag{1.39}$$

Remark 2. It is easy to see that for every, $\boldsymbol{x} \in \mathbb{R}^M$ there is a permutation π: $\hat{M} \mapsto \hat{M}$ such that

$$|x_{\pi(1)}| \geq |x_{\pi(2)}| \geq \cdots \geq |x_{\pi(M)}| \geq 0. \tag{1.40}$$

The vector $\boldsymbol{x}^* \in \mathbb{R}^M$ with components $x^*_j = |x_{\pi(j)}|$ for $j \in \hat{M}$ is called nonincreasing rearrangement of $\boldsymbol{x}$.

Definition 6 (The Best k-term Approximation)**.** Let $k \leq M$ and ℓ_p be a norm, $p > 1$. The best k-term approximation $\sigma_k(\boldsymbol{x})_p$ of $\boldsymbol{x} \in \mathbb{R}^M$ is

$$\sigma_k(\boldsymbol{x})_p = \inf_{\tilde{\boldsymbol{x}} \in \mathbb{R}^M_k} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_p = \Big(\sum_{j=k+1}^{M} |x^*_j|^p \Big)^{\frac{1}{p}}. \tag{1.41}$$

ℓ_0 Minimization and Basis Pursuit

Definition 7 (ℓ_0 Minimization)**.** Let $\boldsymbol{x} \in \mathbb{R}^M$, $\boldsymbol{A} \in \mathbb{R}^{N \times M}$ be known and $\boldsymbol{y} \in \mathbb{R}^N$ be known. The ℓ_0 minimization problem is defined as

$$\min_{\boldsymbol{x} \in \mathbb{R}^M} \|\boldsymbol{x}\|_0 \ \textit{subject to}\ \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}. \tag{1.42}$$

Remark 3. It will be shown that ℓ_0 minimization is numerically a very expensive optimization problem. For this purpose, we introduce the classes of complexity:

- P class - all decision problems which can be solved in polynomial time.
- NP class - a candidate for solution can be tested in polynomial time.
- NP-hard class - decision problems for which all their solving algorithms can be transformed in polynomial time into a solving algorithm of any other NP problem.
- NP-complete class - those decision problems which are NP-hard and NP.

Here, we will present without a proof a problem from complexity theory called Three Cover Problem which is NP-complete.

Three Cover Problem

Let $N \in \mathbb{N}$ be divisible by 3 and $M \in \mathbb{N}$. For a given system $\{T_j : j \in \hat{M}\}$ of subsets of $\hat{N}$ and $\#T_j = 3$ for $\forall j \in \hat{M}$. **Decision problem:** Decide of the existence of a subsystem $\{T_j : j \in J\}$ for which holds:

1. $\bigcup_{j\in J} T_j = \hat{N}$,
2. $T_i \bigcap T_j = \varnothing$ for $i, j \in J, i \neq j$.

Theorem 2. The ℓ_0 minimization problem is NP-hard.

Proof. The problem (1.42) will be reformulated as the Three Cover Problem. Using the notation in the definiton of the Three Cover problem we construct a matrix $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ which columns $\boldsymbol{a}_j$ are the characteristic functions of the given T_j. Therefore the components of $\boldsymbol{a}_j$ are defined as:

$$a_{ij} := \begin{cases} 1 & \text{if } i \in T_j \\ 0 & \text{if } i \notin T_j. \end{cases}$$

The vector and matrix multiplication gives

$$\boldsymbol{A}\boldsymbol{x} = \sum_{j=1}^{M} x_j \boldsymbol{a}_j.$$

It is easy to see from the construction itself that the matrix $\boldsymbol{A}$ can be constructed in polynomial time. Let's presume $\boldsymbol{x}$ is the solution of the ℓ_0 minimization problem with $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} = (1, \dots, 1)^T$. The vector and matrix multiplication causes the amount of nonzero components of $\boldsymbol{x}$ to be at most three times bigger:

$$N = \|\boldsymbol{y}\|_0 = \|\boldsymbol{A}\boldsymbol{x}\|_0 \leq 3\|\boldsymbol{x}\|_0 \Leftrightarrow \|\boldsymbol{x}\|_0 \geq N/3.$$

We will show: The Exact Cover problem has a solution if and only if $\|\boldsymbol{x}\|_0 = N/3$.
$\Rightarrow$: $J \subset \hat{M}$ and the amount of columns needed is precisely $N/3$, that is $|J| = N/3$ and

$$(1, \dots, 1)^T = \sum_{j\in J} \boldsymbol{a}_j = \sum_{j=1}^{M} x_j \boldsymbol{a}_j.$$

Now, it is easy to see that $\boldsymbol{x}$ has nonzero components where each nonzero component is equal to one and only for the indices from index set J. This gives $\|\boldsymbol{x}\|_0 = |J| = N/3$.
$\Leftarrow$: Let $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ with $\|\boldsymbol{x}\|_0 = N/3$. In such case, we choose a subsystem $\{T_j : j \in \text{supp}(\boldsymbol{x})\}$. □

With ℓ_0 minimization being too difficult to solve for any $\boldsymbol{A}$ and $\boldsymbol{y}$, we are forced to find a feasible compromise. We demand the problem to be convex and also promote sparsity. Convexity will be ensured if we choose to use ℓ_p norm where $p \geq 1$. Sparsity will be possible for $p \leq 1$. Therefore we are left with no other choice than $p = 1$ and explore whether such optimization problem can work for our purposes. It turns out that it can recover sparse solutions for certain matrices.

Definition 8 (Basis Pursuit)**.** Let $\boldsymbol{x} \in \mathbb{R}^M$, $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ be known and $\boldsymbol{y} \in \mathbb{R}^N$ be known. The ℓ_1 minimization problem called Basis Pursuit is defined as

$$\min_{\boldsymbol{x}\in\mathbb{R}^M} \|\boldsymbol{x}\|_1 \text{ subject to } \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}. \tag{1.43}$$

Null Space Property

Remark 4. Before we define Null Space Property, we will introduce useful notation which will be used onward. The number of elements of a finite set T is denoted $\#T$. For $T \subset \hat{M}$ we denote by $T^C = \hat{M}\backslash T$ the complement of T in $\hat{M}$. For $\boldsymbol{v} \in \mathbb{R}^M$, we denote $\boldsymbol{v}_T$ the vector in $\mathbb{R}^{\#T}$, which contains the coordinates of $\boldsymbol{v}$ indexed by T or the vector in $\mathbb{R}^M$ which equals $\boldsymbol{v}$ on T and has zero components on T^C.

Definition 9 (Null Space Property)**.** Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ and $k \in \hat{M}$. Then $\boldsymbol{A}$ has the Null Space Property (NSP) of order k if

$$\|\boldsymbol{v}_T\|_1 < \|\boldsymbol{v}_{T^C}\|_1 : \forall \boldsymbol{v} \in \ker\boldsymbol{A}\backslash\{\boldsymbol{0}\} \textit{ and } \forall T \subset \hat{M} \textit{ with } |T| \leq k. \tag{1.44}$$

Remark 5. The Null Space Property of a matrix says that the components of vectors of the kernel are not supported solely on a few components. It is easy to see that the inequality in (1.44) can be equivalently expressed as $\|\boldsymbol{v}\|_1 < 2\|\boldsymbol{v}_{T^C}\|_1$ or $2\|\boldsymbol{v}_T\|_1 < \|\boldsymbol{v}\|_1$. The following theorem shows the relation between k-sparse solutions of (1.43) and NSP.

Theorem 3. Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ and $k \in \hat{M}$. Then every k-sparse vector $\boldsymbol{x} \in \mathbb{R}^M$ is the unique solution of (1.43) with $\boldsymbol{A}$ if and only if $\boldsymbol{A}$ has the NSP of order k.

Proof.
$\Rightarrow$: Let $\boldsymbol{v} \in \ker\boldsymbol{A}\backslash\{\boldsymbol{0}\}$, $T \subset \hat{M}$, $|T| \leq k$ arbitrary. Then from the presumption $\boldsymbol{v}_T$ is the unique solution of (1.43). Also,

$$\boldsymbol{0} = \boldsymbol{A}\boldsymbol{v} = \boldsymbol{A}(\boldsymbol{v}_T + \boldsymbol{v}_{T^C}) \Leftrightarrow \boldsymbol{A}(-\boldsymbol{v}_{T^C}) = \boldsymbol{A}(\boldsymbol{v}_T). \tag{1.45}$$

Since the solution is unique and $-\boldsymbol{v}_{T^C} \neq \boldsymbol{v}_T$ it must hold $\|\boldsymbol{v}_T\|_1 < \|\boldsymbol{v}_{T^C}\|_1$ which means that $\boldsymbol{A}$ has NSP of order k.
$\Leftarrow$: Let $\boldsymbol{x} \in \mathbb{R}^M$ be a k-sparse vector with $\text{supp}(\boldsymbol{x}) = T$. We have to show that this vector is the unique solution of (1.43). That means, that if $\boldsymbol{z} \in \mathbb{R}^M$ is also a solution of (1.43) then $\|\boldsymbol{x}\|_1 < \|\boldsymbol{z}\|_1$ for every such $\boldsymbol{z}$. Using the fact that both $\boldsymbol{x}, \boldsymbol{z}$ are solutions $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} = \boldsymbol{A}\boldsymbol{z}$, we get $(\boldsymbol{x} - \boldsymbol{z}) \in \ker\boldsymbol{A}\backslash\{\boldsymbol{0}\}$. The implication then concludes from the inequality

$$\|\boldsymbol{x}\|_1 \leq \|\boldsymbol{x} - \boldsymbol{z}_T\|_1 + \|\boldsymbol{z}_T\|_1 = \|(\boldsymbol{x} - \boldsymbol{z})_T\|_1 + \|\boldsymbol{z}_T\| < \|(\boldsymbol{x} - \boldsymbol{z})_{T^C}\|_1 + \|\boldsymbol{z}_T\|_1 = \|\boldsymbol{z}_{T^C}\|_1 + \|\boldsymbol{z}_T\|_1 = \|\boldsymbol{z}\|_1,$$

where we used (in order) the triangle inequality, the k-sparsity of $\boldsymbol{x}$, the NSP of $\boldsymbol{A}$, the k-sparsity of $\boldsymbol{x}$ and then the additivy of ℓ_1 norm. □

Remark 6. The theorem above implies that the solutions of the problems (1.42) and (1.43) can overlap. If $\hat{\boldsymbol{x}}$ is a solution of (1.42) and $\boldsymbol{x}$ is a k-sparse solution of (1.43) with $\boldsymbol{A}$ with NSP of order k then $\|\hat{\boldsymbol{x}}\|_0 \leq \|\boldsymbol{x}\|_0 \leq k$. Then Theorem 4 says that $\hat{\boldsymbol{x}}$ is a solution of (1.43) and $\hat{\boldsymbol{x}} = \boldsymbol{x}$. In other words, there is a class of matrices for which the problem of ℓ_0 minimization can be solved in polynomial time and that is done using the Basis Pursuit problem since the solutions coincide.

Remark 7. It is easy to see from the definition of NSP that $\hat{\boldsymbol{A}} = \boldsymbol{M}\boldsymbol{A}$ where $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ has NSP of order k and $\boldsymbol{M} \in \mathbb{R}^{N\times N}$ is full rank then $\hat{\boldsymbol{A}}$ also has NSP of order k.

Restricted Isometry Property

The Null Space Property is rather impractical because finding matrices which satisfy the condition is difficult. Therefore we define a stronger property of $\boldsymbol{A}$ which implies NSP.

Definition 10 (Restricted Isometry Property)**.** Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ and $k \in \hat{M}$. The restricted isometry constant $\delta_k = \delta_k(\boldsymbol{A})$ of $\boldsymbol{A}$ of order k is the smallest $\delta \geq 0$ such that

$$(1 - \delta)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq (1 + \delta)\|\boldsymbol{x}\|_2^2, \ \boldsymbol{x} \in \mathbb{R}_k^M. \tag{1.46}$$

We say $\boldsymbol{A}$ satisfies the Restricted Isometry Property (RIP) of order k with the constant δ_k if $\delta_k < 1$.

Remark 8. The condition (1.46) means that $\boldsymbol{A}$ is almost isometrical on the set of k-sparse vectors. The following theorem says that RIP implies NSP.

Theorem 4 (RIP ⇒ NSP)**.** Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ and $k \in \mathbb{N}$ such that $k \leq M/2$. Then

$$\delta_{2k}(\boldsymbol{A}) < 1/3 \Rightarrow \boldsymbol{A} \text{ has NSP of order } k.$$

Proof. Let $\boldsymbol{v} \in ker\boldsymbol{A}$ and $T \subset \hat{M}$ with $|T| \leq k$. We will prove the inequality

$$\|\boldsymbol{v}_T\|_2 \leq \frac{\delta_{2k}}{1-\delta_k} \cdot \frac{\|\boldsymbol{v}\|_1}{\sqrt{k}}, \tag{1.47}$$

because then under the assumption $\delta_k \leq \delta_{2k} < 1/3$, we get $\|\boldsymbol{v}_T\|_1 \leq \sqrt{k}\|\boldsymbol{v}_T\|_2 < \|\boldsymbol{v}\|_1/2$ where the Hölder's inequality gives the first inenquality and (1.47) gives the sharp inequality which combined with the note in Remark 6 gives NSP of order k.

First, we will prove a small useful statement:

$$\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^M_k \text{ such that } \operatorname{supp}(\boldsymbol{x}) \bigcap \operatorname{supp}(\boldsymbol{z}) = \varnothing \text{ and } \boldsymbol{A} \text{ has NSP of order } 2k \Rightarrow |\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{A}\boldsymbol{z}\rangle| \leq \delta_{2k}\|\boldsymbol{x}\|_2\|\boldsymbol{z}\|_2. \tag{1.48}$$

Proof of the statement. It is easy to consider the validity of the following implication

$$\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^M_k, \|\boldsymbol{x}\|_2 = \|\boldsymbol{z}\|_2 = 1 \text{ such that } \operatorname{supp}(\boldsymbol{x}) \bigcap \operatorname{supp}(\boldsymbol{z}) = \varnothing \Rightarrow \boldsymbol{x} \pm \boldsymbol{z} \in \mathbb{R}^M_{2k} \text{ and } \|\boldsymbol{x} \pm \boldsymbol{z}\|_2^2 = 2. \tag{1.49}$$

Taking the RIP of $\boldsymbol{A}$ for $\boldsymbol{x} \pm \boldsymbol{z}$

$$2(1-\delta_{2k}) \leq \|\boldsymbol{A}(\boldsymbol{x} \pm \boldsymbol{z})\|_2^2 \leq 2(1+\delta_{2k}),$$

and combining it with the polarization identity gives

$$|\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{A}\boldsymbol{z}\rangle| = \frac{1}{4}\Big|\|\boldsymbol{A}(\boldsymbol{x}+\boldsymbol{z})\|_2^2 - \|\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{z})\|_2^2\Big| \leq \frac{1}{4}\Big|2(1+\delta_{2k}) - 2(1-\delta_{2k})\Big| \leq \delta_{2k}.$$

Finally, we plug in $\tilde{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2}$ and $\tilde{\boldsymbol{z}} = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|_2}$ and get the statement $|\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{A}\boldsymbol{z}\rangle| \leq \delta_{2k}\|\boldsymbol{x}\|_2\|\boldsymbol{z}\|_2$.

Proof of the theorem. Let $\boldsymbol{v} \in \ker\boldsymbol{A}$ and let us consider a nonincreasing rearrangement of $\boldsymbol{v}$, that is $|v_{\pi(1)}| \geq |v_{\pi(2)}| \geq \cdots \geq |v_{\pi(M)}|$ for some convenient permutation π. Then we slice the rearrangement into sets of size k (the last set can be smaller):

$$T_0 = \{1, \ldots, k\},\ T_1 = \{k+1, \ldots, 2k\},\ T_2 = \{2k+1, \ldots, 3k\}, \text{ etc.}$$

Then

$$\boldsymbol{A}\boldsymbol{v}_{T_0} = \boldsymbol{A}(-\boldsymbol{v}_{T_1} - \boldsymbol{v}_{T_2} - \ldots). \tag{1.50}$$

We construct an estimate

$$\|\boldsymbol{v}_{T_0}\|_2^2 \leq \frac{\|\boldsymbol{A}\boldsymbol{v}_{T_0}\|_2^2}{1-\delta_k} = \frac{1}{1-\delta_k}\langle \boldsymbol{A}\boldsymbol{v}_{T_0}, \boldsymbol{A}(-\boldsymbol{v}_{T_1}) + \boldsymbol{A}(-\boldsymbol{v}_{T_2}) + \ldots\rangle = \frac{1}{1-\delta_k}\sum_{j\geq 1}\langle \boldsymbol{A}\boldsymbol{v}_{T_0}, \boldsymbol{A}(-\boldsymbol{v}_{T_j})\rangle \leq$$

$$\leq \frac{1}{1-\delta_k}\sum_{j\geq 1}\langle \boldsymbol{A}\boldsymbol{v}_{T_0}, \boldsymbol{A}(-\boldsymbol{v}_{T_j})\rangle \leq \frac{1}{1-\delta_k}\delta_{2k}\sum_{j\geq 1}\|\boldsymbol{v}_{T_0}\|_2\|\boldsymbol{v}_{T_j}\|_2,$$

where we applied the definition of ℓ_2 norm through scalar product together with (1.50) in the first equality and the proved statement (1.48) in the last inequality. Dividing the inequality by $\|\boldsymbol{v}_{T_0}\|_2 \neq 0$ finally gives

$$\|\boldsymbol{v}_{T_0}\|_2 \leq \frac{\delta_{2k}}{1-\delta_k}\sum_{j\geq 1}\|\boldsymbol{v}_{T_j}\|_2. \tag{1.51}$$

The proof is finished through the following chain of inequalities

$$\sum_{j\geq 1}\|\boldsymbol{v}_{T_j}\|_2 = \sum_{j\geq 1}\Big(\sum_{l\in T_j}|\boldsymbol{v}_l|^2\Big)^{1/2} \leq \sum_{j\geq 1}\Big(k\max_{l\in T_j}|\boldsymbol{v}_l|^2\Big)^{1/2} = \sum_{j\geq 1}\sqrt{k}\max_{l\in T_j}|\boldsymbol{v}_l| \leq$$

$$\leq \sum_{j\geq 1}\sqrt{k}\min_{l\in T_{j-1}}|\boldsymbol{v}_l| \leq \sum_{j\geq 1}\sqrt{k}\Big(\sum_{l\in T_{j-1}}\frac{1}{k}|\boldsymbol{v}_l|\Big) = \sum_{j\geq 1}\frac{\|\boldsymbol{v}_{T_{j-1}}\|_1}{\sqrt{k}} = \frac{\|\boldsymbol{v}\|_1}{\sqrt{k}}. \tag{1.52}$$

Plugging the above result into (1.51) gives the inequality (1.47) since $T = T_0$. □

Corollary 1. Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ and $k \in \mathbb{N}$ such that $k \leq M/2$. Then,

$$\delta_{2k} < 1/3 \Rightarrow \text{every } k\text{-sparse vector } \boldsymbol{x} \text{ is the unique solution of (1.43).}$$

Proof. Combining the Theorem 2 and 3 immediately gives the statement. □

Remark 9. In a reductionist manner, we can symbolically showcase the development of the outlined theory as follows

$$\text{RIP} \Rightarrow \text{NSP} \Rightarrow \ell_1 \text{ solution} \Rightarrow \ell_0 \text{ solution.}$$

Stability and Robustness

So far, we assumed $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ but that is not the case for a real setting. The input will always be influenced by errors $\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}$. We will also want to recover vectors or their approximations which are not exactly sparse. We will see that RIP is still a sufficient property for recovery of a solution even in settings with errors.

Definition 11 (Modified Basis Pursuit)**.** Let $\boldsymbol{x} \in \mathbb{R}^M$, $\boldsymbol{A} \in \mathbb{R}^{N\times M}$ be known and $\boldsymbol{y} \in \mathbb{R}^N$ be known. Let $\eta \geq 0$. Then we define

$$\hat{\boldsymbol{x}} = \min_{\boldsymbol{x}\in\mathbb{R}^M}\|\boldsymbol{x}\|_1 \ \textit{subject to}\ \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2 \leq \eta. \tag{1.53}$$

Theorem 5. Let $\boldsymbol{A} \in \mathbb{R}^{N\times M}$, $\boldsymbol{x} \in \mathbb{R}^M$ and $\boldsymbol{y} \in \mathbb{R}^N$. Let $\delta_{2k} < \sqrt{2} - 1$ and $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2 \leq \eta$. Then the solution $\hat{\boldsymbol{x}} \in \mathbb{R}^M$ of (1.53) satisfies

$$\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 \leq \frac{C\sigma_k(\boldsymbol{x})_1}{\sqrt{k}} + D\eta, \tag{1.54}$$

where $C, D > 0$ are constants.

Proof. Let us define $\boldsymbol{h} := \hat{\boldsymbol{x}} - \boldsymbol{x}$. For the purposes of the proof, we also define an index set $T_0 \subset \hat{N}$ which indicates the k largest absolutes values of inputs of $\boldsymbol{x}$. We define $T_1 \subset T_0^C$ as the indices of k largest absolute values of inputs of $\boldsymbol{h}_{T_0^C}$, then $T_2 \subset (T_1 \cup T_0)^C$ as the indices of k largest absolute values of inputs of $\boldsymbol{h}_{(T_1\cup T_0)^C}$ and so on.

We will construct a series of inequalities which will give the sought inequality together with a few previous results.

1. Noticing $\hat{\boldsymbol{x}}$ is a solution of (1.53) and also noticing $\boldsymbol{x}$ satisfies the constraint from the definition from (1.53) gives from triangle inequality

$$\|\boldsymbol{A}\boldsymbol{h}\|_2 = \|\boldsymbol{A}(\boldsymbol{x} - \hat{\boldsymbol{x}})\|_2 \leq \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}} + \boldsymbol{y}\|_2 \leq \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|_2 + \|\boldsymbol{A}\hat{\boldsymbol{x}} - \boldsymbol{y}\|_2 \leq 2\eta, \tag{1.55}$$

therefore we get $\|\boldsymbol{A}\boldsymbol{h}\|_2 \leq 2\eta$.

2. Noticing $\hat{\boldsymbol{x}}$ is a solution of (1.53) also gives $\|\hat{\boldsymbol{x}}\|_1 = \|\boldsymbol{x}+\boldsymbol{h}\|_1 \leq \|\boldsymbol{x}\|_1$. Now, we show the following chain of inequalities

$$\begin{aligned}
\|\boldsymbol{h}_{T_0^C}\|_1 &= \|(\boldsymbol{x}+\boldsymbol{h})_{T_0^C} - \boldsymbol{x}_{T_0^C}\|_1 + \|(\boldsymbol{x}+\boldsymbol{h})_{T_0} - \boldsymbol{h}_{T_0}\|_1 - \|\boldsymbol{x}_{T_0}\|_1 \leq \\
&\leq \|(\boldsymbol{x}+\boldsymbol{h})_{T_0^C}\|_1 + \|\boldsymbol{x}_{T_0^C}\|_1 + \|(\boldsymbol{x}+\boldsymbol{h})_{T_0}\|_1 + \|\boldsymbol{h}_{T_0}\|_1 - \|\boldsymbol{x}_{T_0}\|_1 = \\
&= \|\boldsymbol{h}_{T_0}\|_1 + \|\boldsymbol{x}_{T_0^C}\|_1 - \|\boldsymbol{x}_{T_0}\|_1 + \|(\boldsymbol{x}+\boldsymbol{h})\|_1 \leq \\
&\leq \|\boldsymbol{h}_{T_0}\|_1 + \|\boldsymbol{x}_{T_0^C}\|_1 - \|\boldsymbol{x}_{T_0}\|_1 + \|\boldsymbol{x}\|_1 = \\
&= \|\boldsymbol{h}_{T_0}\|_1 + 2\|\boldsymbol{x}_{T_0^C}\|_1 \leq \\
&\leq \sqrt{k}\|\boldsymbol{h}_{T_0}\|_2 + 2\sigma_k(\boldsymbol{x})_1,
\end{aligned} \tag{1.56}$$

where we used the Hölder's inequality in the first term and the definition of the best k-term approximation from (1.41) in the second term to show the last inequality.

Using the approach as in (1.52) with a simple shift of indices gives

$$\sum_{j\geq 2}\|\boldsymbol{h}_{T_j}\|_2 \leq \frac{\|\boldsymbol{h}_{T_0^C}\|_1}{\sqrt{k}}. \tag{1.57}$$

Finally, combining the two results (1.56) and (1.57) gives the second needed inequality

$$\sum_{j\geq 2}\|\boldsymbol{h}_{T_j}\|_2 \leq \|\boldsymbol{h}_{T_0}\|_2 + \frac{2\sigma_k(\boldsymbol{x})_1}{\sqrt{k}}. \tag{1.58}$$

3. A simple inequality

$$\|\boldsymbol{h}_{T_0}\|_2 + \|\boldsymbol{h}_{T_1}\|_2 \leq \sqrt{2}\|\boldsymbol{h}_{T_0\cup T_1}\|_2, \tag{1.59}$$

easily comes from the simple fact $\frac{a+b}{2} \leq \sqrt{\frac{a^2+b^2}{2}}$ for $a, b \geq 0$.

Combining the triangle inequality, the proven statement (1.48), the definition of RIP (1.46) and the three results (1.55), (1.56) and (1.59) give us

$$\begin{aligned}
(1-\delta_{2k})\|\boldsymbol{h}_{T_0\cup T_1}\|_2^2 &\leq \|A\boldsymbol{h}_{T_0\cup T_1}\|_2^2 = \langle \boldsymbol{A}\boldsymbol{h}_{T_0\cup T_1}, \boldsymbol{A}\boldsymbol{h}\rangle - \langle \boldsymbol{A}\boldsymbol{h}_{T_0\cup T_1}, \sum_{j\geq 2}\boldsymbol{A}\boldsymbol{h}_{T_j}\rangle \leq \\
&\leq \|\boldsymbol{A}\boldsymbol{h}_{T_0\cup T_1}\|_2\|\boldsymbol{A}\boldsymbol{h}\|_2 + \sum_{j\geq 2}|\langle \boldsymbol{A}\boldsymbol{h}_{T_0}, \boldsymbol{A}\boldsymbol{h}_{T_j}\rangle| + \sum_{j\geq 2}|\langle \boldsymbol{A}\boldsymbol{h}_{T_1}, \boldsymbol{A}\boldsymbol{h}_{T_j}\rangle| \leq \\
&\leq 2\eta\sqrt{1+\delta_{2k}}\|\boldsymbol{h}_{T_0\cup T_1}\|_2 + \delta_{2k}(\|\boldsymbol{h}_{T_0}\|_2 + \|\boldsymbol{h}_{T_1}\|_2)\sum_{j\geq 2}\|\boldsymbol{h}_{T_j}\|_2 \leq \\
&\leq \|\boldsymbol{h}_{T_0\cup T_1}\|_2\Big(2\eta\sqrt{1+\delta_{2k}} + \sqrt{2}\delta_{2k}\|\boldsymbol{h}_{T_0}\|_2 + \frac{2\sqrt{2}\delta_{2k}\sigma_k(\boldsymbol{x})_1}{\sqrt{k}}\Big)
\end{aligned} \tag{1.60}$$

Further, we alter the result in a few steps

1. Divide by $(1-\delta_{2k})\|\boldsymbol{h}_{T_0\cup T_1}\|_2$:

$$\|\boldsymbol{h}_{T_0\cup T_1}\|_2 \leq \frac{2\eta\sqrt{1+\delta_{2k}} + \sqrt{2}\delta_{2k}\|\boldsymbol{h}_{T_0}\|_2 + \frac{2\sqrt{2}\delta_{2k}\sigma_k(\boldsymbol{x})_1}{\sqrt{k}}}{1-\delta_{2k}} \tag{1.61}$$

2. Use the trivial observation $\|\boldsymbol{h}_{T_0}\|_2 \leq \|\boldsymbol{h}_{T_0 \cup T_1}\|_2$ and subtract the middle term:

$$\|\boldsymbol{h}_{T_0 \cup T_1}\|_2 - \frac{\sqrt{2}\delta_{2k}\|\boldsymbol{h}_{T_0 \cup T_1}\|_2}{1-\delta_{2k}} \leq \frac{2\eta\sqrt{1+\delta_{2k}} + \frac{2\sqrt{2}\delta_{2k}\sigma_k(\boldsymbol{x})_1}{\sqrt{k}}}{1-\delta_{2k}} \tag{1.62}$$

3. Define constants $\alpha = \frac{2\sqrt{1+\delta_{2k}}}{1-\delta_{2k}}$ and $\rho = \frac{\sqrt{2}\delta_{2k}}{1-\delta_{2k}}$:

$$\|\boldsymbol{h}_{T_0 \cup T_1}\|_2(1-\rho) \leq \alpha\eta + \rho\frac{2\sigma_k(\boldsymbol{x})_1}{\sqrt{k}} \tag{1.63}$$

Finally, we can prove the desired inequality using (1.58), (1.63) and $\|\boldsymbol{h}_{T_0}\|_2 \leq \|\boldsymbol{h}_{T_0 \cup T_1}\|_2$:

$$\begin{aligned} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 \leq \|\boldsymbol{h}\|_2 &\leq \|\boldsymbol{h}_{(T_0 \cup T_1)^C}\|_2 + \|\boldsymbol{h}_{T_0 \cup T_1}\|_2 \leq \\ &\leq \sum_{j \geq 2}\|\boldsymbol{h}_{T_j}\|_2 + \|\boldsymbol{h}_{T_0 \cup T_1}\|_2 \leq \\ &\leq \|\boldsymbol{h}_{T_0}\|_2 + \frac{2\sigma_k(\boldsymbol{x})_1}{\sqrt{k}} + \|\boldsymbol{h}_{T_0 \cup T_1}\|_2 \leq \\ &\leq \frac{2\sigma_k(\boldsymbol{x})_1}{\sqrt{k}} + 2\|\boldsymbol{h}_{T_0 \cup T_1}\|_2 \leq \\ &\leq \frac{C\sigma_k(\boldsymbol{x})_1}{\sqrt{k}} + D\eta, \end{aligned} \tag{1.64}$$

and we defined $C = \frac{2}{1-\rho}$ and $D = \frac{2(1+\rho)}{1-\rho}$. □

Bound Constraints

We will show an inequality which will provide a relationship between the number of measurements needed for a successful recovery of desired sparse solutions. The bottom line obviously is the need for at least $M \geq k$ measurements if we want recover k-sparse solutions. The following theorem shows a more precise relationship between k and M.

Theorem 6 (Constraint of the Matrix)**.** Let $k \leq M \leq N$, $k, M, N \in \mathbb{N}$ and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$. Let $\Delta : \mathbb{R}^M \to \mathbb{R}^N$ be an arbitrary function which for some constant $C > 0$ fulfills

$$\|\boldsymbol{x} - \Delta(\boldsymbol{A}\boldsymbol{x})\|_2 \leq C\frac{\sigma_k(\boldsymbol{x})_1}{\sqrt{k}}, \forall \boldsymbol{x} \in \mathbb{R}^N. \tag{1.65}$$

Then the dimensions of the matrix and the sparsity of the recovery satisfies

$$M \geq \tilde{C}k\ln\left(\frac{\mathrm{e}N}{k}\right), \tag{1.66}$$

where $\tilde{C}$ depends only on C (e is Euler's number).

Proof. Can be found in [4].

The inequality (1.66) gives guidance on the dimensions of $\boldsymbol{A}$. If the number of measurements falls below the bound in (1.66), there is no stable and robust recovery of desired sparse vectors.

1.1.5 Deep Feedforward Networks

Artificial neural networks have been in the spotlight ever since deep learning became truly impactful not even 20 years ago. The applications and innovations of artificial neural networks have been accelerating since then. This section describes the basic principles of the feed-forward network model for regression. Neural networks consist of many possible architectures (convolutional neural networks, LSTMs, etc.) and we concern ourselves only with the deep feedforward network architecture because it is the one used in this work but the principles outlined here are applicable and universal to most other networks. The sources of this section are [12] and [1].

Network Architecture

In general, any network where information propagates only forward is a feedforward network. Every network is composed of units usually called neurons. Each neuron consists of a vector of parameters called weights and a bias. Neurons are organized into groups called layers. The first layer is called the input layer, the last layer is called the output layer. All layers in between are the hidden layers. Let $\boldsymbol{x}$ be an vector of regressors. The structure of the network with n hidden layers is given by the following

$$\begin{aligned} {}^{(1)}\boldsymbol{h} &= {}^{(1)}f\Big({}^{(1)}\boldsymbol{W}^T\boldsymbol{x} + {}^{(1)}\boldsymbol{b}\Big), \\ {}^{(2)}\boldsymbol{h} &= {}^{(2)}f\Big({}^{(2)}\boldsymbol{W}^{T\,(1)}\boldsymbol{h} + {}^{(2)}\boldsymbol{b}\Big), \\ &\vdots \\ \hat{\boldsymbol{y}} = {}^{(n)}\boldsymbol{h} &= {}^{(n)}f\Big({}^{(n)}\boldsymbol{W}^{T\,(n-1)}\boldsymbol{h} + {}^{(n)}\boldsymbol{b}\Big). \end{aligned} \tag{1.67}$$

The weights of neurons of a layer l are the columns of ${}^{(l)}\boldsymbol{W}$ denoted $\boldsymbol{w}_{\bullet j}$, the biases of a layer l are vectors ${}^{(l)}\boldsymbol{b}$. For convenience purposes, the set of all weights and biases will be denoted $\boldsymbol{\theta} = \Big({}^{(1)}\boldsymbol{W}, {}^{(1)}\boldsymbol{b}, {}^{(2)}\boldsymbol{W}, {}^{(2)}\boldsymbol{b}, \ldots, {}^{(n)}\boldsymbol{W}, {}^{(n)}\boldsymbol{b}\Big)$. The weights and biases of the layer l define an affine transformation which is then further fed into a function ${}^{(l)}f$ which is called the activation function. We require this function to be nonlinear, otherwise if ${}^{(l)}f(x) = x$ for all layers $l \in \hat{n}$, the architecture in (1.67) simplifies into one affine transformation. The activation ${}^{(n)}f$ is tailored to provide a meaningful prediction on output and is usually different from all other activation functions. The output of the network is in general a vector of predictions $\hat{\boldsymbol{y}}$. If all neurons of previous layer connect to all neurons of the next layer, the network is sometimes referred to as multilayer perceptron (see Figure 1.1).

Of course, we are free to define a network in a manner where certain connections are omitted. This is done through defining the weights ${}^{(l)}\boldsymbol{W}$ of the network to correspond to the desired architecture. The ideal amount of neurons in each hidden layer and the number of hidden layers is not a priori known and must be determined through trial and error.

The weights of (1.67) are usually initialized and the learning process is to optimize them so that the network gives a meaningful prediction. The techniques used to optimize the values of the weights and biases are the topic of the following text. We also present practices which help the network generalize and achieve good results.

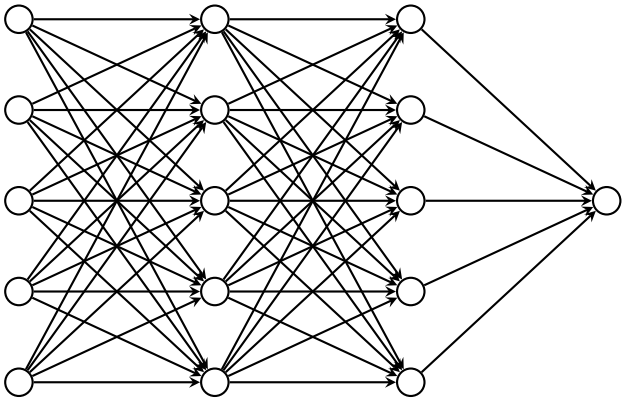


Figure 1.1: An example of a deep feedforward network called the multilayer perceptron. The information flows only in one direction as is symbolized by the connections with arrows. The network has an input layer with 5 neurons (rightmost), 2 hidden layers with 5 neurons each and 1 output neuron in the output layer. Each neuron of the previous layer connects with every neuron of the next layer

Activation Functions

Forward Propagation and Back-Propagation Algorithm

The task of finding a set of weights and biases of a network to give good predictions is done through two algorithms called forward propagation and back-propagation. They require some measure of the error the network makes on predictions. This creates the need to define a loss function of the network. For a single datapoint we put $L(\hat{\boldsymbol{y}}, \boldsymbol{y})$ is the loss of the

Optimizers

Stochastic Gradient Descent

Adam

Regularization

Batch Normalization

Dropout

Weights and Bias Initialization

1.2 Data Transformation Methods

In both traditional statistical inference (LASSO, principal component analysis, etc.) and machine learning, it is either advantages or even required to perform some kind of transformation of the data. The most common purpose is to improve the performance of the model. The given transformation can have a physical meaning or interpretability, the motivation to perform such transformation can even be initiated by the context of the underlying problem.

1.2.1 Feature Standardization

The purpose of data standardization is to remove the difference of scale between features of the data. The standardization used in this work is fairly common and has the following form

$$\boldsymbol{x}'_{\bullet i} = \frac{\boldsymbol{x}_{\bullet i} - \bar{\boldsymbol{x}}_{\bullet i}}{\sigma_i}, \tag{1.68}$$

where $\bar{\boldsymbol{x}}_{\bullet i}$ is the mean of the column $\boldsymbol{x}_{\bullet i}$ of the matrix of regressors and σ_i is the standard deviation of said column. A special case of standardization is mean-centering which we get when σ_i is set to one for all columns. The idea of standardization comes from the assumption that the data was sampled from standard normal distribution with zero mean and unit variance.

1.2.2 Feature Normalization

Feature normalization performs scaling of features to interval $\langle 0, 1\rangle$. The transformation for ith feature (column) of a matrix is given by

$$\boldsymbol{x}''_{\bullet i} = \frac{\boldsymbol{x}_{\bullet i} - \min_j(\boldsymbol{x}_{ji})}{\max_j(\boldsymbol{x}_{ji}) - \min_j(\boldsymbol{x}_{ji})}, \tag{1.69}$$

where we define minimum of ith feature as $\min_j(\boldsymbol{x}_{ji})$ and maximum of feature ith as $\max_j(\boldsymbol{x}_{ji})$. Feature normalization can improve the performance of the model when the features are on different scale by orders of magnitude. This usually happens when dealing with physical problems where variables have different units and therefore the method used is more sensitive to some features than it should be.

1.3 Model Validation Methods

It is required to have the ability to compare performances of models to choose the best performer. The relevant methods of hyperparameter tuning are presented below as well as the error metrics used for scoring of the competing models.

1.3.1 Error Metrics

The prediction quality of a model is measured by functions which determine the error of the model's prediction capabilities. We can calculate the train error which is the prediction error of the train data. Much more important is the behavior of the model on the data which the model was not trained on. This set of datapoints is called the test data.

We shall use the following notation: N is the number test datapoints, y_i is the actual value of the property we want to model and $\hat{y}_i$ is the predicted value.

In the Nomad2018 Predicting Transparent Conductors Kaggle competition, the metric used was Root Mean Squared Logarithmic Error (RMSLE) [36]

$$RMSLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big(\ln(\hat{y}_i + 1) - \ln(y_i + 1)\Big)^2}. \tag{1.70}$$

The reason to use such metric is that there are two properties being modeled. One is on average 10 times bigger than the other and the natural logarithms in (1.70) reasonably erases this scale difference. The average of the two results is then taken and submitted into the competition. We will not use such

practices in this work but we will report the RMSLE test error because it will allow comparison of our results with the relevant literature.

There are many types of commonly used error or scoring metrics one can use to measure the performance of a model. We choose to list the ones we will use and these are Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Maximum Absolute Error (MaxAE).

$$\begin{aligned} MSE &= \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_y)^2, \\ RMSE &= \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_y)^2}, \\ MAE &= \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_y|, \\ MaxAE &= \max_{i\in\hat{N}}|\hat{y}_i - y_y|. \end{aligned} \tag{1.71}$$

We prefer RMSE over MSE because it has the same units as the modeled properties and it makes the physical interpretability of the performance much easier.

1.3.2 Cross Validation

Cross validation is the process of evaluating of performance of a model. It can also be used to optimize hyperparameters of a model. It starts with splitting the dataset into a training set and validation set. We train the model on the train set and then evaluate the perfomance on the validation set. The method can be formulated in terms of the percentage of the whole dataset we choose as the validation set (e.g. Leave 10% Out Cross Validation). We perform random splits multiple times until a good enough statistics is available and report the mean of cross validation errors. However, this method overestimates the error because we fit on a subset of the whole dataset.

Another method of cross validation is the Leave One Out Cross Validation (LOOCV). For N datapoints, we perform N model evaluations and every datapoint is the validation set once. The cross validation error is then given as the mean of the errors on the one datapoint which makes up the validation set. The advantage of this method is that it is not random at all and also the model's performance stays as close as to the performance as if we trained on all the points. However, this method can get very computationally expensive for large datasets which makes it perfectly suitable for small datasets.

A computationally more advantageous is the k-fold cross validation (k-fold CV) where we split the data into k parts where $k-1$ are used for training and the last part is the validation set. We cycle through all k combinations and the k-fold cross validation error is given as the mean of the k errors reported on the validation sets. We typically use $k = 5$ or $k = 10$. This cross validation technique is much less computationally demanding that LOOCV.

If the data we use are structured into groups in some sense, we do not want to leak some datapoints from one group in the validation set into the train set because we would get a spurious result. Therefore, we have to perform the group k-fold cross validation (group k-fold CV). The main difference from the previous k-fold CV is that we split the dataset into folds based on the groups. We also should make sure the amount of groups in each fold does not vary too much and that the overall amount of datapoints in each fold does not vary too much as well.

1.3.3 Hyperparameter Tuning

The process of cross validation from the previous section is commonly used to optimize hyperparameters of models. The process starts with defining a set of values for every hyperparameter and then creating a grid (Cartesian product) from these sets. All the combinations of hyperparameters are then evaluated in the cross validation procedure and the best performing combination of hyperparameters is chosen. This method of hyperparameter tuning is called grid search. The method can be computationally very expensive for large datasets or many hyperparameters. Usually, we define the sets of hyperparameters on a logarithmic scale and if need be, define a finer grid later on.

Chapter 2

Feature Engineering

The regressive power of datasets can be enhanced with the implementation of functions which make the job of extracting the information from the data easier. Such functions are usually called features and the process of creating such functions is called feature engineering. In our case, the main source of inspiration for creation of features is the physical description itself but it is not the rule.

In this chapter, we will briefly explain the source of the available data and build understanding of the underlying physical problem.

2.1 Density Functional Theory Data

Fundamentally, there are two possible ways how to obtain data in physics. Either the physical phenomenon is measured in a laboratory or it is modeled and the physical quantities are calculated. Our case is the latter and a brief overview of the way the data are gained is presented. The underlying theory was studied in the previous work [6].

The central object of quantum mechanics is the wave function but this quantity of quantum mechanics is unwieldy for practical calculations even though it fully describes the physical system. An equivalent formulation to quantum mechanics is called density functional theory (DFT) and uses as its central quantity electron density while allowing for approximation of the many-body problem of quantum mechanics on different levels of complexity. Software packages implementing DFT calculation procedures are the source of all the data used in this work. These software implementations are numerical solvers which converge using an iterative procedure.

2.2 Physical Background

The relevant physical knowledge to understand the material representations is outlined here. The presentation does not go into great depth to explain the crystallographic and physical phenomena and focuses exclusively on concepts relevant to our machine learning regression problem. The conceptualizations and representations of the material data differ based on the application - unfortunately, there does not appear to be a grand solution to the problem of property prediction in solid state physics.

Non-SI units are used in this work. The unit of length is the Ångström and converts as $1\text{Å} = 10^{-10}m$ and the unit of energy is electronvolt which converts as $1eV = 1.602 \cdot 10^{-19}J$.

2.2.1 Crystalline Structure

The structure of a crystalline material is defined by 3 linearly independent vectors $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}^3$ called lattice vectors. We construct

$$\boldsymbol{A} = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) \in \mathbb{R}^{3x3}, \tag{2.1}$$

a matrix of the lattice vectors. In space, the lattice vectors form what is known as Bravais lattice.

The position of an atom of a crystalline material is a vector of Cartesian coordinates $\boldsymbol{R} = (X, Y, Z)^T$. We also define reduced coordinates as $\boldsymbol{r} = (x, y, z)^T$, where $x, y, z \in \langle 0, 1 \rangle$. The reduced coordinates relate to their corresponding Cartesian coordinates as

$$\boldsymbol{R} = \boldsymbol{A}\boldsymbol{r} = \boldsymbol{a}x + \boldsymbol{b}y + \boldsymbol{c}z. \tag{2.2}$$

Therefore, the reduced coordinates hold the information of the position of the atom in the parallelepiped described by the lattice vectors. This volume outlined by the lattice vectors in space is called the unit cell. It is clear that the lattice vectors allow translational symmetry of the unit cell through space. The translation is described as $k\boldsymbol{a} + l\boldsymbol{b} + m\boldsymbol{c}$ where $k, l, m \in \mathbb{Z}$ and such structure covers the whole $\mathbb{R}^3$ space. Assuming there are N atoms in the unit cell and each of them has its atomic number from the set of all atomic number $\mathbb{I} = \{1, \ldots, 118\}$, the N atoms and the lattice vectors completely describe the positional information of the crystalline material. The volume of the unit cell is the aforementioned parallelepiped and is given by

$$V_{cell} = |\det \boldsymbol{A}|. \tag{2.3}$$

The whole situation is showed on the example of a crystal in Figure 2.1.

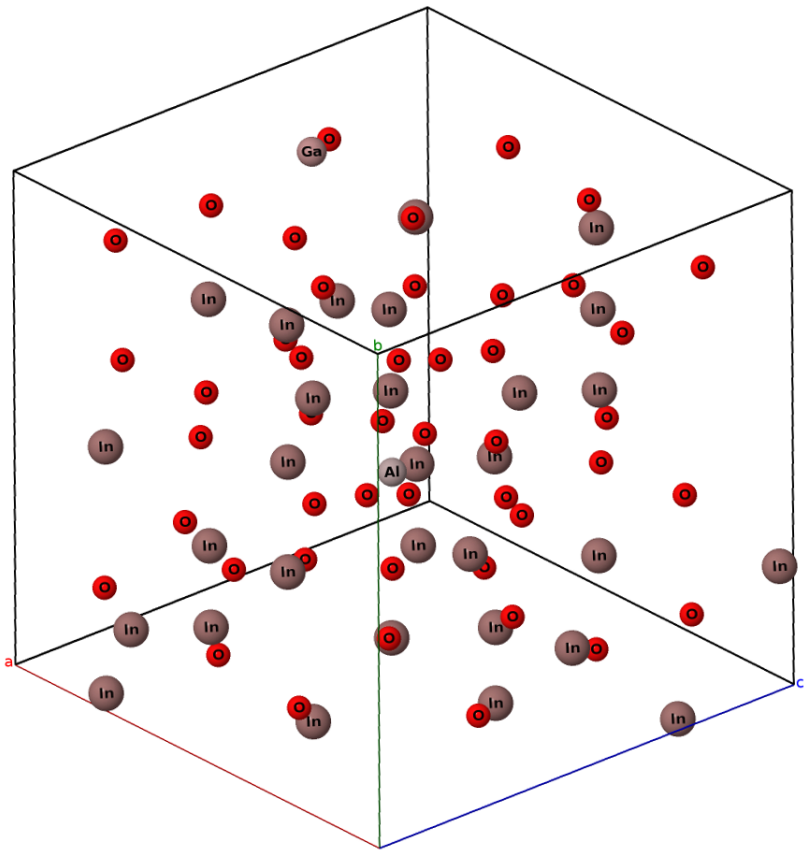


Figure 2.1: Example of unit cell and atoms. There are 80 atoms in the unit cell: 48 oxygens (red), 30 indiums (brown), 1 aluminium (grey) and 1 gallium (pink). The unit cell is outlined by the three lattice vectors [20]

For every crystalline material, there is a set of invariant transformation - a symmetry group - which is called space group and it completely describes the symmetry of the crystal. There are 230 possible space groups for crystalline materials. Interestingly, there are no known crystals for 80 space group and we can choose about 30 most common and important ones which represent the vast majority of the known crystals [18].

2.2.2 Coordination Numbers and Ionic Radii

In the context of a crystal with positively charged atoms (cations) and negatively charged atoms (anions), the amount of ions with opposite charge closest to the given atom is called the coordination number and the geometric shape constructed from the connections of centers of these atoms is called the coordination geometry. It is possible to have many different coordination geometries for a given coordination number. For coordination number equal to 4, the coordination geometry can be tetrahedron or square, for coordination number equal to 6, octahedron, trigonal prism, etc. The free space among the atoms decreases with higher coordination numbers [18].

Ionic radius is the idealized state of an cation or anion where we assume a rigid boundary of the ion based on its charge. Shannon also assumes the ionic radius changes with the coordination number. The Shannon ionic radii are calculated and tabulated values. We report the values used in the Nomad2018

Predicting Transparent Conductors Kaggle competition (coordination VI for metals and coordination II for oxygen) which were calculated by Shannon [33]. All Shannon's values for the four elements making up the studied crystals are in Table 2.1.

In Å	Al^{3+}	Ga^{3+}	In^{3+}	O^{2-}
II	-	-	-	1.35
III	-	-	-	1.36
IV	0.39	0.47	0.62	1.38
V	0.48	0.55	-	-
VI	0.535	0.62	0.8	1.4
VIII	-	-	0.92	1.42

Table 2.1: Shannon ionic radii (in Å) as reported in [33] varying with coordination number for the cations of Al, Ga, In and the anion of O

2.3 Material Descriptors

The following text explains the ideas of feature engineering used in our application.

2.3.1 General Properties

A descriptor (or a feature) is a representation of the material data with a single vector of numbers. Let us assume a descriptor q of a crystalline material. We can construct many different descriptors - a simple example of a trivial descriptor is taking the atomic numbers of all the species in the material and arranging them into a vector $q = (Z_1, Z_2, \ldots, Z_k)$ assuming there are k different species in the material. We already arrive at a problem: How to order the atomic numbers in the vector? The most natural way is to make either an increasing or a decreasing arrangement of the numbers but no answer seems to be good. Much more troubling problem is the difference of dimensions between materials. For example, such descriptor for $CaTiO_3$ will have different dimensions than descriptor for Al_2O_3. Either we define a padding or a cropping of the vectors to ensure the same dimension for all materials. Of course, the simplest solution for this simple example is to realize there is a finite amount of species in the universe (118 as of now) and we can define the descriptor as a sequence of natural numbers from 1 to 118 with zeroes for the species not in the material. An unsolvable problem with this simple descriptor is the fact that it can be the same for two different materials. The simplest examples are H_2O and H_2O_2. This example demonstrates some of the problems we arrive at when we try to define a descriptor which would be sufficient for our application.

We can assume multiple descriptors for every material - we denote $q_1, q_2, \ldots, q_M \in \mathbb{R}^N, M \in \mathbb{N}$ a set of M arbitrary descriptors of a material. A vector can be given by concatenation of the M vectors into one. One can create descriptors using any material data he desires. However, problems arise when it is needed to use the positional data of the atoms in the molecule or lattice. Cartesian positions are not invariant to rotation and translation of the crystal. Additionally, the descriptor should be invariant to swapping two atoms of the same species, in other words, we demand invariance to permutation of atoms of the same species. The descriptor should not lose much of the information encoded in the Cartesian coordinates as well. A system of descriptors $q_1, q_2, \ldots, q_M, M \in \mathbb{N}$ is called complete if and only if there exists a bijective function between the system $q_1, q_2, \ldots, q_M$ and corresponding Cartesian coordinates. A system of descriptors $q_1, q_2, \ldots, q_M, M \in \mathbb{N}$ is called over-complete if there is a subset of $q_1, q_2, \ldots, q_M$ which is complete [3]. The bijective property of the descriptor is very strong and it can be easier not to

enforce it when choosing a material descriptor. Even though the descriptor breaks the bijective property for some crystals it can be bijective so often it does not pose a serious issue. Of course, one can demand additional properties. One of problems with defining a descriptor is the fact that the number of atoms in the unit cell varies a lot and this causes many descriptors not to have a fixed length. Naturally, the problem with assembling these descriptor into a matrix is that either the vectors in the matrix have to be padded or cropped.

Material descriptors which are not based on Cartesian coordinate system are usually quantities describing the species (atomic number, electronegativity, atomic radius, etc.) of the compound or their state (ionic radius, coordination number, charge, etc.). The choice of proper descriptors is determined by the physical feature of the crystal we wish to predict.

2.3.2 Brief Overview of Already Developed Descriptors

In recent years, many material descriptors have been proposed. We choose to briefly list only a few relevant ones which inspired some of the propositions outlined in the upcoming text below. We also note that many descriptors listed below are designed to work with molecules rather than solids. However, this does not take from the value and inspiration they offer because some of the descriptors can be generalized to be used with crystals but performance of such descriptors can vary greatly.

The early Coulomb matrix representation [28] has had a great impact on the chemoinformatics community and has been improved upon many times in later publications. For a molecule with N atoms we construct a matrix with entries

$$\boldsymbol{M}_{ij} = \begin{cases} 0.5 Z_i^{2.4}, i = j \\ \frac{Z_i Z_j}{\|\boldsymbol{R}_i - \boldsymbol{R}_j\|_2}, i \neq j \end{cases} \tag{2.4}$$

where Z_i is the atomic number of the ith atom, $\|\boldsymbol{R}_i - \boldsymbol{R}_j\|_2$ is the distance of atoms i and j in the ℓ_2 norm. The shape of the descriptor is inspired by the Coulomb repulsion potential which is in the Hamiltonian of the DFT equations used to calculate the data. Because the matrix $\boldsymbol{M}$ is symmetric, one can find the eigenvalues and conveniently vectorize them as $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_N)$, $\lambda_i > \lambda_{i+1}$ for $\forall i \in 1, \ldots, N$ and construct a measure of difference of two materials as

$$d(\boldsymbol{M}, \boldsymbol{M'}) = d(\boldsymbol{\lambda}, \boldsymbol{\lambda'}) = \|\boldsymbol{\lambda} - \boldsymbol{\lambda'}\|_2, \tag{2.5}$$

where $\boldsymbol{\lambda}$ is the spectrum of $\boldsymbol{M}$ ordered in decreasing fashion according to absolute value of the eigenvalues. For molecules with different amount of atoms, the shorter spectrum is appended with zeros to match the length of the longer spectrum. Even though it is a very simple representation, it has shown great success. The sorted spectrum of a matrix elegantly solves the problem of invariance outlined before. However, it has been shown [15] this compression of information from $\frac{1}{2}(N^2 + N)$ unique numbers to N numbers leads to a big information loss. The eigenvalue representation of the Coulomb matrix also results in loss of bijectivity of the representation [29], [25].

To solve the problem of information loss, it has been proposed to construct sorted Coulomb matrices where the rows (or columns) are sorted according to their ℓ_2 norm in decreasing fashion. Alternatively, the so called random Coulomb matrices can be constructed and these improvements lead to 4-5 times lower prediction error [15]. The important conclusion of these results is that the reduction of information of a matrix to its eigenvalues to represent materials is too crude and better representations can be found.

The Coulomb matrix descriptor has been improved into a representation called Bag of Bonds (BoB) [14] in the following way: the entries of matrix $\boldsymbol{M}$ in equation (2.4) are put into bags (groups) based on the two atom species of the $\boldsymbol{M}_{ij}$ pair. This way, multiple bags of numbers are created, vectorized and padded with zeros to get vectors of equal sizes across all molecules in the dataset. The elements of the vectors are sorted according to their absolute values and finally, all the vectors are concatenated

in arbitrary but consistent order across the whole dataset. This representation is invariant under rotations, translations and permutations of atoms of the same species. Also, it conveniently vectorizes the descriptor however, the padding with zeros procedure reduces the elegance of this approach. The further introduction of atomic species into the descriptor building procedure is a valuable idea and will play central role in the following sections.

The attempts to extended the Coulomb matrix approach to crystalline materials have not performed well [8].

Ionic radii and functions of ionic radii appear to be good descriptors for predicting stability of perovskites structures [22]. The accuracy of the prediction of the same regression problem was attempted to be improved by adding electronegativities of the species as another descriptor. This lead to some increase in prediction accuracy on the computational data but the unexpected cost was much worse agreement of the prediction with experimentally gathered data. The authors concluded this addition of electronegativity as another descriptor resulted in overfitting [22]. The conclusion we can gather from this is that we want to choose the simplest model possible which explains the phenomenon. Also, such models possess much better interpretability and we can learn more easily from their behavior about the problem we are solving.

Recently, crystal graph multilayer descriptor (CGMD) was used to predict properties of 2D materials [23] and utilizes the adjacency matrix of a crystal graph for solids [40]. One of the caveats of this representation is that the descriptor length depends on the size of the structure it describes.

Among other descriptors reviewed belong the many-body tensor representation (MBTR) which works for both molecules and crystals and extends the Coulomb matrix and Bag of Bonds approaches [17], partial radial distribution function (PRDF) representation for crystals [31], and the crystal graph of [40].

2.3.3 Studied Descriptors

The following text dives deeper into the structure of two descriptors which are ngram and Smooth Overlap of Atomic Positions (SOAP). Eventually, proposals of improvement of these descriptors are made based on the knowledge gathered about the physical problem and the available data.

2.3.3.1 ngram

The ngram representation won the Nomad2018 Predicting Transparent Conductors Kaggle competition which was studied in [35], [36], [16]. For the purposes of the crystal graph representation of crystalline materials, we introduce additional quantities which build upon the text of Section 2.2.

The positions itself of the atoms in the unit cell introduced in Section 2.2 do not possess enough information for the construction of a meaningful representation for solids. Considering periodicity of the unit cell, we start by defining the reduced distance between atoms i and j (using the knowledge from Section 2.2.1) as

$$\boldsymbol{r}_{ij}^{k,l,m} = \boldsymbol{r}_i - \boldsymbol{r}_j + (k,l,m)^T, k,l,m \in \mathbb{Z}. \tag{2.6}$$

The conversion to actual physical distance is given by

$$\boldsymbol{R}_{ij}^{k,l,m} = \boldsymbol{A}\boldsymbol{r}_{ij}^{k,l,m}. \tag{2.7}$$

Finally, the spatial distance of two atoms i and j over all possible neighboring cells is

$$d_{ij} = \min_{k,l,m\in\mathbb{Z}} \|\boldsymbol{R}_{ij}^{k,l,m}\|_2. \tag{2.8}$$

Therefore, we introduce the distance matrix as the closest distances between two atoms over all possible unit cells

$$\boldsymbol{D}_{ij} = \begin{cases} 0, i = j \\ d_{ij}, i \neq j. \end{cases} \tag{2.9}$$

As outlined in Section 2.2.2, the coordination number and coordination geometry considers only pairs of atoms where one is negatively charged and the other positively (for our dataset it means that for oxygen only the connections with the three metals are considered and vice versa). The Shannon ionic radius of an atom will be denoted as R^S and we will assume we implicitly know what is the species of the atom when we will use Shannon ionic radius in its context and we will use coordination VI for metals and coordination II for oxygen unless explicitly stated otherwise (see Table 2.1 for the list of all the Shannon ionic radii). The following rule will be used to collect the count of the amount of neighbors of each atom:

$$\boldsymbol{D}_{ij} < \alpha(R_i^S + R_j^S). \tag{2.10}$$

If the inequality above holds (in the language of computer science if it evaluates to be $True$), then atom i is in the coordination environment of j and vice versa. The amount of times (2.10) holds for an atom is the amount of its neighbors. The number α is determined beforehand and depends on the space group of the material. The value is chosen so that the amount of coordination atoms of each atom is roughly similar to actual physical coordination of such geometric configuration of the atoms. We enforce more physical meaning this way into the descriptor because the distribution of coordinations with these numbers is very physical and limits spurious behavior of the ngram descriptor.

	12	33	167	194	206	227 ($\gamma < 60°$)	227 ($\gamma \geq 60°$)
α	1.4	1.4	1.5	1.3	1.5	1.4	1.5

Table 2.2: The value of α depends on the space group, γ is the angle between lattice vectors $\boldsymbol{a}$ and $\boldsymbol{b}$

An atom X with $n \in \mathbb{N}$ of coordination atoms is referred to as X-n. For example, aluminum with 4 oxygens in its coordination environment is called Al-4. The number of atoms with the same coordination numbers is calculated, the values are divided by the volume of the unit cell V_{cell} because the cell sizes differ for every space group and the lattice vectors of Bravais lattice are not given unambiguously[1] and a histogram of these values is created. This representation is the unigram. The same principle works for counting chains of various length of particular atom-coordination pairs. We can count the number of normalized atom-coordination pairs, denoted X-n/Y-m (for example Al-4/O-3). It is easy to see that X-n/Y-m and Y-m/X-n have the same meaning. This representation is called the bigram. We can go further and count numbers of normalized triples (trigram), denoted X-n/Y-m/Z-l. Obviously we can go even further and construct quadgrams etc. Collectively, we call these representations ngrams. The ngram representation combined with kernel ridge regression won the Nomad2018 Predicting Transparent Conductors Kaggle competition [35], [36].

The representation can be viewed as an extension of the concept of percentages of different atoms in the compound. It is important to note that the coordination numbers and geometries which can be obtained from the Cartesian coordinate data which went through DFT calculations are not strictly within the crystallographic rules (e.g. symmetries can be broken because of the numerical nature of the result from

[1] for example, the length of every lattice vector can be two times bigger and consequently, the unit cell would include 8 times bigger volume with more atoms. This way, the counts of atoms of given coordination are normalized.

the DFT software calculation). However, this does not take anything from the physical interpretability of this representations.

In practice, the amount of X-n atom-coordination pairs is determined by the dataset. In general, the amount of columns of the regressor matrix is kc for unigram, $\binom{kc}{2}$ for bigram, $\binom{kc}{3}$ for trigram and so on, where k is the amount of atom species in the whole dataset and c is the amount of coordinations. The ngram can get very cumbersome for datasets with many atomic species however it turns out many columns are completely empty for trigram and quadgram. This sparsity is heavily influenced by the dataset we have. Our dataset has only 6 spacegroups and 4 atoms in it. A dataset with more spacegroups and more atoms would promote much lower sparsity.

This descriptor is invariant to rotations, translations, reflections and permutations of atoms of the same species.

2.3.3.2 ngram Extended

We propose to extend the ngram representation to include in some sense the physical distances of the atoms in the coordination environment of every atom. We start with the atom-coordination information constructed previously. For an atom X with coordination n denoted by a pair X-n we denote |X-n| the set of all atoms in the coordination environments of all atoms which are X-n. In other words, we find all the neighbors of atoms with the same coordination and species and make one set from them. The distance between an atom i of the coordination environment and the central atom of the coordination environment X-n is in the notation of distance matrix $\boldsymbol{D}_{iX}$ where we shorten the subscript for better readability.

With the details prepared, we can define an extension of the ngram descriptor as

$$\Sigma_{\text{X-n}}(p) = \begin{cases} \sum_{i \in |\text{X-n}|} 1, p = 0, \\ \frac{1}{\sum_{i \in |\text{X-n}|} 1} \sum_{i \in |\text{X-n}|} \boldsymbol{D}_{iX}^{p}, p \neq 0, \end{cases} \tag{2.11}$$

and a scaled version where the distances between atoms are scaled by the Shannon ionic radius (Table 2.1) of the central atom of the coordination environment (radius of coordination II is used for oxygen and VI is used for metals)

$$\tilde{\Sigma}_{\text{X-n}}(p) = \begin{cases} \sum_{i \in |\text{X-n}|} 1, p = 0, \\ \frac{1}{\sum_{i \in |\text{X-n}|} 1} \sum_{i \in |\text{X-n}|} \left(\frac{\boldsymbol{D}_{iX}}{R_i^S} \right)^p, p \neq 0. \end{cases} \tag{2.12}$$

Number $p \in \mathbb{Z}$ manages further extension of the descriptor. The choice $p = 0$ gives the amount of neighboring atoms over all atoms of the same coordination and the same species. Probably the most interesting is $p = 1$ which has the meaning of average distance of the atoms of the coordination environment which can also be interpreted as the average bond distance of metal atoms from oxygen atoms for the given atom and coordination. The choice $p = -1$ possesses information similar to the Coulomb matrix descriptor from Section 2.3.2. A physical meaning of terms for $p = -6$ and $p = -12$ can be found in the structure of a well-known Lennard-Jones potential for classical modeling of atomic interactions [39]

$$v_{LJ}(r) = 4\epsilon \left(\left[\frac{\sigma}{r} \right]^{12} - \left[\frac{\sigma}{r} \right]^{6} \right), \tag{2.13}$$

where ϵ is a constant describing the depth of a potential energy well, r is the interatomic distance and σ is the interparticle distance where the Lennard-Jones potential changes sign and this change describes the change between repulsive and attractive nature of the potential. Our descriptor therefore attempts to model in some sense the r^{-6} and r^{-12} terms which the developed physical understanding of the problem

showed to bring good description of the problem of crystalline stability. Other values of p do not necessarily have a direct physical meaning but one can view them as a part of the expansion into Laurent series or an attempt to take into account higher order interactions similar to the LASSO experiment in Chapter 3 where exponentials or squares of distances were used.

The physical interpretation of this descriptor comes from the assumption that all atoms of the same species and the same coordination have roughly the same coordination environment in terms of distances and species. The repulsing or attracting significance (in the context of (2.13)) of the term in the sum of $\Sigma_{\text{X-n}}(p)$ or $\tilde{\Sigma}_{\text{X-n}}(p)$ is determined by the coefficient in the kernel ridge regression model. This extension of ngram is also invariant to rotations, translations, reflections and permutations of atoms of the same species but as can be easily seen from the construction. However, it is not bijective to the Cartesian coordinate system.

The mathematical interpretation of this descriptor is that the distances of atoms of the same coordination and the same species have a certain distribution and we apply a function to these distances and then take an average. For example, for $p = 1$, this means taking the average distance.

The procedure of ngram and extended ngram construction outlined above is applied to $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$ alloy as an example which can be found in Appendix B.

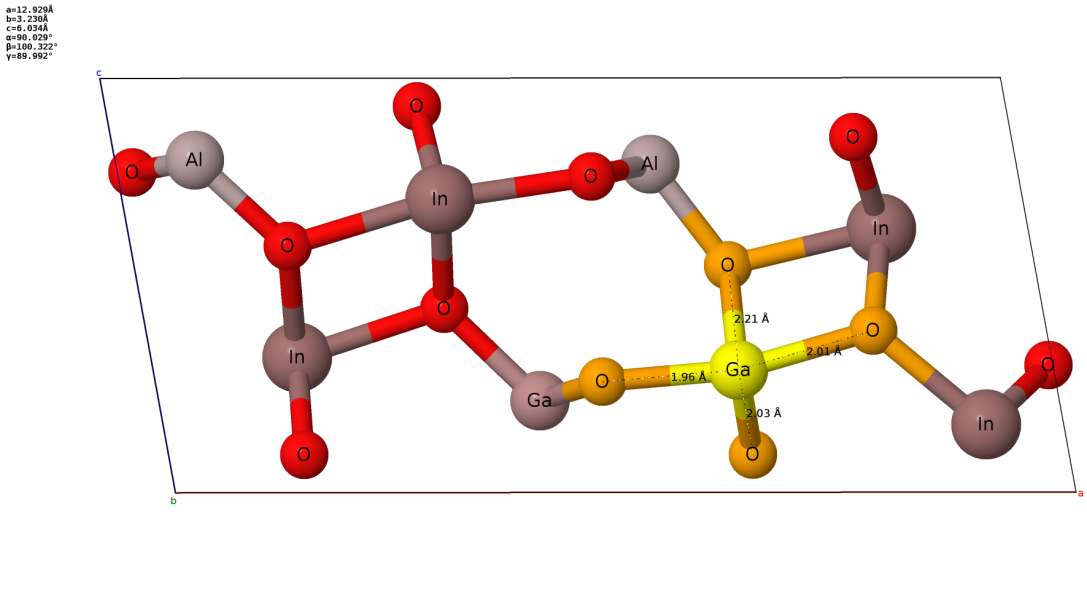


Figure 2.2: Alloy $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$ with one atom of gallium in yellow and its coordination environment in orange with the interatomic distances (notice the distances are approximately 2 Å). This gallium atom has 4 oxygens in its coordination environment, therefore it is Ga-4. The other gallium atom of the alloy is Ga-3 where the third oxygen atom is not displayed because it is in the neighboring unit cell. Still, it is close enough to fulfill the condition in equation (2.10) [20]

2.3.3.3 Smooth Overlap of Atomic Positions (SOAP)

The SOAP representation [3], [2] placed 3rd in the Nomad2018 Predicting Transparent Conductors Kaggle competition which was studied [35], [36], [16]. For the purposes of developing the SOAP representation, we define the neighborhood density function of an atom i as

$$\rho_i(\boldsymbol{r}) = \sum_j w_i \delta(\boldsymbol{r} - \boldsymbol{r}_j), \tag{2.14}$$

where $\delta(\boldsymbol{r})$ is the Dirac δ-function. We sum over all the atoms in the atomic neighborhood of an atom which is defined with some cutoff distance from the central atom, w_j is a weighing factor of an atom j based on its species and $\boldsymbol{r} - \boldsymbol{r}_j$ is the vector from the central atom i to the atom j in the neighborhood.

For simplicity, we will first assume there is only one species in the material which allows us to put $w_i = 1$ and i runs through atoms of the same species only. The smoothness of the descriptor is achieved by smearing of the Dirac δ-functions into Gaussian functions

$$\rho^{(i)}(\boldsymbol{r}) = \sum_j \exp\left(\frac{\|\boldsymbol{r} - \boldsymbol{r}_j\|_2^2}{2\sigma_j^2}\right). \tag{2.15}$$

The parameter σ_j (in Å) controls the smearing effect. Now, the atomic neighborhood density in (2.15) is expanded into a basis of spherical harmonics Y_{lm} and a set of orthonormal radial basis functions $\{g_n\}$

$$\rho^{(i)}(\boldsymbol{r}) = \sum_{n,l,m} c_{nlm}^{(i)} g_n(\boldsymbol{r}) Y_{lm}(\hat{\boldsymbol{r}}), \tag{2.16}$$

where $\hat{\boldsymbol{r}}$ is the normalization of the vector $\boldsymbol{r}$ and $c_{nlm}^{(i)}$ are the expansion coefficients corresponding to atom i.

The SOAP descriptor vector for a material is constructed from the expansion coefficients as

$$p_{nn'l}^{(i)} = \frac{1}{\sqrt{2l+1}} \sum_m c_{nlm}^{(i)} (c_{nlm}^{(i)})^*, \tag{2.17}$$

where the star symbol means complex conjugate. Indices n, n' and l run from 0 to some predefined n_{max}, n'_{max} and l_{max}. The magnitude of the maximal values of the indices regulates the length of the descriptor.

The SOAP descriptor generalizes to more that one species by constructing a separate neighborhood density function of (2.15) and expansion of (2.16) for every species in the material (with corresponding weighing w_i). Then, for the atom i of species Z and the atom j of species Z' we get

$$p_{nn'l}^{(Z_j Z'_i)} = \frac{1}{\sqrt{2l+1}} \sum_m c_{nlm}^{(Z_j)} (c_{nlm}^{(Z'_i)})^*. \tag{2.18}$$

The entries for each pair of atoms in the material are then concatenated into one vector which is then the actual SOAP descriptor for every atom. The SOAP vector is then given as a the average over all atoms in the material.

Chapter 3

Classification Problem of Binary Compounds Experiment

The goal of this experiment is to develop a model for prediction of the crystal structure of semiconductors. These compounds are simple: binary compounds AB consisting of element A and element B. The dataset consists of compounds which crystallize in three distinct structures: rocksalt (RS), zincblende (ZB) and wurtzite (WZ). Figure 3.1 shows the geometry of the aforementioned structures. However, the energies of ZB and WZ are very close (see Appendix A, column $E(ZB) - E(WZ)$) and for the sake of simplicity, these two structures are not distinguished in this experiment (the corresponding ΔE_{WZ} for the dataset is in a table in Appendix A). The target property of this experiment is the difference between the energy of rocksalt E_{RS} and the energy of zincblende E_{ZB} for the given compound AB - that is $\Delta E = E_{RS} - E_{ZB}$. Therefore, our goal is to find a model for binary compounds AB which assigns them the right crystalline structure - the sign of the energy difference ΔE_{AB} gives the structure.

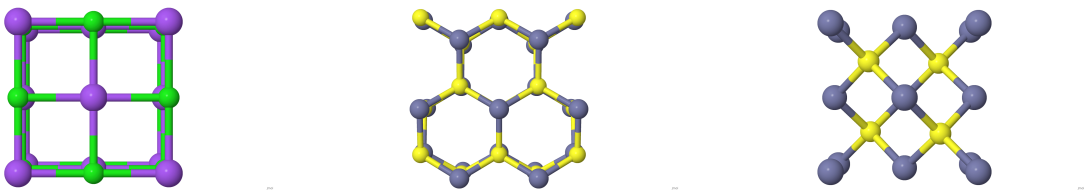


Figure 3.1: From left to right: rocksalt, wurtzite and zincblende structures [20], [34]

3.1 The Dataset

The dataset consists of 34 elements for which 7 physically meaningful features were calculated:

- Ionization potential IP [eV]
- Electron Affinity EA [eV]
- Highest Occupied Molecular Orbital H [eV]
- Lowest Unoccupied Atomic Orbital L [eV]
- Radius where Radial Probability Density of valence s orbital is maximal r_s [Å]
- Radius where Radial Probability Density of valence p orbital is maximal r_p [Å]
- Radius where Radial Probability Density of valence d orbital is maximal r_d [Å]

In total 82 datapoints are available in this dataset (the table with the used data is available in Appendix A). The combinations of the 34 elements into 82 compounds models actual materials which exists in nature which means they are not picked arbitrarily. It is only natural to do such a constraint because we want to predict actual natural phenomena. It comes from quantum mechani cs that the physical features are correlated in terms of the Pearson correlation coefficient of two 82-dimensional feature vectors. The physical meaning of the differences H-L and IP-EA is very similar. The radii are correlated with the energy quantities as well. The correlation of the features is an issue and it will come up later on during the conducted experiments.

For the 82 compounds, the energy difference $\Delta E = E(RS) - E(ZB)$ was calculated. We put $\boldsymbol{y} = \boldsymbol{\Delta E}$ in accord with the notation used previously. The labelling AB is not arbitrary. The label A is assigned to the element with lower Mulliken Electronegativity given by $EN = -\frac{1}{2}(EA + IP)$. This way, 14 primary features of a compound AB are defined as

$$\boldsymbol{x}_{AB} = \Big(IP(A), EA(A), H(A), L(A), r_s(A), r_p(A), r_d(A), IP(B), EA(B), H(B), L(B), r_s(B), r_p(B), r_d(B)\Big). \tag{3.1}$$

3.1.1 The Feature Space Generation

To capture the relationship between the variables in (3.1), a nonlinear mapping $\Phi : \mathbb{R}^{14} \to \mathbb{R}^M$ is defined together with a set of unary and binary operations

$$\{|-|, +, /, \cdot, ()^2, \exp[]\}, \tag{3.2}$$

that execute the mapping and the primary features $\mathbf{x}_{AB}$ are used to generate a feature space of expressions from which the matrix $\boldsymbol{X}$ is constructed. The operations defined in (3.2) are (in order of appearance) difference with absolute value, summation, multiplication, power of two and exponential.

From this high dimensional matrix $\boldsymbol{X}$, the optimal descriptors (features) will be chosen. The mapping procedure introduces non-linearity needed for better description of the relationship between the primary features in (3.1) and the energy difference ΔE_{AB}.

We will use the inequality (1.66) which gives the estimate that roughly a few thousand features are admissible for our amount of measurements. The primary features are divided into subsets based on their physical units and meaning for more convenient illustration of the generation procedure.

ID	Quantities	Set size
A1	$IP(A), EA(A), IP(B), EA(B)$	4
A2	$H(A), L(A), H(B), L(B)$	4
A3	$r_s(A)$, $r_p(A)$, $r_d(A)$, $r_s(B)$, $r_p(B)$, $r_d(B)$	6

Table 3.1: The primary features divided into subsets based on their units and meaning

The features in (3.1) are then combined as follows and their number is calculated:

- B1: The sum and the absolute difference of two different features from A1
- B2: The sum and the absolute difference of different two features from A2
- B3: The sum and the absolute difference of two different features from A3
- C3: Squares of all A3 features and squares of all sums of features in B3
- D3: Exponentials of all A3 features and all sums in B3
- E3: Exponentials of C3
- F1: The following 4 expressions:

$$\begin{vmatrix} |IP(A) - EA(A)| + |IP(B) - EA(B)| \\ |IP(A) - EA(A)| - |IP(B) - EA(B)| \\ |IP(A) + EA(A)| + |IP(B) + EA(B)| \\ |IP(A) + EA(A)| - |IP(B) + EA(B)| \end{vmatrix}$$

- F2: The following 4 expressions:

$$\begin{vmatrix} |H(A) - L(A)| + |H(B) - L(B)| \\ |H(A) - L(A)| - |H(B) - L(B)| \\ |H(A) + L(A)| + |H(B) + L(B)| \\ |H(A) + L(A)| - |H(B) + L(B)| \end{vmatrix}$$

- F3: The same 4 expressions as in F1, F2 with inputs of all pairs of $r_s(A)$, $r_p(A)$, $r_d(A)$ in the first absolute term and all pairs of $r_s(B)$, $r_p(B)$, $r_d(B)$ in the second term
- G: Ratios of all expressions in {Ai, Bi} with all expressions in {A3, C3, D3, E3} for i = 1, 2. The ratio 1/A3. Ratios A3/A3, A3/C3, B3/A3, B3/C3 such that only the unique expressions are chosen

This gives total of 4376 potential descriptors and therefore 4376 columns of $\boldsymbol{X}$ which comfortably fits the estimate. The number of descriptors and examples for each set are given in Table 3.2.

The feature space used in this work is almost the same as the one in [9].

3.1.2 The LASSO+ℓ_0 Method

It was briefly noted earlier in this chapter that the correlation of the features is an issue. Additionally, we use combinations of already correlated quantities and therefore we can expect the correlation of the generated features to be a problem as well. Indeed, as it turns out LASSO itself performs unpredictably for values of λ which choose 1D, 2D, 3D and 4D descriptors and the optimal descriptor cannot be chosen this way reliably as we would do so for columns of a matrix with low correlation (see Table 3.3).

ID	Feature examples	Set size
B1	$\lvert IP(A) + IP(B)\rvert$, $\lvert IP(B) - EA(B)\rvert$, ...	$2\binom{4}{2} = 12$
B2	$\lvert H(A) + H(B)\rvert$, $\lvert H(B) - L(B)\rvert$, ...	$2\binom{4}{2} = 12$
B3	$\lvert r_s(A) - r_p(A)\rvert$, $(r_d(B) + r_s(A))$, ...	$2\binom{6}{2} = 30$
C3	$r_s(A)^2$, $(r_d(B) + r_s(A))^2$, ...	$6 + \binom{6}{2} = 21$
D3	$\exp[r_s(A)]$, $\exp[r_d(B) + r_s(A)]$, ...	$6 + \binom{6}{2} = 21$
E3	$\exp[r_s(A)^2]$, $\exp[(r_d(B) + r_s(A))^2]$, ...	$6 + \binom{6}{2} = 21$
F1	$\big\lvert\lvert IP(A) - EA(A)\rvert + \lvert IP(B) - EA(B)\rvert\big\rvert$, ...	4
F2	$\big\lvert\lvert H(A) - L(A)\rvert + \lvert H(B) - L(B)\rvert\big\rvert$, ...	4
F3	$\big\lvert\lvert r_s(A) - r_p(A)\rvert + \lvert r_s(B) - r_p(B)\rvert\big\rvert$, ...	36
G	$\frac{IP(A)}{r_s(A)}, \frac{IP(A)}{(r_p(A)+r_s(B))^2}, \frac{IP(A)}{\exp[r_s(A)]}, \frac{\lvert r_p(A)-r_s(B)\rvert}{\exp[r_p(B)]}$, ...	4201

Table 3.2: Application of the chosen operations on primary features and the corresponding set sizes

λ value	Column index
$\lambda_1 = 0.4229$	None
$\lambda_2 = 0.3944$	819
$\lambda_3 = 0.3679$	819, 1105
$\lambda_4 = 0.3431$	819, 1105, 1470
$\lambda_5 = 0.3199$	819, 1105, 1470, 2172
$\lambda_8 = 0.2595$	819, 1105, 1470, 2172, 966
$\lambda_{10} = 0.2257$	819, 1105, 1470, 2172, 966, 903, 1021
$\lambda_{12} = 0.1963$	819, 1105, 1470, 2172, 903, 1021, 966
$\lambda_{14} = 0.1707$	1105, 819, 1470, 2172, 903, 1021, 2540, 966
$\lambda_{15} = 0.1592$	1105, 819, 1470, 2172, 1021, 903, 2540
⋮	⋮
$\lambda_{72} = 0.003$	3441, 2562, 1259, 4125, 4270, 3013, 2235, 2561, ...

Table 3.3: Found descriptors for given λ value sorted from the most significant to the least significant

The λ_i value is chosen using the following recursive formula

$$\lambda_i = \left(\frac{1}{1000}\right)^{dim(\lambda)-1} \lambda_{i-1}, i \in \{2, \ldots, dim(\lambda)\} \tag{3.3}$$

where $dim(\lambda)$ means the number of λ values evaluated during the LASSO+ℓ_0 procedure, $\lambda_1 = \frac{1}{N} \max_i |\langle \boldsymbol{x}_i, \boldsymbol{\Delta E}\rangle|$ is the threshold value when the first non-zero coefficient appears. The vector $\boldsymbol{x}_i$ is a column of $\boldsymbol{X}$ and $\boldsymbol{\Delta E}$ is the vector of energy differences. This formula was derived from the approaches to choosing λ values in [9].

The procedure appears to be reasonably stable for λ close the threshold λ_1. The significance of the first descriptor decreases with the descend of λ and it completely disappears for certain low values of λ. The more non-zero coefficients are admissible, the less reliable the LASSO selection appears to be. For $\lambda \approx 0.003$, there are 42 non-zero coefficients and the selection hierarchy is completely different compared to selections with higher λ values. Therefore, the following approach has been proposed [10]:

The LASSO selection is carried out for a beforehand chosen vector of λ values with length $dim(\lambda)$. From these LASSO selections for various λ, the best Θ ocurring descriptors throughout the calculations are gathered and among these the best 1D, 2D, 3D and 4D descriptor are found using the least squares regression method for all $\binom{\Theta}{1}, \binom{\Theta}{2}, \binom{\Theta}{3}, \binom{\Theta}{4}$ which is effectively the ℓ_0 minimization. The 1D, 2D, 3D and 4D OLS models with the lowest MSEs are selected as the winners.

The issues and peculiarities of this approach which naturally appeared will be discussed thoroughly in the following text.

3.1.3 Results and Discussion

The setting was chosen to be $dim(\lambda) = 100$ values of λ starting from the threshold λ_1 and decreasing as a geometric sequence with $\lambda_{100} \approx 4.23 \cdot 10^{-4}$ being the lowest. The amount of contending descriptors was set to be $\Theta = 30$. This choice will be discussed later on further. Before the training, the data were standardized and the sci-kit learn [27] LASSO implementation was used.

The LASSO+ℓ_0 method found the following models in the sense of minimal MSE:

$$\Delta E_{D1} = 0.055\frac{|IP(A) + IP(B)|}{r_p(A)^2} - 0.332 \tag{3.4}$$

$$\Delta E_{D2} = 0.113\frac{|IP(B) - EA(B)|}{r_p(A)^2} - 1.558\frac{|r_s(A) - r_p(B)|}{\exp(r_s(A))} - 0.133 \tag{3.5}$$

$$\Delta E_{D3} = 0.108\frac{|IP(B) - EA(B)|}{r_p(A)^2} - 1.751\frac{|r_s(A) - r_p(B)|}{\exp(r_s(A))} - 9.042\frac{|r_s(B) - r_p(B)|}{\exp(r_d(A) + r_s(B))} - 0.027 \tag{3.6}$$

$$\Delta E_{D4} = 0.186\frac{|H(A) + H(B)|}{\exp(r_p(A)^2)} - 1.031\frac{|r_s(A) - r_p(B)|}{\exp(r_s(A))} - 11.246\frac{|r_s(B) - r_p(B)|}{\exp(r_d(A) + r_s(B))} + 234.153\frac{|r_s(A) - r_d(B)|}{\exp[(r_s(A) + r_d(B))^2]} + 0.072 \tag{3.7}$$

The starting point is the worst possible RMSE = 0.457 eV which is for prediction $\Delta E_{D_j} = 0$ for $j \in \{1, 2, 3, 4\}$ (the coefficients of the model are zero). Test RMSE and MaxAE of the models are reported in Table 3.4. The 2D and 3D descriptors match exactly the ones from [10]. However, slightly different

In eV	1D	2D	3D	4D
RMSE	0.138	0.099	0.076	0.063
MaxAE	0.421	0.287	0.243	0.163

Table 3.4: Test RMSE and test MaxAE of the best models

coefficients were recovered. This is most likely a rounding error during the handling of the values and feature space generation. Also, it is possible that a different implementation of the numerical solver was used which can result in slight inaccuracies. Nevertheless, the results are in accord with the available literature [10], [9]. The sign of the second terms in 2D and 3D descriptor are different. Given the fact that our 2D descriptor in (3.5) gives the very same classification line (Figure 3.2), this is most likely a mistake in the publications. The 1D descriptor does not match the one from [9] which is the first expression of 2D and 3D descriptor, our 1D descriptor is only slightly better in terms of MSE. Interestingly, the 1D descriptor we found is the very first feature which LASSO recovers (labeled 819) whereas the other slightly worse feature (labeled 966) appears a couple steps later during the procedure (see Table 3.3). The 4D descriptor was not reported in the publications [10] and [9]. It is interesting to notice the second and the third terms in D4 match the second and the third terms in 3D. The test error RMSE and MaxAE drop most significantly between 1D and 2D descriptors. Also, the test MaxAE drops substantially between 3D and 4D descriptors.

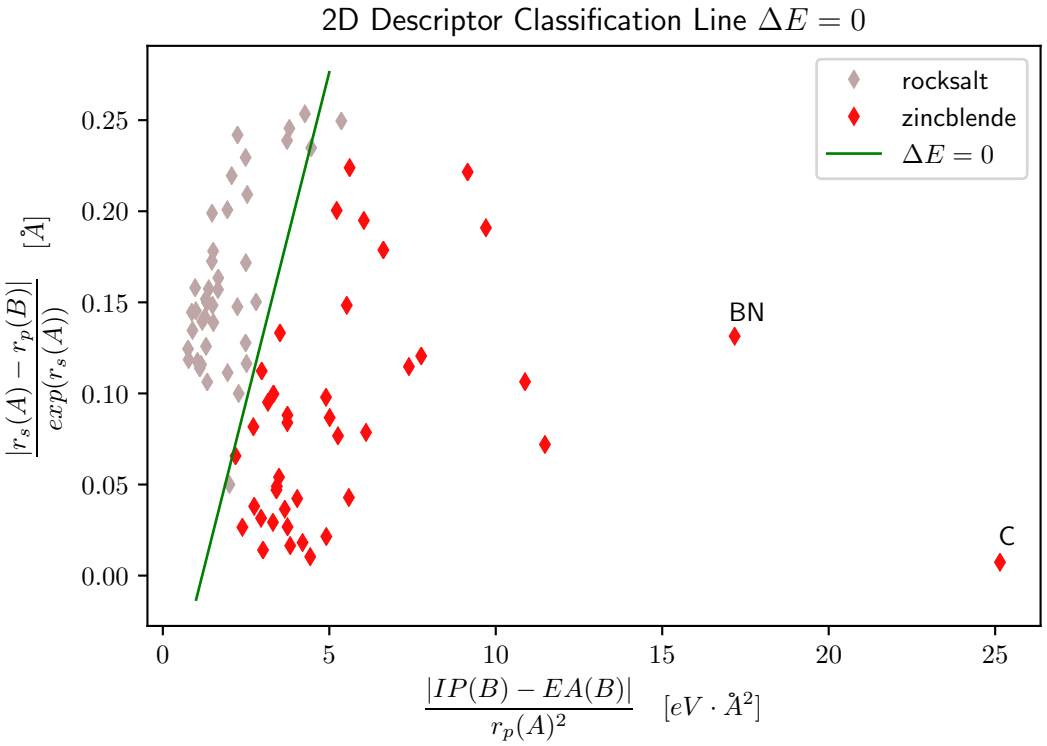


Figure 3.2: The 2D descriptor classification line which is equivalent to $\Delta E = 0$ in equation (3.5). Two outliers BN and C (diamond) are labeled

The optimal choice of the amount of λ parameters and the amount of contenders Θ depends on the data set. In this case, there are 53 non-zero coefficients for the lowest value λ_{100}. Extending the geometric sequence up to λ_{150} yields up to 55 non-zero coefficients by the end of the procedure but the recovered models did not change. However, the models did change for choice of 250 values of λ. The third term of D3 and D4

$$\frac{|r_s(B) - r_p(B)|}{\exp(r_d(A) + r_s(B))}$$

changed to

$$\frac{|r_s(B) - r_p(B)|}{\exp(r_d(A))}.$$

The change in the errors was barely noticeable. In fact, the biggest change was 1% increase in MaxAE of the 4D descriptor which suggests the method actually performed slightly worse with a more expensive setting. The value of Θ was increased gradually to 35, 40, 45 and finally 50. The models recovered did not change with $\Theta > 30$. Examining the Θ set shows that there are 15 and 54 pairs of columns with the absolute value of the Pearson correlation coefficient higher than 0.98 and 0.95 respectively. The two very competitive features of the 1D descriptor (labeled 819) and the first term of 2D descriptor (labeled 966) show Pearson correlation coefficient of 0.985. This is the aforementioned problem of correlation between the primary features and consequently, the whole feature space.

The commentary of the implementation of the experiment can be found in Appendix C.

3.1.4 Cross Validation, Sensitivity Analysis and Extrapolation

We perform a thorough study of the used methodoly to develop an idea of how reliable the results we managed to obtain really are.

3.1.4.1 Leave One Out Cross Validation (LOOCV)

Simple LOOCV was employed. In total 82 fits of the LASSO+ℓ_0 were performed - each material was the test set once. The results are reported in Table 3.5. Average RMSE and MaxAE are the same measure of error for LOOCV so only average RMSE over all test materials is reported - each material was a test datapoint once.

In eV	1D	2D	3D	4D
RMSE CV	0.132	0.104	0.085	0.062

Table 3.5: The LOOCV RMSE for 1D, 2D, 3D, 4D descriptors

3.1.4.2 Complexity of the Feature Space

The feature space was divided into 6 tiers. We want to see how well the LASSO+ℓ_0 method can navigate the feature space. The division rule is the number of operations needed to build the final feature from the primary features:

- Tier 0 - the 14 primary features
- Tier 1 - 1 operation (unary or binary), e. g. $|IP(A) + IP(B)|$ or $r_s(A)^2$, 164 features in total.
- Tier 2 - 2 operations, e .g. $\exp\left[r_s(A)^2\right]$ or $\exp\left[r_d(B) + r_s(A)\right]$, 596 features in total.
- Tier 3 - 3 operations, e. g. $\exp\left[\left(r_d(B) + r_s(A)\right)^2\right]$, 1669 features in total.
- Tier 4 - 4 operations, e. g. $\frac{|r_p(A)-r_s(B)|}{\exp\left[r_p(B)^2\right]}$, 3566 features in total.
- Tier 5 - 5 operations, e. g. $\frac{|r_s(A)-r_d(B)|}{\exp[(r_s(A)+r_d(B))^2]}$, 4376 features in total.b d

Then, 6 feature subspaces are generated and LASSO+ℓ_0 is applied to them. Each feature subspace includes its tier as well as all lower tiers, e. g. Tier 2 feature subspace includes Tier 0, Tier 1 and Tier 2 descriptors. Given the small size of the data set, the cross validation approach of verification of the model was chosen. The data is split at random into two parts consisting of 10% and 90% of the data (this means 75 measurements for training and 7 measurements for testing). The model is learned on the train set and the RMSE and MaxAE are evaluated on the test set. The random split is carried out 150 times to gain a good statistic. The average RMSE and MaxAE after the procedure is reported in Table 3.6.

In eV		Tier 0	Tier 1	Tier 2	Tier 3	Tier 4	Tier 5
1D	RMSE	0.262	0.187	0.162	0.159	0.172	0.172
	MaxAE	0.529	0.380	0.304	0.305	0.340	0.340
2D	RMSE	0.202	0.192	0.138	0.103	0.129	0.129
	MaxAE	0.404	0.394	0.257	0.180	0.246	0.247
3D	RMSE	0.190	0.145	0.133	0.079	0.108	0.110
	MaxAE	0.418	0.294	0.267	0.144	0.217	0.220
4D	RMSE	0.194	0.136	0.102	0.086	0.112	0.093
	MaxAE	0.415	0.266	0.194	0.167	0.233	0.187

Table 3.6: L10%OCV RMSE and MaxAE of cross validation with different feature space sizes

We observed expected reduction in test errors when the complexity of the feature subspace increases. However, the minimal error was found to be for Tier 3 feature subspace size. This is not in accord with [9]. The cause of this has been investigated and the reason is that upon cross validation, it is not guaranteed the best descriptors make it into the Θ set. If the sought best descriptors are artificially added into Θ, they are chosen every single time. The conclusion is that the richness of the higher tiers which contain many highly correlated features combined with a small dataset leads to unstable behavior when cross validation scheme is employed. One solution to this problem would be increasing the size of the Θ set. This is not a big problem for 1D and 2D descriptor recovery but can become cumbersome with 3D and 4D descriptors where the $\binom{\Theta}{3}$ and $\binom{\Theta}{4}$ can be too big to be advantageous in terms of time.

3.1.4.3 Sensitivity Analysis

Sensitivity analysis aims to find which features affect the model the most and how much the model depends on certain values of the features. The LASSO+ℓ_0 finds the best descriptor from the feature space generated by the primary features. Applying noise to the primary features and the fitted property is a way of determining the effect of numerical inaccuracies of the DFT calculations and ultimately validates the model and the physical relationship which was found.

Noised Primary Features In this series of tests, we multiply one feature with Gaussian noise with mean 1 and standard deviation σ taken from $\{0.001, 0.01, 0.03, 0.05, 0.1, 0.13, 0.3\}$ for every test. The feature space is then generated from the primary features where one is always noised.

Feature	Quantity	Gaussian noise $N(1,\sigma)$						
		σ = 0.001	σ = 0.010	σ = 0.030	σ = 0.05	σ = 0.100	σ = 0.130	σ = 0.300
	Recovery of all data 2D [%/100]	0.74	0.65	0.31	0.18	0	0	0
$IP(A)$	Recovery of LOOCV 2D [%/100]	0.95	0.83	0.05	0	0	0	0
	Recovery of L10%CV 2D [%/100]	0.92	0.5	0.36	0	0	0	0
	Recovery of all data 2D [%/100]	0.74	0.75	0.73	0.71	0.7	0.67	0.62
$EA(A)$	Recovery of LOOCV 2D [%/100]	1	0.99	0.95	0.91	0.83	0.56	0.82
	Recovery of L10%CV 2D [%/100]	1	0.98	0.92	0.9	0.8	0.84	0.52
	Recovery of all data 2D [%/100]	0.74	0.73	0.36	0.25	0.02	0.02	0
$IP(B)$	Recovery of LOOCV 2D [%/100]	0.98	0.7	0.24	0.33	0.11	0.02	0.02
	Recovery of L10%CV 2D [%/100]	1	0.86	0.82	0.36	0.24	0.22	0.22
	Recovery of all data 2D [%/100]	0.75	0.71	0.6	0.45	0.39	0.42	0.14
$EA(B)$	Recovery of LOOCV 2D [%/100]	0.99	0.98	0.49	0.28	0.24	0.26	0.24
	Recovery of L10%CV 2D [%/100]	1	0.86	0.88	0.7	0.82	0.5	0.46
	Recovery of all data 2D [%/100]	0.74	0.74	0.74	0.73	0.75	0.74	0.87
$H(A)$	Recovery of LOOCV 2D [%/100]	1	0.98	0.85	0.84	0.83	0.83	0.83
	Recovery of L10%CV 2D [%/100]	0.98	1	0.76	0.8	0.76	0.72	0.6
	Recovery of all data 2D [%/100]	0.74	0.75	0.75	0.75	0.75	0.75	0.71
$L(A)$	Recovery of LOOCV 2D [%/100]	1	1	1	1	0.99	1	1
	Recovery of L10%CV 2D [%/100]	1	1	0.96	0.92	0.96	0.96	0.92
	Recovery of all data 2D [%/100]	0.74	0.74	0.74	0.73	0.72	0.74	0.71
$H(B)$	Recovery of LOOCV 2D [%/100]	1	1	1	1	0.85	0.98	1
	Recovery of L10%CV 2D [%/100]	1	0.98	0.98	0.78	0.92	0.92	0.92
	Recovery of all data 2D [%/100]	0.74	0.74	0.77	0.71	0.69	0.66	0.75
$L(B)$	Recovery of LOOCV 2D [%/100]	1	0.99	0.99	0.98	0.99	0.98	0.83
	Recovery of L10%CV 2D [%/100]	1	0.96	0.98	0.96	0.94	0.92	0.8
	Recovery of all data 2D [%/100]	0.74	0.74	0.72	0.71	0.65	0.51	0
$r_s(A)$	Recovery of LOOCV 2D [%/100]	0.99	0.99	0.99	0.85	0.87	0.01	0.02
	Recovery of L10%CV 2D [%/100]	0.98	0.92	0.82	0.82	0.42	0.68	0.1
	Recovery of all data 2D [%/100]	0.74	0.67	0.49	0.31	0	0	0
$r_p(A)$	Recovery of LOOCV 2D [%/100]	0.99	0.93	0.02	0.12	0	0	0
	Recovery of L10%CV 2D [%/100]	0.96	0.8	0.36	0	0	0	0
	Recovery of all data 2D [%/100]	0.74	0.74	0.74	0.74	0.74	0.74	0.74
$r_d(A)$	Recovery of LOOCV 2D [%/100]	1	1	1	0.99	0.98	0.99	0.98
	Recovery of L10%CV 2D [%/100]	1	1	0.94	0.94	0.84	0.9	0.82
	Recovery of all data 2D [%/100]	0.74	0.75	0.77	0.75	0.76	0.75	0.75
$r_s(B)$	Recovery of LOOCV 2D [%/100]	1	1	0.95	0.98	0.94	0.9	0.95
	Recovery of L10%CV 2D [%/100]	0.96	0.88	0.88	0.78	0.8	0.78	0.78
	Recovery of all data 2D [%/100]	0.74	0.75	0.76	0.76	0.69	0.59	0
$r_p(B)$	Recovery of LOOCV 2D [%/100]	1	0.99	0.99	0.96	0.98	0.87	0.01
	Recovery of L10%CV 2D [%/100]	0.98	0.96	0.74	0.78	0.58	0.4	0.02
	Recovery of all data 2D [%/100]	0.74	0.75	0.67	0.67	0.71	0.67	0.72
$r_d(B)$	Recovery of LOOCV 2D [%/100]	0.99	0.96	0.88	0.93	0.82	0.87	0.85
	Recovery of L10%CV 2D [%/100]	0.98	0.94	0.82	0.8	0.46	0.64	0.64

Table 3.7: Noise applied to the primary features. Reported as the fraction of times recovery was successful

Table 3.7 summarizes the tests carried out. For every feature and for each noise level σ, 50 instances of the Gaussian noise are applied to the feature for 50 L10%OCV procedures and 82 LOOCV procedures which results in a statistic of 2500 samples and 4100 respectively. Noising primary features $EA(A)$, $H(A)$, $L(A)$, $H(B)$, $L(B)$, $r_d(A)$, $r_s(B)$ and $r_d(B)$ has much less effect that the other 6 features where 5 of these appear in the 2D descriptor and the 6th is $IP(A)$ which appears in the expression of the 1D descriptor which is highly correlated with the first expression of the 2D descriptor as mentioned before. Therefore it is not surprising this feature also affects the recovery of the descriptor because it apparently appears to be a part of the best 2D descriptor recovered during cross validation. The LOOCV RMSE remained stable throughout tests with its value being 0.1-0.13 eV for all noise levels and noised features except $IP(A)$ where the RMSE rose gradually up to 0.24 eV. The L10%OCV RMSE remained stable throughout the tests as well at 0.12-0.14 eV and $IP(A)$ was unstable with gradual increase of RMSE up to 0.35 eV. The percentage of recovery of the 2D descriptor obtained on all data remains stable for the 8 aforementioned

features and recovery of the same LOOCV and L10%OCV descriptors with or without noise is also very stable.

Adding Noise to ΔE In this test, the energy difference ΔE was perturbed with additive noise drafted from uniform distribution $\mathcal{U}(-\delta, \delta)$ where $\delta \in \{0.01, 0.03, 0.1, 0.2\}$. The sources [10] and [9] do not mention how the relevant subsets of the whole feature space were selected. Therefore, we choose Tier 3 and Tier 4 subspaces of the feature space defined in 3.1.4.2. Their respective sizes 1669 and 3566 match reasonably the original mysterious subsets with sizes 1568 and 2924 respectively. To match the carried out test as much as possible with the limited information, the found 1D, 2D, 3D, 4D descriptors were added into both sets if they were not already present. This is the case for the last term of 3D which is also the third term of 4D and it is also the case for the last term of 4D because it was made using 4 and 5 operations respectively and they do not appear naturally in the feature spaces we selected for this test (see equations (3.6) and (3.7)). Effectively, the feature space sizes are 1671 and 3567 respectively after the addition. The test was carried out for the 2D descriptor only as in [9]. In total, for every noise level in δ, 10 drafts of the random number from the uniform distribution were performed and for each draft, 90% - 10% cross validation split was caried out 50 times. This sums to a statistic of 500 prediction samples for each noise level. The results are reported in Table 3.8.

Number of features	Quantity	$\mathcal{U}(-\delta, \delta)$				
		$\delta = 0$	$\delta = 0.01$	$\delta = 0.03$	$\delta = 0.1$	$\delta = 0.2$
1671	RMSE CV [eV]	0.102	0.103	0.106	0.128	0.178
	AveMaxAE CV [eV]	0.180	0.180	0.187	0.228	0.319
	Recovery of all data 2D [%]	94	92	89.2	64	15
	Recovery of L10%OCV 2D [%]	100	96	90.4	62.8	15
3567	RMSE CV [eV]	0.130	0.129	0.131	0.148	0.185
	AveMaxAE CV [eV]	0.253	0.248	0.247	0.277	0.336
	Recovery of all data 2D [%]	42	38.8	34	11.2	1
	Recovery of L10%OCV 2D [%]	100	88	68.8	30.2	7.4

Table 3.9: RMSE, average MaxAE, recovery rate of 2D descriptor of equation (3.5) and recovery rate of leave 10% out cross validation descriptor ($\delta = 0$) for the two feature sets with different noise levels

The drop in performance of the model described by the four quantities is in accord with [9]. Understandably, the values differ because our feature spaces are different. The stability of descriptor recovery is measured by two quantities. Recovery of all data D2 means the percentage of times when the 2D descriptor of equation (3.5) was found upon cross validation and Recovery of L10%OCV D2 reports the percentage of times when the 2D descriptor of noised data was the same as the 2D descriptor of non-noised data with the same cross validation split. For the smaller feature space, the errors are in great accord with a feature space of similar size [9] where the AveMaxAE CV is in exact accord and RMSE CV differs by less than 0.01 eV on all occasion. We found the same behavior of descriptor stability for noise levels up to $\delta = 0.03$. The descriptor found by the cross validation procedure almost always matches the one in equation (3.5) for all noise levels. For the larger feature space, which is much bigger than the one in [9], we can observe trend of worse descriptor recovery success. The RMSE CV and AveMaxAE CV differs by less than 0.02 eV and 0.06 eV respectively on all ocassions even for much bigger feature space. Combined with the low recovery rates, this hints towards the fact, that many competitive models can be recovered from the pool of 30 candidates.

3.1.5 Extrapolation Capabilities of the Model

Two distinct tests were performed to evaluate how the model will hold when a certain group of materials is removed altogether from the training and reserved as a test set.

BN and C (diamond) In this test, two most stable zincblende materials (labeled in Figure 3.2) are chosen as the test set and the model is trained on the remaining 80 datapoints. It is worthwhile to note that the recovered models can be different from the ones we receive after training on the whole dataset. This effect might be amplified by the extraordinary stability of these materials. The predictions and the actual LDA energies are listed in Table 3.10.

In eV	1D	2D	3D	4D	LDA
C	1.188	1.638	1.637	2.937	2.638
BN	0.826	1.096	1.387	1.754	1.713

Table 3.10: Model performances for C and BN as the test set

In [9], only 2D descriptor model was tested. We list all the models because it illustrates the prediction improvement with increased model complexity. Particularly, the 4D descriptor performs much better than any other. The difference between our prediction with 2D descriptor and [9] differs by 0.2 eV for diamond and less than 0.3 eV for BN. All descriptors always classify diamond and BN as zincblende structures. Out of curiousity, to demonstrate leakage of information from the test set into the training procedure, we forced the 2D descriptor to be the one in equation (3.5). That is, we remove all but 2 contenders from Θ - we put

$$\Theta = \left\{ \frac{|IP(B) - EA(B)|}{r_p(A)^2}, \frac{|r_s(A) - r_p(B)|}{\exp(r_s(A))} \right\}$$

and consequently, the model is trained on 80 datapoints and prediction for diamond and BN is made. The results of absolute error of 0.032 eV for diamond and 0.130 eV for BN is surprisingly low even though they are obviously spurious.

Carbon Out In this test, all compounds containing carbon were removed from the training. Namely, it is C (diamond), SiC, GeC, SnC. The model was trained on the remaining 78 datapoints. Again, 4 models are listed in Table 3.11 whereas [9] lists only performance of the 2D descriptor model.

In eV	1D	2D	3D	4D	LDA
C	1.438	1.339	1.561	2.315	2.638
SiC	0.485	0.469	0.527	0.590	0.668
GeC	0.443	0.461	0.502	0.597	0.808
SnC	0.215	0.228	0.244	0.259	0.450

Table 3.11: Model performances for C, SiC, GeC, SnC as the test set

The 2D descriptor performance exactly matches the results in [9] for 3 compounds containing carbon. For diamond, our error is lower by 0.03 eV. Our 4D descriptor model performs excellently for diamond given the fact it has never seen carbon.

Chapter 4

Transparent Conducting Oxides Experiment

Transparent conducting oxides are a class of materials which are widely used in optical electronics, solar cells, LEDs, touch screens, lasers and more. The compound needs to be both electrically conductive and transparent to visible light to be usable for such applications. Alloys of group-III oxides have been experimentally examined to be worthwhile contenders for these use cases. Namely, it is Al_2O_3, Ga_2O_3 and In_2O_3. We shall denote an alloy with x fraction of Al_2O_3, y fraction of Ga_2O_3 and z fraction of In_2O_3 as $(Al_xGa_yIn_z)_2O_3$ and then obviously $x+y+z=1$. The problem is that the aforementioned three oxides exhibit different properties and it is unclear whether an alloy with the given composition will be stable or possess desired conductivity. The goal of this experiment is to find a regression model which accurately predicts the stability and transparency of group-III oxide alloys [35], [36].

4.1 The Dataset

In this section, the structure of the used dataset will be explained and some of its properties examined.

4.1.1 The Structure of the Dataset

The available dataset consists of the Nomad2018 Predicting Transparent Conductors Kaggle competition dataset (3000 datapoints) that was cleaned which results in 2991 distinct materials (the 9 datapoints turned out be duplicates). The data include the Cartesian coordinates of every atom in the unit cell of the material, its species and the lattice vectors of the unit cell. The two quantities to be predicted are two types of energies and they will be explained later.

Additionally, the raw text files of the DFT calculation were provided by Fritz Haber Institute of the Max Planck Society in Berlin. These files contain additional data which consists of non-converged materials. In other words, we include some of the data generated during the DFT calculation and add them into the dataset. Unfortunately, the files contain only a few last relaxation step (see Section 2.1 for an explanation of the calculation methodology). The dataset of the starting configuration (called Vegard dataset [37]) with corresponding DFT calculation results was downloaded from the NOMAD Repository [37].

The last relaxation step in the additional data is the final configuration of the material. The number of relaxation steps is not a priori known therefore every material has different amount of relaxation steps. The reason behind this is the iterative nature of the DFT relaxation procedure and convergence of the iterative algorithm. In total, this way we managed to gather additional 2991 Vegard dataset datapoints and

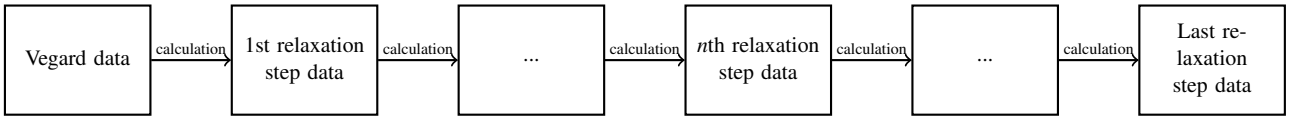


Figure 4.1: The flowchart explaining the connections of the three datasets for every material

additional 34219 relaxation dataset datapoints to the final 2991 datapoints from the Kaggle competition. To gain intuition of the behavior of the DFT calculation, an example of material is presented in Figure 4.2 where the changes in Cartesian coordinates can be seen. A barplot of the amount of relaxation steps for every material is in Figure 4.3.

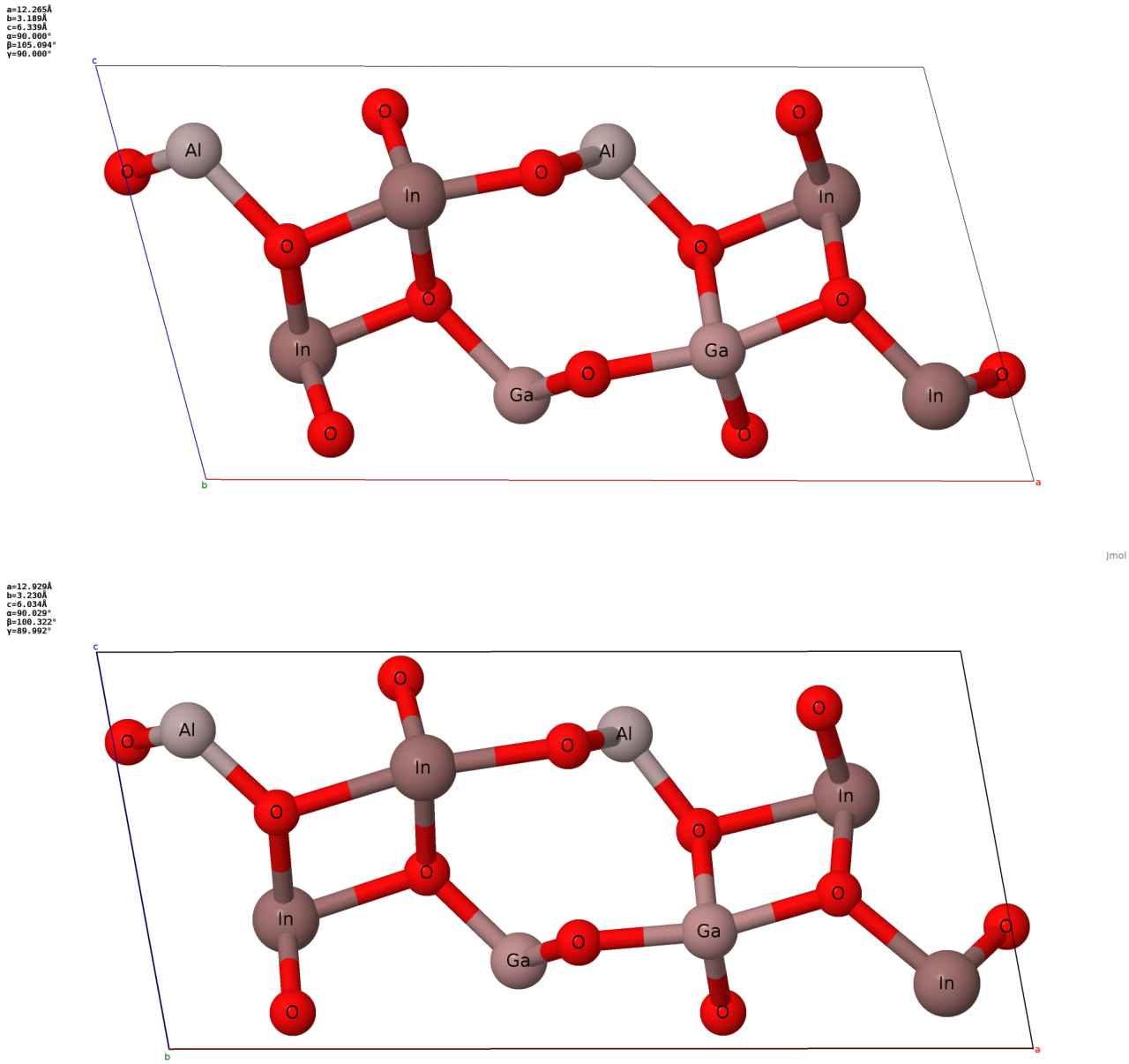


Figure 4.2: Alloy $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$ presented as a relaxation example. Top: Vegard data positions (initial positions fed into the DFT calculation). Bottom: The last relaxation data position (final data). This material has only one relaxation step - after only one iteration, the procedure converged. Notice the change in the lattice vectors as well as the change in atomic position of the oxygen and indium on the right [20]

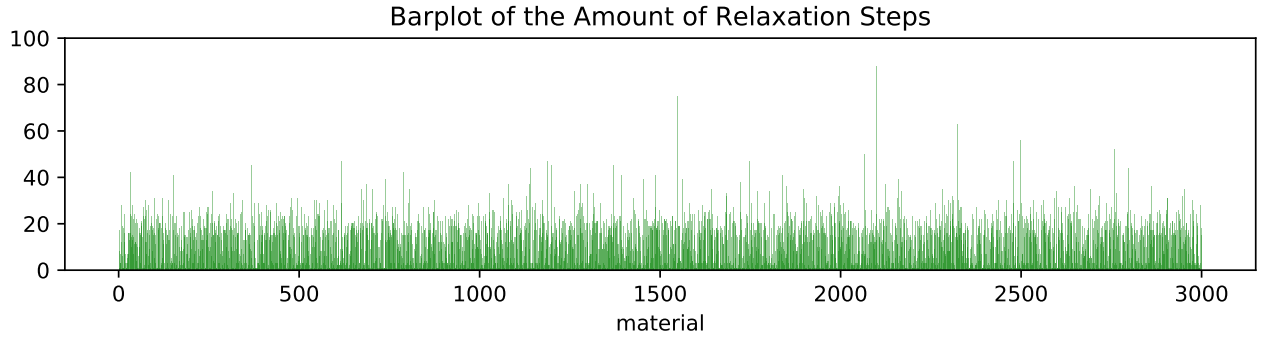


Figure 4.3: The amount of relaxation steps recovered from the additional files for the original 2991 materials plotted against the index of the material. The mean value of relaxation steps is 12.440.

A very important detail of the Kaggle competition is that the Cartesian coordinates provided in the competition are from the Vegard dataset - the initial positions before relaxation. However, the quantities to be predicted correspond to the final dataset. This is practical: a model which would need the final positions as inputs does not save any time because the calculation of the final configuration is the procedure we would like mimic with our model. However, this was not explicitly stated during the competition which could be quite troublesome for some of the participants. Also, the ability to analyze the relaxation procedure and compare the initial data with the final data can bring valuable insight which was not possible. Another important detail is that the values of the predicted quantities of the test set of 600 materials (drawn from the 2991 datapoints) was hidden from the competitors and they could interact with it only through a web interface where they submitted the prediction and automatically received the test set error. Therefore, the participants had only the 2391 unique datapoints to work with.

For coherence, our datasets were constructed to possess much more natural structure. The atomic positions correspond to the calculated energies of the given relaxation step. If need be, the atomic positions and calculated energies can be matched as seems fit for the given task (e.g. combine Vegard positions and final energies as in the competition etc.).

4.1.2 Formation Energy and Bandgap

The two physically meaningful quantities we want to predict are the formation energy and the band gap of the alloy. The formation energy is defined as the difference between the calculated DFT energy of the alloy $(Al_xGa_yIn_z)_2O_3$ and the calculated DFT energy of the pure oxides α-Al_2O_3, β-Ga_2O_3 and c-In_2O_3

$$E_f = E[(Al_xGa_yIn_z)_2O_3] - xE[Al_2O_3] - yE[Ga_2O_3] - zE[In_2O_3] \tag{4.1}$$

where $x = \frac{N_{Al}}{N_{Al}+N_{Ga}+N_{In}}$, $y = \frac{N_{Ga}}{N_{Al}+N_{Ga}+N_{In}}$ and $z = \frac{N_{In}}{N_{Al}+N_{Ga}+N_{In}}$ as is defined in [36]. The α, β and c in the name of the oxides determine the space group of the crystal for the given oxide (one oxide can crystallize in multiple space groups). The traditional definition of the formation energy is the energy of the compound minus the individual atomic energies. Therefore the difference is that the reference is the pure oxides and not individual atoms. The physical meaning of the formation energy is that the bigger the value, the higher is the stability of the alloy in terms of degradation and loss of desirable properties. It is also possible that this quantity has negative value for the given material. This means the alloy is not a stable crystal and should degrade.

The band structure of a solid material is the manifestation of the principles of the quantum mechanical behavior of electrons in crystals. The shape of the band structure determines the response of the

material to light (transparency to certain wavelengths, absorption of certain wavelengths etc.), changes in temperature. Its shape classifies to materials to metals, semiconductors and insulators. The bandgap is the shortest distance between the conduction and valence bands of the band structure and it is the energy needed to excite the electron from the top of the valence band to the bottom of the conduction band [36], [18]. We desire to predict this value to be able to determine if the material at hand would be useful for the given application.

4.2 Results and Discussion of ngram

The motivation and construction of ngram was presented in Chapter 2. We implement the ngram descriptor and analyze some unclear heuristic choices which were made during its definition and construction. Then, the proposed extension is implemented and tested on the competition dataset and the mined dataset which extends the competition dataset.

4.2.1 Analysis

The ngram descriptor won the Nomad2018 Predicting Transparent Conductors Kaggle competition because of its ability to use the positional information (lattice vectors and Cartesian coordinates) in a way which smears the fact the data are not the relaxed positions but the Vegard positions. The ngram was implemented together with kernel ridge regression. Since we possess the relaxation data to test the robustness of the condition in equation (2.10) some tests were designed.

4.2.1.1 Different Datasets with ngram

The coordination environment of every atom in the first relaxation step is compared with the coordination environment of every atom in the second relaxation step. The coordination environment of every atom in the third relaxation step is compared with the coordination environment of every atom in the second relaxation step and so on until we reach the last relaxation step (final data). In other words, pairwise comparison of the consecutive relaxation steps was done. This procedure was carried out for the combination of Shannon ionic radii used in the competition (coordination VI for metals (Al, Ga and In) and coordination II for oxygen, see Table 2.1 for details). We found out that the condition in (2.10) holds for remarkable 2635 materials out of 2991 for all available relaxation steps. The possible changes of the coordination environment of the Vegard data and the final data was tested as well and all 2991 materials undergo a change in the coordination environments between the starting Vegard data and finishing final data which means that the relaxation procedure is significant and results in changes of the descriptor. In Figure 4.4, we showcase the change of coordinations on an example of two unigrams for the Vegard and final datasets.

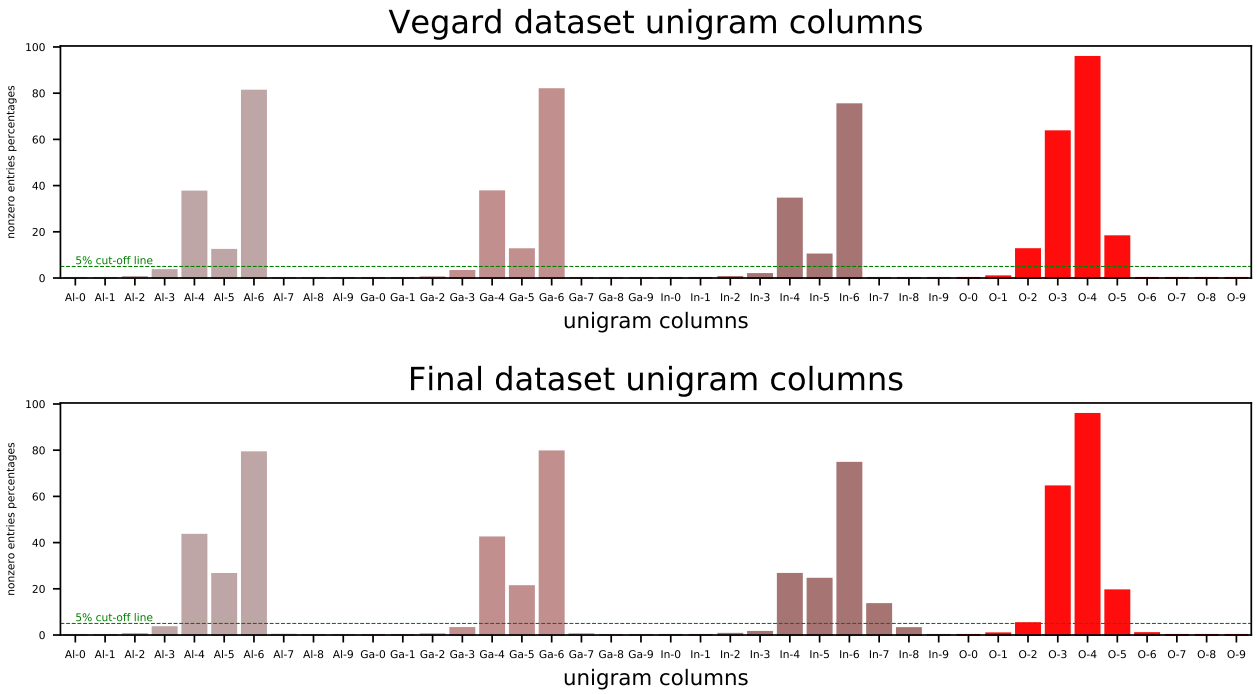


Figure 4.4: The comparison of the nonzero entries of the unigram descriptor for the vegard dataset and the final dataset. After the DFT calculation, the amount of nonzero numbers in the descriptor increases but their percentage stays very low. For indium, even coordinations In-7 and In-8 emerge. The heuristic green 5% cut-off line is drawn

The emergence of the new nonzero entries in the descriptor was interesting and we decided to test if the columns which are zero for more than 95% of the materials carry any critical descriptive information. The reduction from 21 non-empty columns of the Vegard dataset to 13 columns (the very same 13 columns as in [36]) and from 26 non-empty columns to 14 non-empty columns for the final dataset was carried out. The reduced matrices were used for training KRR models and we predicted the formation and bandgap energies. The reduction of the amount of the columns of the matrices led to a significant loss of prediction accuracy of the models and we will not discuss this reduction anymore.

4.2.1.2 The Choice of Shannon Ionic Radii

The correct choice of the values of the Shannon ionic radii is not clear since it was chosen heuristically. We get to choose these values from the ones in Table 2.1 but the best possibilities for our application are not a priori known. We can perform an exhaustive search over all combinations of values in Table 2.1. However, this would be extremely computationally expensive for the relaxation dataset (considering the cross validation procedures for both the formation energy as well as the bandgap energy of the KRR models). A worthwhile compromise was tested: we take the average of Shannon ionic radii for every

atom in Table 2.1. The resulting values are listed in (4.2).

$$\begin{aligned} R^S_{Al} &= \frac{0.39 + 0.48 + 0.535}{3} = 0.468\,\text{Å} \\ R^S_{Ga} &= \frac{0.47 + 0.55 + 0.62}{3} = 0.547\,\text{Å} \\ R^S_{In} &= \frac{0.62 + 0.8 + 0.92}{3} = 0.780\,\text{Å} \\ R^S_{O} &= \frac{1.35 + 1.36 + 1.38 + 1.4 + 1.42}{5} = 1.382\,\text{Å} \end{aligned} \tag{4.2}$$

The geometric average of the Shannon ionic radii for every atom was also considered but the geometric average and the arithmetic average vary negligibly so we test only the values from the arithmetic average.

The robustness test of relaxation steps was performed for the Shannon ionic radii averages from (4.2). The condition in (2.10) does not undergo a change throughout any consecutive relaxation steps for 1612 materials out of 2991. The robustness of the coordination environment of the Vegard data and the final configuration was tested again and all 2991 materials undergo a change in the coordination environments between these two configurations of the positional data. The Shannon ionic radii averages had overall a slightly worse performance on test data. The choice of Shannon ionic radii averages did not improve the performance of the descriptor.

Next, the combination of Shannon ionic radii VI for Al and Ga, coordination VIII for In and coordination VI for O was tested. We chose the maximum of ionic radii for metals and coordination VI for oxygen because it is the most likely coordination for oxygen. It is possible that the change of the value of the Shannon ionic radius for oxygen does not bring anything new because all the difference between the values is less than 10%. The value of the Shannon ionic radius for coordination VIII of indium is very large and introduces coordinations of indium up to 11 in the datasets. This combination of Shannon ionic radii resulted in great increase of test error and it is clear that the main culprit is the creation of the spurious coordinations up to In-11 as the result of choosing the maximal values of Shannon ionic radii. The important result of this test is that the increase of information in the coordination environment (e.g. more atoms in the environment) does not lead to the improvement of the model. We need to capture the structure of the crystal conveniently and this is done through careful consideration of what the important information actually is.

The minimal value of Shannon ionic radii for all atoms was tested and performed slightly worse. The maximal values for all atoms performed also badly.

To conclude the tests of the choices of the best way to choose the atomic neighborhood, we state that it appears that the choice II for O and VI for Al, Ga and In holds the best physical meaning for the available dataset and we will use these choices in our conducted experiments.

4.2.1.3 The Choice of Hyperparameters

It is also important to discuss the hyperparameter tuning procedure using cross validation in the KRR models. The two kernels we used were the Gaussian and Laplacian kernels from Table (1.2). The choice of the kernel does not seem to bring any substantial difference in performance in comparison with using only the Gaussian kernel which is used almost always compared to the Laplacian kernel which does not seem to be as known. The parameter grid for the λ parameter was chosen to be on a logarithmic grid. The choices for values higher that 10^0 were never chosen by the estimator during cross validation. To test how well this parameter grid manages to choose good parameters, we define a finer grid around the chosen value after a successful cross validation. It turns there areas of the parameter space where we can get up to 10-20% better performance of every dataset tested. The problem with this is the incredible

computational difficulty for big datasets. Also given the fact that we stick to the train and test split of the data as was in the competition, overfitting can be an issue. We also have unlimited number of tries for finding truly the best set of parameters which is something the competitors could not do because the amount of predictions they could submit was limited. Therefore, we note that possibly, there are some combinations of hyperparameters which perform much better than the ones chosen by the correct procedure of searching logarithmic grid and we can find better ones. However, to make the results comparable with the competition setting, we will behave like just another competitor and not perform any other grid search that the one on the logarithmic grid.

4.2.2 Modelling

This section lists the results of the ngram descriptor and its extensions. The kernel ridge regression implementation from the Python 3 scikit-learn [5] package was used. Other scikit-learn features (cross validation, data transformations etc.) were also used. Details of the computations are discussed in Appendix C.

The unigram, bigram, trigram and quadgram descriptors were generated on the Vegard, final and relaxation datasets and their performance was tested. In detail, we have 2391 train and 600 test datapoints for the Vegard and final datasets. Also, we have 29864 train and 7346 test datapoints for the relaxation dataset. The test set is roughly 20% of all data for all three datasets. The hyperparameters were optimized on a logarithmic grid using grid search. The train set is used for the validation as well - we take a subset of the train set and use it as a validation set and the rest of the train data is used for the actual training. For the Vegard and final datasets, 5-fold cross validation was used with mean squared error (1.71) as the metric for determining the best parameters on the validation set. For the relaxation dataset, group 5-fold cross validation was used with mean squared error. The groups were determined by the composition of the material - we put alloys with the same percentages of Al_2O_3, Ga_2O_3 and In_2O_3 into the same group. This was done because the structure of these materials are very similar and we risk leakage of information from the validation set into the train set during the cross validation procedure. All the data are standardized before the training using (1.68). We take special care not to leak any information from the train set into the validation set - the standardization is done on the training data and then the validation data is standardized using the mean and variance found on the training data.

4.2.2.1 Results

We report the results in charts and tables below. In total, we report 3 types of test errors (RMSE, MAE and RMSLE) from (1.71). Firstly, we report the results for our implementation of the ngram and compare it with the competition results. Then, the results of ngram with Σ and $\hat{\Sigma}$ are reported. The symbol $p \leq |k|, k \in \mathbb{Z}$ in tables means values of p which fulfill it are included in the descriptor, e.g. $p \leq |2|$ means that $\Sigma(-2)$, $\Sigma(-1)$, $\Sigma(0)$, $\Sigma(1)$, $\Sigma(2)$ are in the descriptor.

ngram The results of our implementation of the ngram descriptor compared with the competition implementation.

ngram with Σ The results for extension of ngram with the Σ descriptor. The best result in every row is highlighted.

1-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17	12.7	13.3	13.5	12.4	13.6	13.3	13.1	12.8	12.6	12.4	**12.3**	12.3
	Bandgap	116.8	118.3	114.5	112.1	109.1	106.6	**105.8**	106.1	106.5	113.4	113.6	114.1	114.2
Vegard	Formation	14.4	14	13.9	13.8	13.8	13.9	**13.3**	13.3	14	14	14	14	13.6
	Bandgap	**111.6**	111.9	112	111.9	121.7	113.7	113.1	112.4	120.4	121.2	122.1	113	112.6
Relaxation	Formation	20.1	15.5	15.4	14.9	16.4	15.4	15	**14.8**	14.8	15.8	15.5	17.9	17.5
	Bandgap	117.6	**115.7**	118.3	118.5	119	119	118.5	122.1	121.8	121.7	121.5	121	120

Table 4.1: Unigram MAE [meV]

1-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	31.9	21.1	20.4	22.6	20.5	22.5	23.5	19.9	19.3	19	18.9	**18.8**	19
	Bandgap	200.1	211.8	195.7	189.7	183.5	**179.5**	180.7	187	195	197.1	205.6	216.2	225.8
Vegard	Formation	25.9	25.8	25.7	25.8	26.2	26.9	**24.9**	24.6	27.9	28.3	28.6	28.8	24.9
	Bandgap	201.3	205.9	202.7	193.3	202.8	191.1	**189.6**	188.3	201.9	202.1	202.2	191.6	191.5
Relaxation	Formation	40.3	35.2	35.7	34.4	30.4	30.2	30.6	31.2	32	26.2	**26.1**	29.6	29.8
	Bandgap	192.3	**188.3**	197.3	196.1	195	194	192.8	199.2	198.4	197.9	197.4	196.7	195.7

Table 4.2: Unigram RMSE [meV]

1-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	23.9	16.8	16.3	17.5	15.9	16.7	16.7	16	15.5	15.2	15	**14.8**	14.8
	Bandgap	93.1	100.2	89.1	86.4	83.9	**82.3**	82.8	86.1	92.4	89.9	97.5	116.5	225.6
Vegard	Formation	20.5	20.3	20.2	20.2	20.5	20.9	19.8	**19.5**	21.3	21.6	21.8	21.9	19.7
	Bandgap	92.4	93.7	92.6	89.2	92.4	87.2	86.3	**85.6**	91.7	91.7	91.6	87.9	87.9
Relaxation	Formation	31	26.8	27.3	26.2	22.8	22.6	23	23.6	24.2	**20.4**	20.4	22.8	23
	Bandgap	88.4	**86**	90.7	89.8	89.2	88.4	87.4	90.5	90.3	90.3	90.2	89.9	89.3

Table 4.3: Unigram RMSLE [meV]

2-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.3	14	12.9	12	12.1	11.4	11	**10.6**	12.8	12.6	12.5	12.2	12.3
	Bandgap	128.1	123.9	120.2	117.6	116.2	115.1	113.9	112.4	111	**109.4**	118.7	117.2	118.2
Vegard	Formation	13.5	13.3	13.6	13.5	13.3	13.2	13.8	13.8	14.1	14	13.9	13.2	**13.1**
	Bandgap	117.5	117.2	114.1	112.3	111.6	111.2	110.9	110.3	**109.5**	120.2	120.3	119.7	128.8
Relaxation	Formation	25.1	24	22.9	21.2	19.8	19	18.1	17.3	18.8	18	17.2	16.4	**15.9**
	Bandgap	126.6	122.4	119.6	119.9	120.3	119.9	119.6	119.4	119.3	119	118.4	117.5	**116.3**

Table 4.4: Bigram MAE [meV]

2-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	35.4	25.6	21.5	20	21.3	20.1	**19.8**	20.1	20.5	20.5	20.5	24.1	30.8
	Bandgap	208.6	205.8	203.6	202.1	201.6	200.7	199.4	197.3	194.7	**191.8**	202.3	232	250.7
Vegard	Formation	24.8	24.5	24.7	24.6	24.3	**24.2**	26.8	27.5	27.7	28	28.3	24.2	24.2
	Bandgap	200.2	198.6	193.2	188.1	186.5	184.7	183.2	182	**181.2**	199.6	197.3	196.5	208.9
Relaxation	Formation	46.9	46.1	45.1	41	38.3	38	37.9	37.4	36.5	35.7	35.1	**28.6**	28.7
	Bandgap	207.4	204.9	202	200.4	198.4	197.1	195.9	194.8	193.6	192.4	191.2	189.9	**188.8**

Table 4.5: Bigram RMSE [meV]

2-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	26.3	18.9	16.9	15.7	16.3	15.4	**15.2**	15.4	16.1	16	15.9	18.8	27.6
	Bandgap	93.8	93.4	92.8	92.3	92.3	92.1	91.8	91.1	90.3	**89**	93.2	239.1	-
Vegard	Formation	19.5	19.4	19.4	19.3	**19.1**	19.1	20.5	20.9	21	21.2	21.4	19.1	19.1
	Bandgap	89.7	89.1	87.5	86.6	85.5	84.7	84	83.3	**82.6**	90.4	88.6	88.3	91.9
Relaxation	Formation	35.6	35.2	34.7	31.4	29.5	29.2	29.2	28.9	28.2	27.5	27	**22**	22
	Bandgap	110.3	108.2	102.7	96.7	91.4	91.8	90.3	88.7	87.1	85.6	84.3	83.4	**83.1**

Table 4.6: Bigram RMSLE [meV]

3-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.4	16.9	16.6	15.5	15.1	13	**12.4**	14.3	14.1	14	15.1	14.8	14.5
	Bandgap	111.3	110.2	108.4	106.7	105.2	104.1	102.7	101.5	**100.6**	109.1	109.1	108	107.2
Vegard	Formation	15.7	15.3	15.2	15.1	**15**	15.6	15.6	15.5	15.5	15.5	15.4	15.4	15.5
	Bandgap	109.9	109.3	109.3	**109.2**	114.7	113.3	112.9	117.1	116.1	115.6	110.5	109.9	120
Relaxation	Formation	24.7	26.1	25.6	25.3	25.4	23.9	22.3	21.4	20.4	19.4	18.6	17.7	**16.8**
	Bandgap	133.5	119.7	**118.1**	118.6	125.5	125.3	125.5	125.8	125.8	125.8	132	131.4	130.9

Table 4.7: Trigram MAE [meV]

3-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	29	29.1	29.2	29.7	29.8	26.7	**25.5**	30.4	30.6	30.8	27.8	27.3	26.7
	Bandgap	187.1	187.9	187.3	186.6	186.1	185.4	184.2	183	**181.4**	195.2	194.2	192.2	190.3
Vegard	Formation	27.5	27.5	27.6	27.7	27.9	27.4	27.3	27.3	27.2	27.1	27.1	27	**26.9**
	Bandgap	190.4	188.8	186.6	185	193	188.9	186.7	192.5	190.5	188.9	180.8	**180.2**	200.2
Relaxation	Formation	41	45.4	45	45.1	45.5	47.2	42.1	39	36.9	35.3	33.9	32.6	**31.6**
	Bandgap	207.5	199.6	**197.5**	197.8	198.5	198.2	198.8	199.9	201.1	202.4	204.3	203.5	202.8

Table 4.8: Trigram RMSE [meV]

3-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	22	21.9	21.9	22.1	22.1	20.2	**19.3**	22.4	22.5	22.6	20.8	20.4	20
	Bandgap	87.3	87.5	87.2	87	86.9	86.8	86.5	86.2	**85.8**	89.8	89.8	89.2	88.7
Vegard	Formation	21.1	21	21	21	21.1	21.1	21.1	21	21	20.9	20.9	**20.8**	20.8
	Bandgap	85.1	85	84.6	84.4	88.3	85.7	85	86.6	86	85.4	82	**81.6**	89.8
Relaxation	Formation	30.7	35	34.7	34.9	35.4	37.7	33.6	31.1	29.2	27.7	26.4	25.2	**24.2**
	Bandgap	89.2	88.2	**87.7**	88.3	88.3	88.7	89.3	89.8	90.2	90.6	90.1	89.8	89.7

Table 4.9: Trigram RMSLE [meV]

4-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.1	17.8	17.7	17.6	16.2	15.7	15.5	15.3	15	15.2	15.1	15	**14.9**
	Bandgap	111.4	112.5	112.8	112.8	112.6	112.3	111.7	110.8	109.5	108.6	107.5	106.3	**105.6**
Vegard	Formation	14.9	14.7	14.7	**14.6**	14.6	15.4	15.3	15.2	15.1	15	15	14.9	14.8
	Bandgap	105.7	105.9	**105.6**	118.7	117.9	117.6	116.6	115.7	114.9	114.1	113.4	112.7	112.1
Relaxation	Formation	23	22.5	22.2	**22.1**	23.8	23.9	24.2	24.7	25.2	27.1	26.2	25.6	23.6
	Bandgap	132.9	133.2	133.5	133.6	133.6	133.4	133	132.5	131.9	131.3	130.9	**130.8**	130.8

Table 4.10: Quadgram MAE [meV]

4-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	31.6	31.7	31.4	31.2	30.5	30.8	30.8	30.6	30.2	29.4	29.1	28.8	**28.3**
	Bandgap	202.8	203.4	203.6	203.2	202.4	201.7	200.8	200	198.7	197.4	195.9	194.3	**193**
Vegard	Formation	27.7	27.7	27.8	27.9	28	27.2	27.1	26.9	26.8	26.6	26.5	26.4	**26.3**
	Bandgap	187.9	187.8	**186.8**	202.2	200.4	199.3	197.7	196.3	195.1	194.1	193	192.1	191.1
Relaxation	Formation	38.6	37.8	37.5	**37.4**	41.6	41.7	42.1	42.8	43.5	49.1	47.4	46.5	39.1
	Bandgap	208.7	208.9	209.3	209.5	209.5	209.4	209	208.4	207.7	206.9	206.2	205.7	**205.4**

Table 4.11: Quadgram RMSE [meV]

4-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	23.6	23.8	23.6	23.4	22.8	23	22.9	22.7	22.4	21.9	21.6	21.4	**21.1**
	Bandgap	94.6	94.6	94.6	94.3	94	93.9	93.5	93.3	92.8	92.4	92	91.6	**91.3**
Vegard	Formation	21	20.9	21	21	21	20.8	20.7	20.6	20.5	20.4	20.3	**20.2**	20.2
	Bandgap	**87.6**	87.6	87.3	90.1	89.7	89.6	89.3	89.1	89	88.8	88.5	88.2	87.9
Relaxation	Formation	28.8	28.2	28	**27.9**	31.3	31.4	31.8	32.5	33.2	39.7	38.3	37.4	30.3
	Bandgap	91.5	91.9	92.1	92.2	92.3	92.3	92.3	92.1	92	91.8	91.6	91.5	**91.4**

Table 4.12: Quadgram RMSLE [meV]

ngram with $\hat{\Sigma}(p)$ The results for extension of ngram with the $\hat{\Sigma}(p)$ descriptor.

1-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17	14.5	13.9	13.9	12.8	11.9	**11.3**	13.4	13.1	12.8	12.5	12.3	12.1
	Bandgap	116.8	117.9	112.9	112	109.7	**108.7**	116.7	115.5	115.3	114	112.5	111.2	110.5
Vegard	Formation	14.4	**13.7**	13.9	14	14	14	14	14	14	14.2	14.1	13.9	13.8
	Bandgap	111.6	111.7	111.9	114.2	113.2	112.4	111.5	108.5	**108**	119.9	119.5	111.2	117.2
Relaxation	Formation	20.1	15.4	15.6	**15**	16.6	19.3	17.5	16.6	16	15.5	15.2	16.3	16
	Bandgap	117.6	**115.9**	118.6	118.4	118.3	117.7	122.4	121.8	121.2	120.7	120.3	119.6	118.7

Table 4.13: Unigram MAE [meV]

1-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	31.9	22.8	22.8	23.8	22.1	20.7	**19.5**	21.1	20.8	20.8	20.7	20.4	19.7
	Bandgap	200.1	201.3	190.9	190.7	**188.2**	189.1	207.7	206.5	205.6	203.2	200.5	197.5	195
Vegard	Formation	25.9	**25.2**	25.8	26.6	27.3	27.7	26.8	27.1	27.4	27.2	27	26.6	26.3
	Bandgap	201.3	202.2	202.9	193.5	189.7	187.4	185.9	**185.8**	189.5	204.9	204.1	193.7	196.4
Relaxation	Formation	40.3	34.9	36.6	34.8	31.2	35.3	32.4	32.5	32.4	32.2	31.6	26.1	**25.6**
	Bandgap	192.3	**187.9**	197.1	196.6	195.6	194.1	200.3	198.9	197.8	196.9	196	195.1	194.2

Table 4.14: Unigram RMSE [meV]

1-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	23.9	18.3	18.3	18.8	17.5	16.4	**15.4**	16.8	16.5	16.3	16.2	15.9	15.4
	Bandgap	93.1	91.4	86.9	86.8	**85.9**	85.9	95.4	95	94.8	93.8	92.8	91.6	90.5
Vegard	Formation	20.5	**19.9**	20.2	20.6	21	21.3	20.6	20.8	21	21	20.8	20.7	20.5
	Bandgap	92.4	91.7	91	88.1	86.2	84.5	83	**81**	82.4	91.8	91.3	84.3	87.1
Relaxation	Formation	31	26.5	28.2	26.6	23.5	27	23.3	22.7	22.1	21.6	21.3	20.5	**20.1**
	Bandgap	88.4	**85.6**	90.5	89.9	89.2	88.2	90.6	89.8	89.2	88.7	88.1	87.5	86.8

Table 4.15: Unigram RMSLE [meV]

2-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.3	13.7	13.4	12.6	11.8	11.1	10.8	10.6	**10.5**	12.2	12.1	12.6	12.5
	Bandgap	128.1	122.7	119.4	117.8	116.8	115.5	115	114	113.2	112	110.4	109	**108**
Vegard	Formation	13.5	**13.3**	13.4	13.6	13.4	13.3	13.3	13.6	13.5	13.4	13.4	13.3	13.3
	Bandgap	117.5	116.3	113.2	110.8	110.4	110.1	109.7	109.1	108.6	**108.4**	110.4	110.3	114.3
Relaxation	Formation	25.1	24	23.9	21.6	20.2	19.5	18.3	17.4	19	18.2	17.5	15.5	**15.1**
	Bandgap	126.6	122.1	119.4	119.3	119.5	119	119.1	118.7	118.3	117.7	117	116.2	**115.1**

Table 4.16: Bigram MAE [meV]

2-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	35.4	32	30.9	30.1	29.8	29.8	30.1	30.7	31.5	33.5	34.4	23	**22.9**
	Bandgap	208.6	204.7	202.8	201.8	200.9	199.4	198.6	197.1	195.6	193.1	190.3	187.3	**185.2**
Vegard	Formation	24.8	24.6	24.8	24.6	24.5	**24.4**	24.5	25.3	25.4	25.4	25.3	25.1	24.9
	Bandgap	200.2	197.6	192.5	185.5	185	182.8	**181.7**	182.4	185.4	189.7	189	190.6	196
Relaxation	Formation	46.9	46.2	47.5	40.2	38.5	38.6	38.4	37.7	36.9	36.3	35.7	25.8	**25.4**
	Bandgap	207.4	204.8	202	200.2	198.5	196.8	196.1	194.8	193.6	192.6	191.6	190.7	**189.8**

Table 4.17: Bigram RMSE [meV]

2-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	26.3	24.2	22.6	22	21.6	21.6	21.8	22.2	22.8	24.5	25.2	17.3	**17.2**
	Bandgap	93.8	92.5	91.9	91.8	91.7	91.2	90.9	90.4	89.9	88.8	87.6	86.4	**85.5**
Vegard	Formation	19.5	19.4	19.5	19.3	**19.2**	19.2	19.2	19.7	19.7	19.7	19.7	19.6	19.5
	Bandgap	89.7	88.6	86.6	84.3	83.8	82.4	81.1	**80.1**	80.1	81.4	82.2	82.5	83.5
Relaxation	Formation	35.6	35.3	35.5	31	29.8	29.9	29.9	29.5	28.7	28	27.5	20	**19.6**
	Bandgap	110.3	108	102.4	95.9	91.2	88.4	89.9	88.3	86.9	85.8	85	84.4	**84**

Table 4.18: Bigram RMSLE [meV]

3-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.4	16.8	16.5	15.5	13.4	12.9	12.5	**12.2**	14.1	15.2	15	14.9	14.7
	Bandgap	111.3	109.8	108.2	106.5	105.1	103.6	102.4	**101.3**	108.3	108.2	108.4	108.3	108.2
Vegard	Formation	15.7	15.3	15.1	15	15	15.6	15.5	15.4	15.4	15.3	15.3	15.2	**12.6**
	Bandgap	109.9	109.2	109	108.9	108.9	112.8	112.3	111.8	124.3	123.2	122.2	121.4	**108.1**
Relaxation	Formation	24.7	26.2	25.7	25.3	25.3	23.3	21.6	20.5	19.6	18.6	17.9	17.3	**16.8**
	Bandgap	133.5	120	**118.5**	119.1	119.7	125.1	125	124.8	124.7	124.7	124.9	124.9	124.8

Table 4.19: Trigram MAE [meV]

3-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	29	29	29.2	29.5	27.2	26.2	25.4	**24.9**	30	26.7	26.3	25.8	25.4
	Bandgap	187.1	187.3	186.9	186.3	185.6	184.3	183	**181.4**	194.6	194.4	194.3	193.6	192.9
Vegard	Formation	27.5	27.5	27.7	27.8	28	27.5	27.4	27.4	27.3	**27.1**	27.1	27	23.3
	Bandgap	190.4	189.3	187.4	185.3	183.2	188.8	186.5	184.7	207.3	205.9	204.5	203.1	**181.5**
Relaxation	Formation	41	45.4	44.9	44.8	44.9	45.9	40.8	37.8	35.6	33.8	32.4	31.5	**31**
	Bandgap	207.5	200.9	198.8	198.8	199.8	**197.8**	197.9	198.6	199.8	201.4	203.4	204.8	204.9

Table 4.20: Trigram RMSE [meV]

3-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	22	21.8	21.8	22	20.7	19.9	19.2	**18.8**	22	20	19.7	19.4	19.1
	Bandgap	87.3	87.2	87	86.9	86.9	86.6	86.3	**85.8**	89.9	89.9	90.2	90.1	89.9
Vegard	Formation	21.1	21	21	21.1	21.2	21.1	21	21	20.9	20.9	20.8	20.8	**18.2**
	Bandgap	85.1	85	84.4	83.7	83	84.5	83.6	**82.6**	91.8	91.3	90.7	90.1	80
Relaxation	Formation	30.7	35	34.7	34.8	35	36.8	32.5	30	28.2	26.5	25.2	24.2	**23.6**
	Bandgap	89.2	88.7	**88.1**	88.4	89.2	88.6	89	89.5	90	90.6	91.2	91.7	91.8

Table 4.21: Trigram RMSLE [meV]

4-g MAE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	17.1	17.8	17.6	17.6	17.5	16	15.8	15.6	15.4	15.4	15.3	15.1	**15**
	Bandgap	111.4	112.5	113	113.1	112.9	112.4	111.7	110.7	109.9	109.3	108.7	108.3	**108.1**
Vegard	Formation	14.9	14.7	**14.6**	14.6	14.6	15.4	15.2	15.1	14.9	14.8	14.7	14.6	14.6
	Bandgap	**105.7**	105.9	105.5	118.3	117.4	116.4	116.3	115.4	114.7	114.1	113.4	112.9	112.6
Relaxation	Formation	23	22.5	22.2	**22.1**	23.8	23.7	23.8	23.9	24.2	25.6	24.7	24.1	22.9
	Bandgap	132.9	133.2	133.4	133.5	133.3	133	132.5	131.9	131.2	130.6	130	129.6	**129.2**

Table 4.22: Quadgram MAE [meV]

4-g RMSE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	31.6	31.7	31.4	31.2	31	29.8	29.4	29	28.6	28.2	27.9	27.5	**27.1**
	Bandgap	202.8	203.4	203.6	203.3	202.8	201.7	200.7	199.6	198.4	197.3	196.4	195.8	**195.4**
Vegard	Formation	27.7	27.7	27.8	27.9	28	27.1	26.9	26.8	26.6	26.5	26.4	26.3	**26.2**
	Bandgap	187.9	188.2	187.3	202.8	201.2	199.6	199.1	197.8	196.7	195.8	194.7	193.8	**192.9**
Relaxation	Formation	38.6	37.8	37.5	**37.4**	41.4	41.1	41.1	41.3	41.6	46.3	44.8	43.8	37.9
	Bandgap	208.7	208.9	209.2	209.3	209.3	209	208.4	207.7	206.7	205.7	204.7	203.8	**203**

Table 4.23: Quadgram RMSE [meV]

4-g RMSLE	in meV	$p\leq\|0\|$	$p\leq\|1\|$	$p\leq\|2\|$	$p\leq\|3\|$	$p\leq\|4\|$	$p\leq\|5\|$	$p\leq\|6\|$	$p\leq\|7\|$	$p\leq\|8\|$	$p\leq\|9\|$	$p\leq\|10\|$	$p\leq\|11\|$	$p\leq\|12\|$
Final	Formation	23.6	23.8	23.6	23.4	23.3	22.3	22	21.7	21.4	21.1	20.8	20.6	**20.3**
	Bandgap	94.6	94.7	94.7	94.7	94.7	94.4	94.2	93.9	93.6	93.4	**93.1**	93.1	93.1
Vegard	Formation	21	20.9	21	21	21	20.7	20.5	20.4	20.3	20.2	20.1	20.1	**20**
	Bandgap	87.6	87.8	**87.5**	90.4	90.1	89.8	89.9	89.6	89.3	89	88.6	88.3	88
Relaxation	Formation	28.8	28.2	28	**27.9**	31.2	31	31.1	31.3	31.7	37.2	36	35	29.5
	Bandgap	91.5	91.8	92.1	92.2	92.2	92.1	91.9	91.7	91.4	91.1	90.7	90.3	**90**

Table 4.24: Quadgram RMSLE [meV]

Comparison of ngram with or without Σ or $\hat{\Sigma}$ Graphical comparison of the results for ngram with or without the extension Σ or $\hat{\Sigma}$.

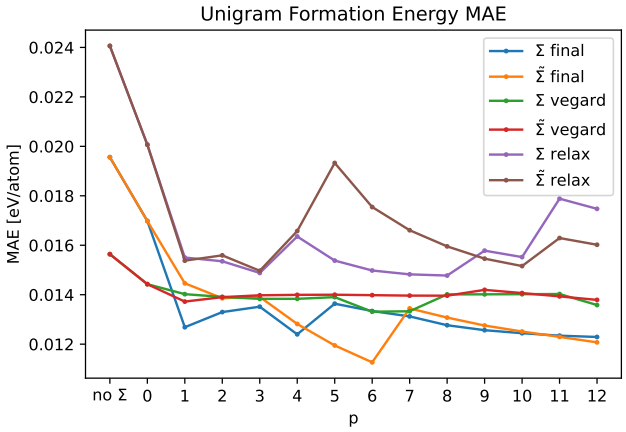


Figure 4.5

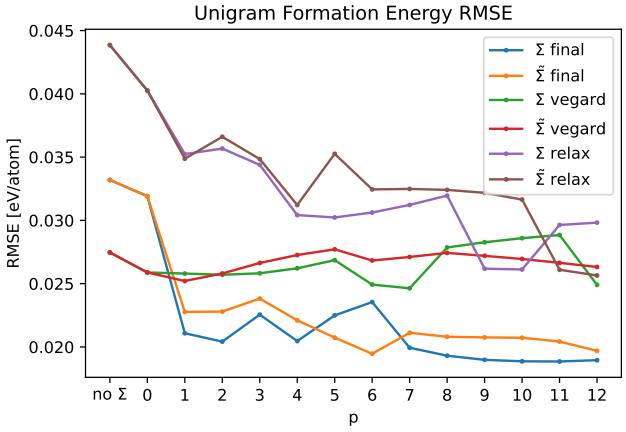


Figure 4.6

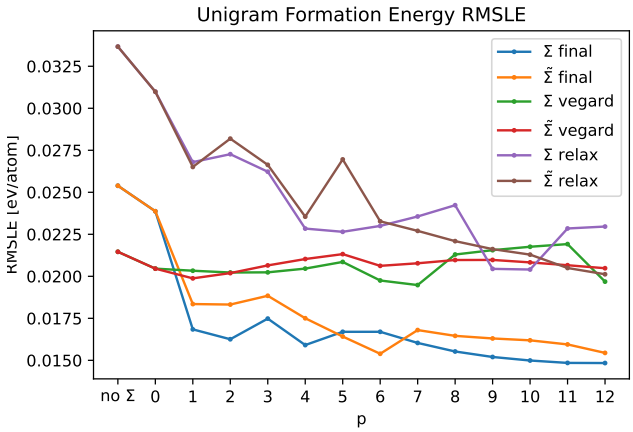


Figure 4.7

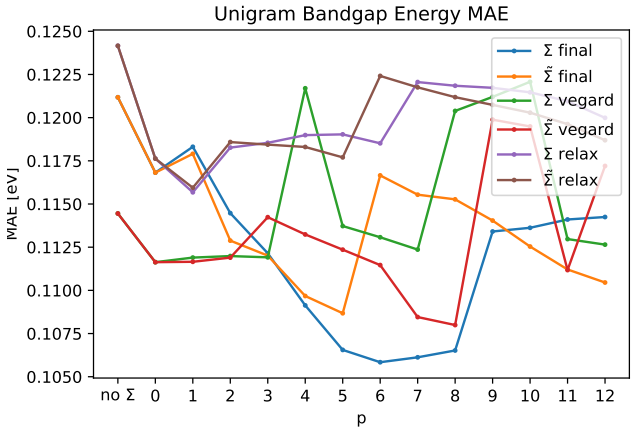


Figure 4.8

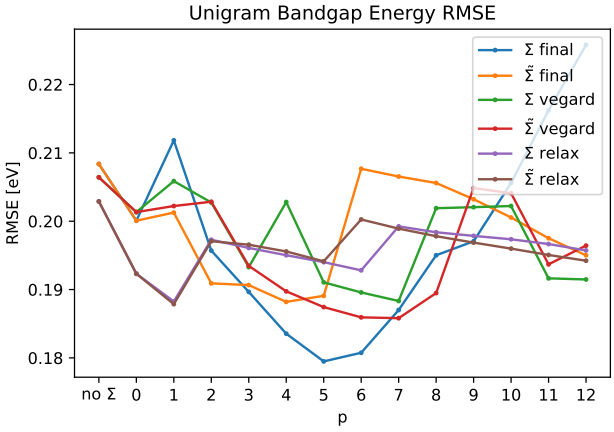


Figure 4.9

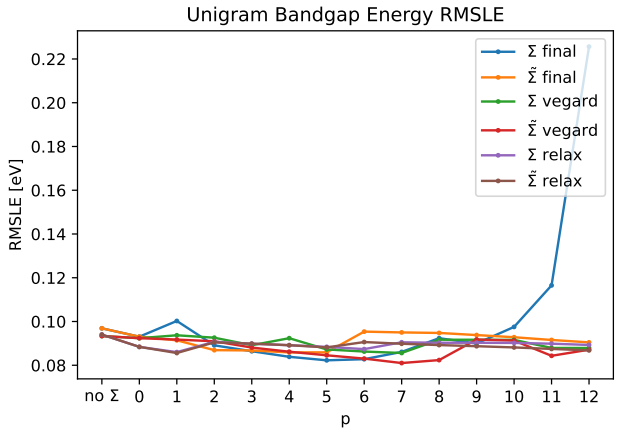


Figure 4.10

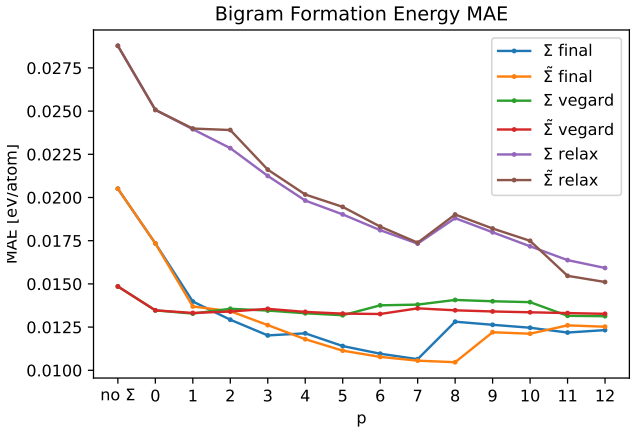


Figure 4.11

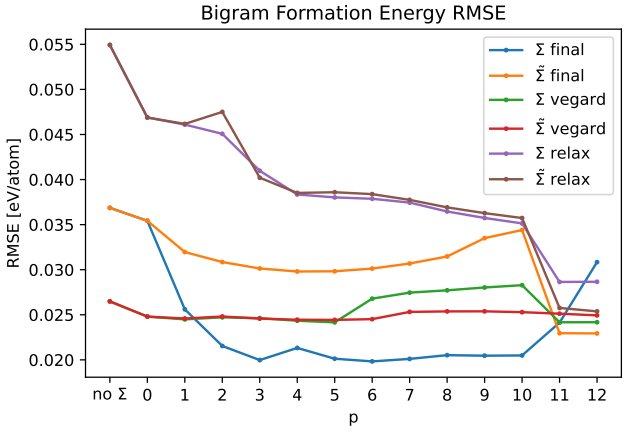


Figure 4.12

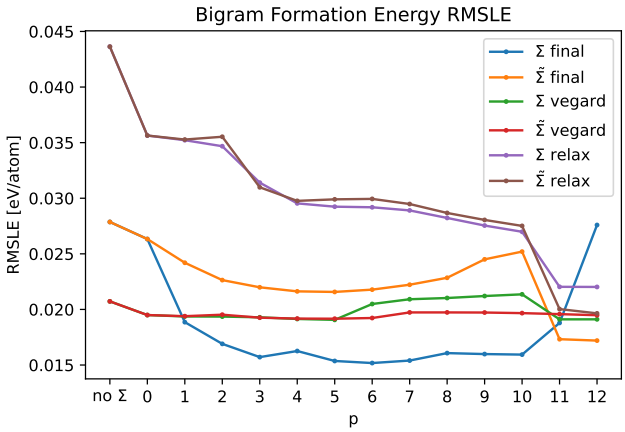


Figure 4.13

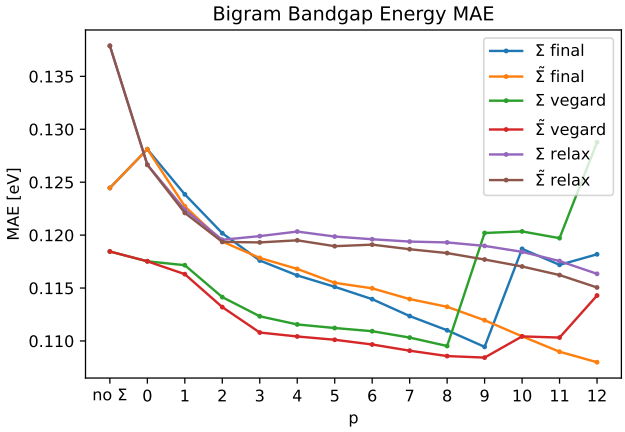


Figure 4.14

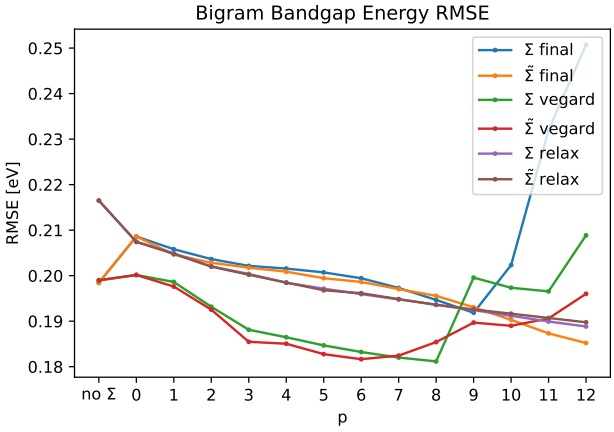


Figure 4.15

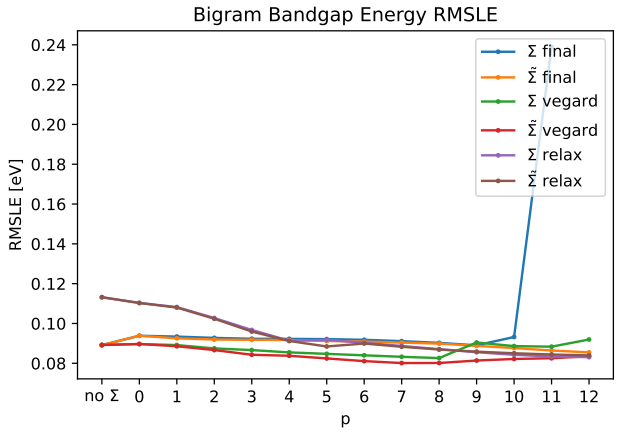


Figure 4.16

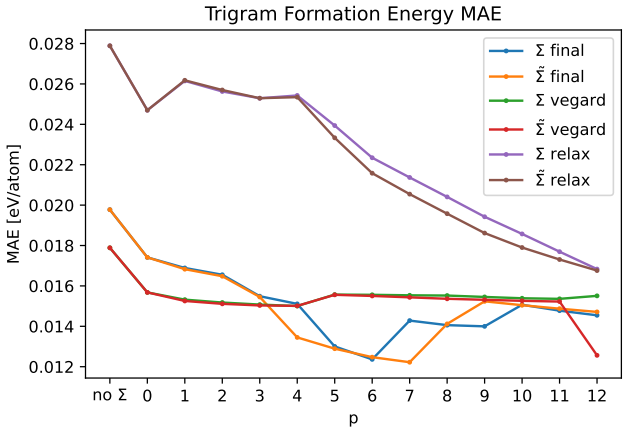


Figure 4.17

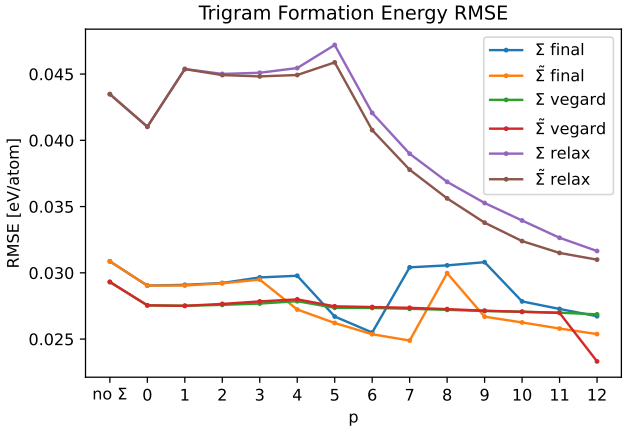


Figure 4.18

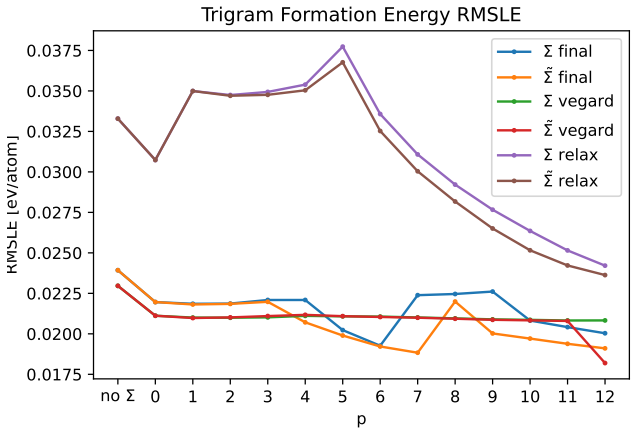


Figure 4.19

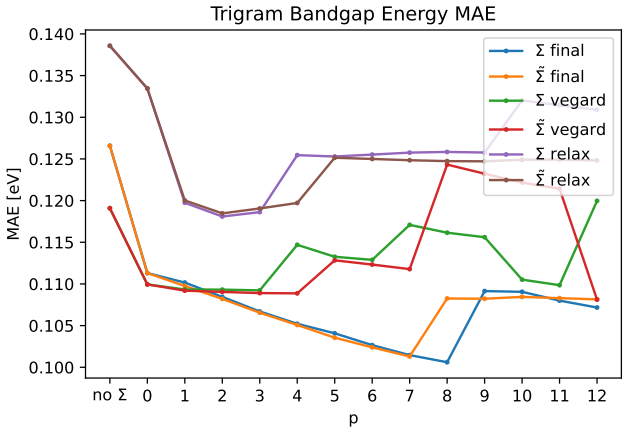


Figure 4.20

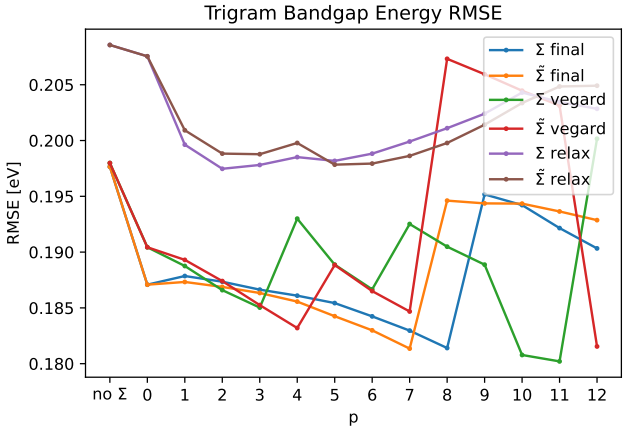


Figure 4.21

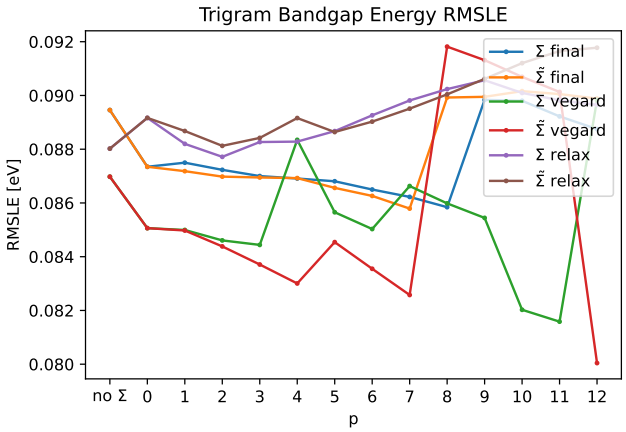


Figure 4.22

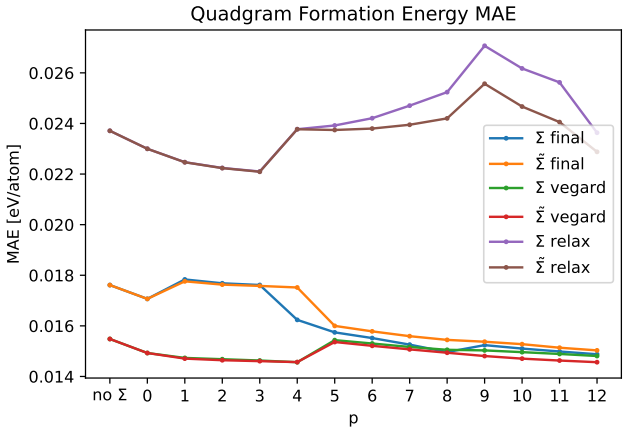


Figure 4.23

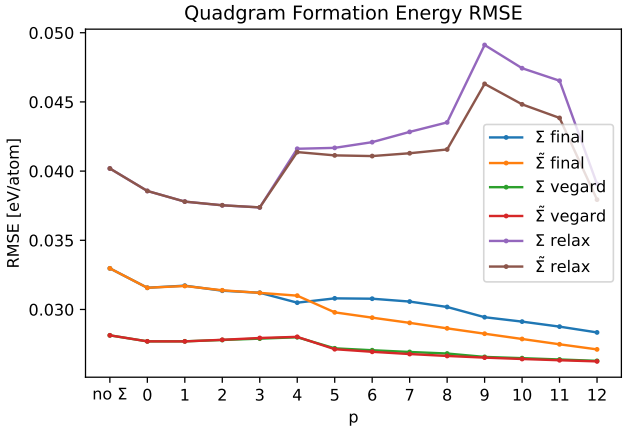


Figure 4.24

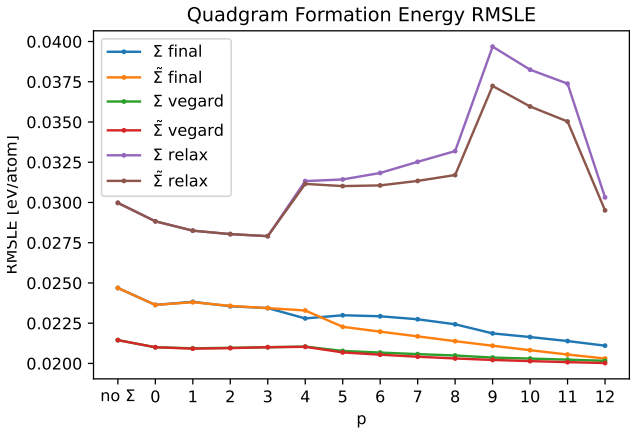


Figure 4.25

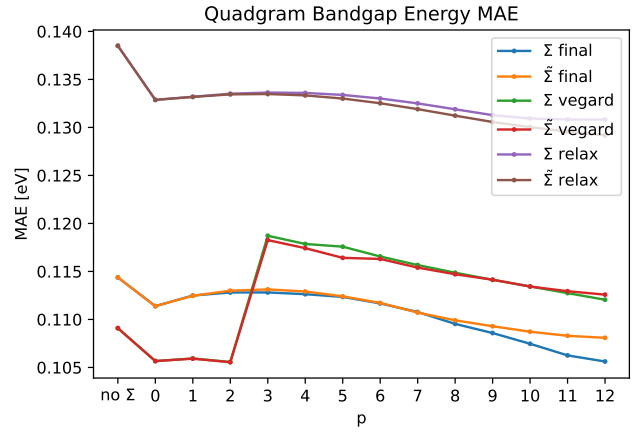


Figure 4.26

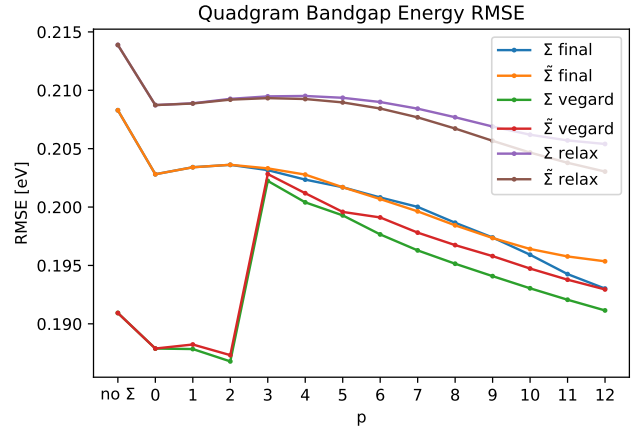


Figure 4.27

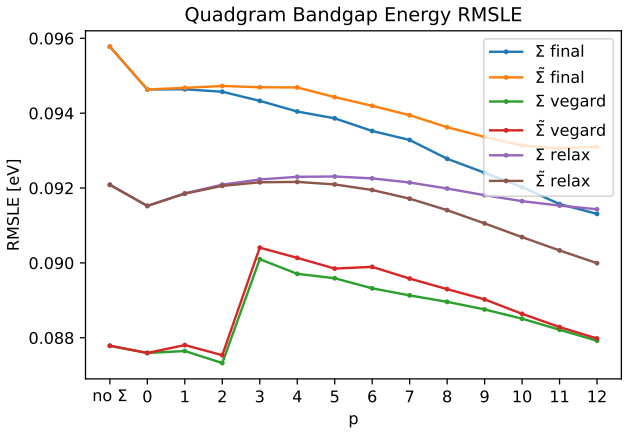

Quadgram Bandgap Energy RMSLE
RMSLE [eV]
0.096
0.094
0.092
0.090
0.088
no Σ
0
1
2
3
4
5
6
7
8
9
10
11
12
p
Σ final
Σ̃ final
Σ vegard
Σ̃ vegard
Σ relax
Σ̃ relax

Figure 4.28

Conclusion

something

Appendix A

The Rocksalt-Zincblende Classification Dataset

Z	Name	IP	EA	EN	HOMO	LUMO	r_s	r_p	r_d
3	Li	-5.329	-0.698	3.014	-2.874	-0.978	1.652	1.995	6.93
4	Be	-9.459	0.631	4.414	-5.6	-2.098	1.078	1.211	2.877
5	B	-8.19	-0.107	4.149	-3.715	2.248	0.805	0.826	1.946
6	C	-10.852	-0.872	5.862	-5.416	1.992	0.644	0.63	1.631
7	N	-13.585	-1.867	7.726	-7.239	3.057	0.539	0.511	1.54
8	O	-16.433	-3.006	9.72	-9.197	2.541	0.462	0.427	2.219
9	F	-19.404	-4.273	11.839	-11.294	1.251	0.406	0.371	1.428
11	Na	-5.223	-0.716	2.969	-2.819	-0.718	1.715	2.597	6.566
12	Mg	-8.037	0.693	3.672	-4.782	-1.358	1.33	1.897	3.171
13	Al	-5.78	-0.313	3.046	-2.784	0.695	1.092	1.393	1.939
14	Si	-7.758	-0.993	4.375	-4.163	0.44	0.938	1.134	1.89
15	P	-9.751	-1.92	5.835	-5.596	0.183	0.826	0.966	1.771
16	S	-11.795	-2.845	7.32	-7.106	0.642	0.742	0.847	2.366
17	Cl	-13.902	-3.971	8.936	-8.7	0.574	0.679	0.756	1.666
19	K	-4.433	-0.621	2.527	-2.426	-0.697	2.128	2.443	1.785
20	Ca	-6.428	0.304	3.062	-3.864	-2.133	1.757	2.324	0.679
29	Cu	-8.389	-1.638	5.014	-4.856	-0.641	1.197	1.68	2.576
30	Zn	-10.136	1.081	4.527	-6.217	-1.194	1.099	1.547	2.254
31	Ga	-5.818	-0.108	2.963	-2.732	0.13	0.994	1.33	2.163
32	Ge	-7.567	-0.949	4.258	-4.046	2.175	0.917	1.162	2.373
33	As	-9.262	-1.839	5.551	-5.341	0.064	0.847	1.043	2.023
34	Se	-10.946	-2.751	6.848	-6.654	1.316	0.798	0.952	2.177
35	Br	-12.65	-3.739	8.194	-8.001	0.708	0.749	0.882	1.869
37	Rb	-4.289	-0.59	2.44	-2.36	-0.705	2.24	3.199	1.96
38	Sr	-6.032	0.343	2.844	-3.641	-1.379	1.911	2.548	1.204
47	Ag	-8.058	-1.667	4.862	-4.71	-0.479	1.316	1.883	2.968
48	Cd	-9.581	0.839	4.371	-5.952	-1.309	1.232	1.736	2.604
49	In	-5.537	-0.256	2.897	-2.697	0.368	1.134	1.498	3.108
50	Sn	-7.043	-1.039	4.041	-3.866	0.008	1.057	1.344	2.03
51	Sb	-8.468	-1.847	5.158	-4.991	0.105	1.001	1.232	2.065
52	Te	-9.867	-2.666	6.266	-6.109	0.099	0.945	1.141	1.827
53	I	-11.257	-3.513	7.385	-7.236	0.213	0.896	1.071	1.722
55	Cs	-4.006	-0.57	2.288	-2.22	-0.548	2.464	3.164	1.974
56	Ba	-5.516	0.278	2.619	-3.346	-2.129	2.149	2.632	1.351

Table A.1: The data for the 34 elements as presented in [11]

Z_A	Z_B	A	B	$E(RS)-E(ZB)$	$E(ZB)-E(WZ)$
3	9	Li	F	-0.059	0.011
3	17	Li	Cl	-0.038	0.005
3	35	Li	Br	-0.033	0.003
3	53	Li	I	-0.022	0.002
4	8	Be	O	0.43	0.011
4	16	Be	S	0.506	-0.004
4	34	Be	Se	0.495	-0.004
4	52	Be	Te	0.466	-0.004
5	7	B	N	1.713	-0.014
5	15	B	P	1.02	-0.008
5	33	B	As	0.879	-0.006
6	6	C	C	2.638	-0.024
11	9	Na	F	-0.146	0.011
11	17	Na	Cl	-0.133	0.007
11	35	Na	Br	-0.127	0.005
11	53	Na	I	-0.115	0.004
12	8	Mg	O	-0.178	0.03
12	16	Mg	S	-0.087	0.009
12	34	Mg	Se	-0.055	0.006
12	52	Mg	Te	-0.005	0.002
13	7	Al	N	0.072	0.025
13	15	Al	P	0.219	-0.002
13	33	Al	As	0.212	-0.003
13	51	Al	Sb	0.15	-0.005
14	6	Si	C	0.668	0.003
14	14	Si	Si	0.275	-0.009
19	9	K	F	-0.146	0.01
19	17	K	Cl	-0.165	0.007
19	35	K	Br	-0.166	0.007
19	53	K	I	-0.168	0.006
20	8	Ca	O	-0.266	0.04
20	16	Ca	S	-0.369	0.024
20	34	Ca	Se	-0.361	0.02
20	52	Ca	Te	-0.35	0.014
29	9	Cu	F	-0.019	-0.007
29	17	Cu	Cl	0.156	0
29	35	Cu	Br	0.152	-0.001
29	53	Cu	I	0.203	-0.002
30	8	Zn	O	0.102	0.008
30	16	Zn	S	0.275	-0.002
30	34	Zn	Se	0.259	-0.004
⋮	⋮	⋮	⋮	⋮	⋮

Z_A	Z_B	A	B	$E(RS)-E(ZB)$	$E(ZB)-E(WZ)$
⋮	⋮	⋮	⋮	⋮	⋮
30	52	Zn	Te	0.241	-0.005
31	7	Ga	N	0.433	0.009
31	15	Ga	P	0.341	-0.008
31	33	Ga	As	0.271	-0.011
31	51	Ga	Sb	0.158	-0.011
32	32	Ge	Ge	0.202	-0.015
37	9	Rb	F	-0.136	0.008
37	17	Rb	Cl	-0.161	0.007
37	35	Rb	Br	-0.164	0.007
37	53	Rb	I	-0.169	0.006
38	8	Sr	O	-0.221	0.035
38	16	Sr	S	-0.369	0.026
38	34	Sr	Se	-0.375	0.023
38	52	Sr	Te	-0.381	0.017
47	9	Ag	F	-0.156	0.001
47	17	Ag	Cl	-0.044	0.003
47	35	Ag	Br	-0.03	0.002
47	53	Ag	I	0.037	0
48	8	Cd	O	-0.087	0.011
48	16	Cd	S	0.07	0.002
48	34	Cd	Se	0.083	-0.001
48	52	Cd	Te	0.113	-0.004
49	7	In	N	0.15	0.013
49	15	In	P	0.17	-0.005
49	33	In	As	0.122	-0.007
49	51	In	Sb	0.08	-0.01
50	50	Sn	Sn	0.016	-0.014
5	51	B	Sb	0.581	-0.001
55	9	Cs	F	-0.112	0.006
55	17	Cs	Cl	-0.152	0.006
55	35	Cs	Br	-0.158	0.006
55	53	Cs	I	-0.165	0.005
56	8	Ba	O	-0.095	0.018
56	16	Ba	S	-0.326	0.024
56	34	Ba	Se	-0.35	0.023
56	52	Ba	Te	-0.381	0.019
32	6	Ge	C	0.808	0
50	6	Sn	C	0.45	0.007
32	14	Ge	Si	0.264	-0.011
50	14	Sn	Si	0.136	-0.01
50	32	Sn	Ge	0.087	-0.013

Table A.2: The data for the 82 binary materials as presented in [11]

Appendix B

Example of ngram, Σ and $\hat{\Sigma}$ Construction

We show the construction of the ngram step by step for better undestanding on an alloy $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$. The space group of this material is 12, α = 1.4 and there are 20 atoms in the unit cell: 2 Al, 2 Ga, 4 In, 12 O. The Cartesian positions of the atoms and the fractional coordinates of the atoms are:

	species	x [Å]	y [Å]	z [Å]		species	L1	L2	L3
1	In	11.635	-0.031	0.982	1	In	0.914	-0.015	0.164
2	Al	0.274	-0.036	4.764	2	Al	0.089	-0.015	0.802
3	Ga	5.229	1.578	1.317	3	Ga	0.423	0.485	0.222
4	Al	6.812	1.587	4.707	4	Al	0.594	0.485	0.793
5	In	10.125	1.591	3.779	5	In	0.837	0.485	0.636
6	In	1.744	1.574	1.94	6	In	0.163	0.485	0.327
7	In	3.8	-0.032	4.206	7	In	0.353	-0.015	0.708
8	Ga	8.078	-0.03	1.761	8	Ga	0.65	-0.014	0.295
9	O	9.727	-0.021	5.093	9	O	0.824	-0.015	0.856
10	O	1.848	-0.045	0.545	10	O	0.151	-0.015	0.092
11	O	3.464	1.586	5.536	11	O	0.346	0.485	0.933
12	O	8.285	1.58	0.55	12	O	0.649	0.485	0.092
13	O	10.011	-0.029	2.322	13	O	0.807	-0.015	0.39
14	O	1.609	-0.037	3.535	14	O	0.174	-0.015	0.595
15	O	3.855	1.578	2.635	15	O	0.336	0.485	0.444
16	O	7.928	1.584	3.256	16	O	0.659	0.484	0.548
17	O	-0.618	1.577	4.586	17	O	0.017	0.485	0.773
18	O	12.527	1.589	1.829	18	O	0.995	0.485	0.307
19	O	5.963	-0.029	4.531	19	O	0.525	-0.015	0.762
20	O	6.128	-0.035	1.592	20	O	0.496	-0.015	0.267

The lattice vectors:

	x [Å]	y [Å]	z [Å]
$\boldsymbol{a}$	12.929	0.019	0.011
$\boldsymbol{b}$	-0.004	3.23	-0.01
$\boldsymbol{c}$	-1.086	0.014	5.935

All values were rounded to 3 decimal places for better readability. The unit cell of this material viewed in the direction of the lattice vector $\boldsymbol{b}$ is in Figure B.1. The following three steps capture the construction of the ngram descriptor of the chosen crystal.

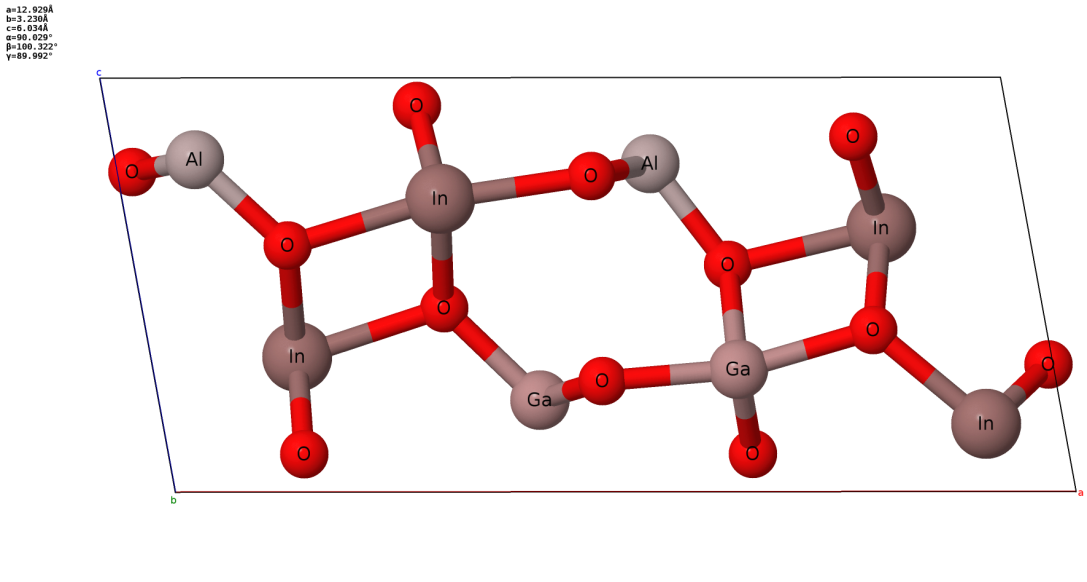


Figure B.1: The unit cell of $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$ as viewed along the lattice vector $\boldsymbol{b}$ [20]

1. The distance matrix $\boldsymbol{D}$ is calculated. If the reduced coordinates are available, the procedure follows the steps outlined in Chapter 2. If we have only the Cartesian coordinates and the lattice vectors, the reduced coordinates can be calculated as $\boldsymbol{r} = \boldsymbol{A}^{-1}\boldsymbol{R}$ for every atom in the unit cell. The corresponding distance matrix has the form (the order of the rows/columns is B.1 in accord with the position of the atoms tables above):

$$\boldsymbol{D} = \begin{pmatrix}
0 & 3.411 & 6.613 & 4.628 & 3.549 & 3.577 & 6.034 & 3.641 & 2.001 & 3.17 & 6.216 & 3.741 & 2.106 & 3.873 & 5.65 & 4.639 & 3.327 & 2.033 & 5.169 & 5.541 \\
3.411 & 0 & 4.879 & 6.59 & 3.612 & 3.53 & 3.569 & 5.945 & 3.491 & 1.785 & 3.66 & 6.448 & 4.026 & 1.815 & 4.467 & 5.718 & 1.852 & 3.422 & 5.694 & 5.51 \\
6.613 & 4.879 & 0 & 3.688 & 5.48 & 3.54 & 3.469 & 3.302 & 6.085 & 3.828 & 1.846 & 3.151 & 5.143 & 4.541 & 1.904 & 3.323 & 5.456 & 5.653 & 3.653 & 1.867 \\
4.628 & 6.59 & 3.688 & 0 & 3.441 & 5.774 & 3.456 & 3.398 & 3.351 & 6.509 & 3.449 & 1.82 & 4.305 & 5.575 & 3.61 & 1.83 & 5.5 & 5.548 & 1.834 & 3.578 \\
3.549 & 3.612 & 5.48 & 3.441 & 0 & 4.901 & 6.544 & 3.3 & 2.117 & 4.757 & 6.512 & 3.717 & 2.182 & 4.703 & 6.374 & 2.259 & 2.334 & 3.094 & 4.528 & 4.838 \\
3.577 & 3.53 & 3.54 & 5.774 & 4.901 & 0 & 3.455 & 6.536 & 5.035 & 2.139 & 3.653 & 6.54 & 4.949 & 2.27 & 2.222 & 6.322 & 3.527 & 2.149 & 5.203 & 4.682 \\
6.034 & 3.569 & 3.469 & 3.456 & 6.544 & 3.455 & 0 & 4.73 & 5.993 & 3.795 & 2.122 & 4.404 & 6.491 & 2.291 & 2.25 & 4.534 & 4.717 & 5.091 & 2.188 & 3.5 \\
3.641 & 5.945 & 3.302 & 3.398 & 3.3 & 6.536 & 4.73 & 0 & 3.717 & 6.348 & 4.437 & 2.026 & 2.013 & 6.702 & 4.603 & 2.205 & 5.349 & 4.735 & 3.328 & 1.958 \\
2.001 & 3.491 & 6.085 & 3.351 & 2.117 & 5.035 & 5.993 & 3.717 & 0 & 4.203 & 6.481 & 3.307 & 2.785 & 5.054 & 6.563 & 3.031 & 3.088 & 3.565 & 3.805 & 5.021 \\
3.17 & 1.785 & 3.828 & 6.509 & 4.757 & 2.139 & 3.795 & 6.348 & 4.203 & 0 & 3.286 & 6.638 & 5.082 & 2.999 & 3.321 & 6.799 & 2.841 & 3.047 & 5.555 & 4.406 \\
6.216 & 3.66 & 1.846 & 3.449 & 6.512 & 3.653 & 2.122 & 4.437 & 6.481 & 3.286 & 0 & 3.854 & 6.307 & 3.174 & 2.927 & 4.977 & 4.191 & 5.364 & 3.14 & 3.006 \\
3.741 & 6.448 & 3.151 & 1.82 & 3.717 & 6.54 & 4.404 & 2.026 & 3.307 & 6.638 & 3.854 & 0 & 2.949 & 6.527 & 4.896 & 2.73 & 5.45 & 4.431 & 2.824 & 2.889 \\
2.106 & 4.026 & 5.143 & 4.305 & 2.182 & 4.949 & 6.491 & 2.013 & 2.785 & 5.082 & 6.307 & 2.949 & 0 & 4.69 & 6.37 & 2.796 & 3.621 & 3.031 & 4.611 & 3.952 \\
3.873 & 1.815 & 4.541 & 5.575 & 4.703 & 2.27 & 2.291 & 6.702 & 5.054 & 2.999 & 3.174 & 6.527 & 4.69 & 0 & 2.909 & 6.529 & 2.945 & 3.094 & 4.466 & 4.918 \\
5.65 & 4.467 & 1.904 & 3.61 & 6.374 & 2.222 & 2.25 & 4.603 & 6.563 & 3.321 & 2.927 & 4.896 & 6.37 & 2.909 & 0 & 4.12 & 4.88 & 4.334 & 3.258 & 2.976 \\
4.639 & 5.718 & 3.323 & 1.83 & 2.259 & 6.322 & 4.534 & 2.205 & 3.031 & 6.799 & 4.977 & 2.73 & 2.796 & 6.529 & 4.12 & 0 & 4.584 & 4.816 & 2.844 & 2.938 \\
3.327 & 1.852 & 5.456 & 5.5 & 2.334 & 3.527 & 4.717 & 5.349 & 3.088 & 2.841 & 4.191 & 5.45 & 3.621 & 2.945 & 4.88 & 4.584 & 0 & 2.777 & 6.552 & 6.575 \\
2.033 & 3.422 & 5.653 & 5.548 & 3.094 & 2.149 & 5.091 & 4.735 & 3.565 & 3.047 & 5.364 & 4.431 & 3.031 & 3.094 & 4.334 & 4.816 & 2.777 & 0 & 6.567 & 6.607 \\
5.169 & 5.694 & 3.653 & 1.834 & 4.528 & 5.203 & 2.188 & 3.328 & 3.805 & 5.555 & 3.14 & 2.824 & 4.611 & 4.466 & 3.258 & 2.844 & 6.552 & 6.567 & 0 & 2.944 \\
5.541 & 5.51 & 1.867 & 3.578 & 4.838 & 4.682 & 3.5 & 1.958 & 5.021 & 4.406 & 3.006 & 2.889 & 3.952 & 4.918 & 2.976 & 2.938 & 6.575 & 6.607 & 2.944 & 0
\end{pmatrix}$$

2. The coordination of every atom j is determined by the number of times the expression

$$\boldsymbol{D}_{ij} < \alpha(R_i^S + R_j^S)$$

holds for all $i \neq j$ where only metal/oxygen pairs are considered (as is in detail explained in Chapter 2). This results in the following:

X-n	In-3	Al-3	Ga-3	Al-3	In-4	In-4	In-4	Ga-4	O-2	O-2	O-2	O-2	O-3	O-3	O-3	O-3	O-2	O-2	O-2	O-2

Table B.1: The atom-coordination pairs of $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$

We can see from the Table B.1 above the element and its amount of neighbors. For metals, the neighbors are oxygen, for oxygen, the neighbors are metals.

Now, we can generate the unigram, bigram, trigram, etc. matrix. The connections between all the atoms of the unit cell can be conveniently visualized as a graph (Figure B.2) where the nodes are atoms and edges mean the equation $\boldsymbol{D}_{ij} < \alpha(R_i^S + R_j^S)$ for two given atoms i and j holds.

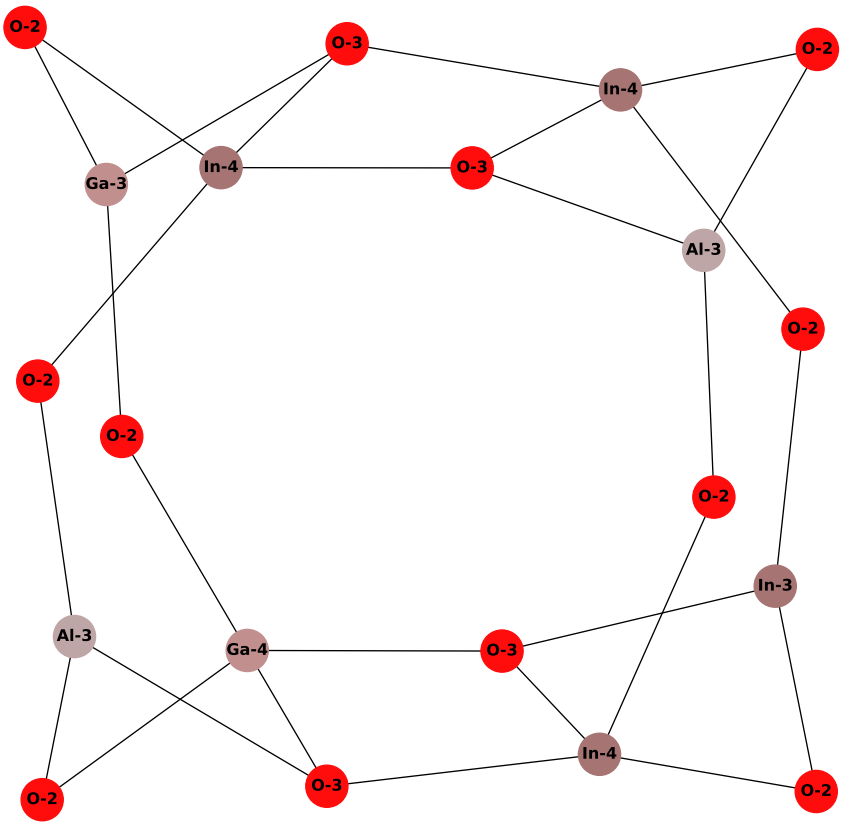


Figure B.2: Connections between coordination environments of an alloy $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$ visualized as a graph [13]

3. To conclude the example, the unigram matrix of the material is constructed. We choose the unigram matrix because it can be easily visualized for demonstration purposes. The construction of higher order matrices is similar. We count the number of atom-coordination pairs in Table B.1 and divide each count by the volume of the unit cell $V_{cell} = |\det \boldsymbol{A}| = \left|\det(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})\right| = 247.918\ \text{Å}^3$. The result can be seen in Figure B.3. The reason for choosing the amount of coordinations $c = 10$ (e. g. Al-0, Al-1,..., Al-9) is that other materials in the dataset have different nonzero atom-coordination pairs and we want

to include all the information into the matrix before we leave out any columns either for the reason of reducing the size of the matrix or simply because some columns are only zeros for all materials in the dataset.

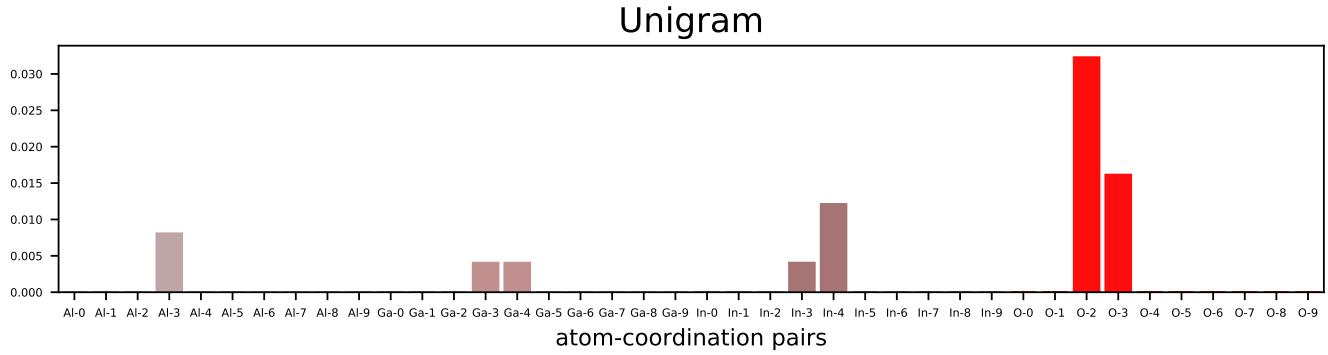


Figure B.3: Unigram of an alloy $(Al_{0.25}Ga_{0.25}In_{0.5})_2O_3$

Naturally, the bigram descriptor would be constructed by counting the number of adjacent pairs of atom-coordinations present in the graph. The trigram would be constructed by counting the number of adjacent triplets of atom-coordinations present in the graph, etc.

Now, the Σ and $\hat{\Sigma}$ descriptors will be constructed.

Appendix C

The Computation Details and Attached Storage Device Commentary

The computations were performed on the high performance computing (HPC) cluster HELIOS of Department of Mathematics, Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague. The cluster is under supervision of Ing. Pavel Strachota, Ph.D.

The nodes utilized for the CPU computations were mainly the cpu_b nodes 2x8-core Intel XEON Gold 6134@3.2GHz CPU (hyper-threading disabled), 384 GB RAM, 360 GB local SSD storage and occasionally the cpu_a nodes 2x16-core AMD EPYC 7281@2.1GHz CPU (SMT mode disabled), 128 GB RAM, 360 GB local SSD storage.

The GPU computations were performed on the gpu node containing 4x NVIDIA TESLA V100 (Volta) w. 16GB HBM2 RAM, 2x16-core Intel XEON Gold 6130@2.1GHz CPU (hyper threading disabled), 384 GB RAM, 360 GB local SSD storage.

The HELIOS cluster runs on CentOS 7.6 Linux. The PBS Pro 18.1 job scheduler is used to run the computations on the nodes of the cluster.

C.1 Classification Problem of Binary Compounds Experiment

The implementation was done in Python 3 using the sci-kit learn package [5]. The first implementation was computationally very slow. The new implementation utilizes the lasso path algorithm [38] which is also implemented in the sci-kit learn library and greatly enhances the speed of the computation.

The storage device includes the scripts used during the computation as well as the (python) pickled results.

C.2 Transparent Conducting Oxides Experiment

The implementation was done in Python 3. The ngram descriptors were implemented using the NetworkX package [13] for some parts of the code. The SOAP descriptors were generated using the QUIP package [7], concretely the quippy Python 3 interface [21]. the The models were implemented using the sci-kit learn package [5] for the kernel ridge regression models and tensorflow [1] for the feedforward neural networks.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[2] A. P. Bartok and G. Csanyi. "Gaussian approximation potentials: A brief tutorial introduction." In: *International Journal of Quantum Chemistry* 115.16 (2015), pp. 1051–1057. DOI: 10.1002/qua.24927. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/qua.24927. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/qua.24927.

[3] A. P. Bartok, R. Kondor, and G. Csanyi. "On representing chemical environments." In: *Phys. Rev. B* 87 (18 2013), p. 184115. DOI: 10.1103/PhysRevB.87.184115. URL: https://link.aps.org/doi/10.1103/PhysRevB.87.184115.

[4] H. Boche, R. Calderbank, G. Kutyniok, and J. Vybiral. *Compressed Sensing and its Applications*. Birkhäuser Basel, 2015. ISBN: 978-3-319-16042-9. DOI: 10.1007/978-3-319-16042-9.

[5] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. "API design for machine learning software: experiences from the scikit-learn project." In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013, pp. 108–122.

[6] J. Chmel. *Simulations of Quantum Plasmonic Structures*. 2019.

[7] G. Csányi, S. Winfield, J. R. Kermode, A. De Vita, A. Comisso, N. Bernstein, and M. C. Payne. "Expressive Programming for Computational Physics in Fortran 95+." In: *IoP Comput. Phys. Newsletter* (2007), Spring 2007.

[8] F. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento. "Crystal structure representations for machine learning models of formation energies." In: *International Journal of Quantum Chemistry* 115.16 (2015), pp. 1094–1101. ISSN: 0020-7608. DOI: 10.1002/qua.24917. URL: https://doi.org/10.1002/qua.24917.

[9] L. M. Ghiringhelli, J. Vybiral, E. Ahmetcik, R. Ouyang, S. V. Levchenko, C. Draxl, and M. Scheffler. "Learning physical descriptors for materials science by compressed sensing." In: *New Journal of Physics* 19.2 (Feb. 2017), p. 023017. DOI: 10.1088/1367-2630/aa57bf. URL: https://doi.org/10.1088/2F1367-2630/2Faa57bf.

[10] L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler. "Big Data of Materials Science: Critical Role of the Descriptor." In: *Phys. Rev. Lett.* 114 (10 Mar. 2015), p. 105503. DOI: 10.1103/PhysRevLett.114.105503. URL: https://link.aps.org/doi/10.1103/PhysRevLett.114.105503.

[11] L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler. "Big Data of Materials Science: Critical Role of the Descriptor (Supplemental Material)." In: *Phys. Rev. Lett.* 114 (10 Mar. 2015), p. 105503. DOI: 10.1103/PhysRevLett.114.105503. URL: http://link.aps.org/supplemental/10.1103/PhysRevLett.114.105503.

[12] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[13] A. A. Hagberg, D. A. Schult, and P. J. Swart. "Exploring Network Structure, Dynamics, and Function using NetworkX." In: *Proceedings of the 7th Python in Science Conference*. Ed. by Gael Varoquaux, Travis Vaught, and Jarrod Millman. Pasadena, CA USA, 2008, pp. 11 –15.

[14] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. von Lilienfeld, K.-R. Muller, and A. Tkatchenko. "Machine Learning Predictions of Molecular Properties: Accurate Many-Body Potentials and Nonlocality in Chemical Space." In: *The Journal of Physical Chemistry Letters* 6.12 (2015), pp. 2326–2331. DOI: 10.1021/acs.jpclett.5b00831. URL: https://doi.org/10.1021/acs.jpclett.5b00831.

[15] K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. von Lilienfeld, A. Tkatchenko, and K.-R. Muller. "Assessment and Validation of Machine Learning Methods for Predicting Molecular Atomization Energies." In: *Journal of Chemical Theory and Computation* 9.8 (2013), pp. 3404–3419. ISSN: 1549-9618. DOI: 10.1021/ct400195d. URL: https://doi.org/10.1021/ct400195d.

[16] *How to get atomic coordinates, Nomad2018 Predicting Transparent Conductors Kaggle competition*. URL: https://www.kaggle.com/tonyyy/how-to-get-atomic-coordinates.

[17] Haoyan Huo and Matthias Rupp. *Unified Representation of Molecules and Crystals for Machine Learning*. 2017. arXiv: 1704.06439 [physics.chem-ph].

[18] J. Fiala I. Kraus. *Elementární fyzika pevných látek*. České vysoké učení technické v Praze, 2016. ISBN: 978-80-01-05942-5.

[19] J. Im, S. Lee, T. Ko, H. W. Kim, Y. Hyon, and H. Chang. "Identifying Pb-free perovskites for solar cells by machine learning." In: *npj Computational Materials* 5.1 (2019), p. 37. ISSN: 2057-3960. DOI: 10.1038/s41524-019-0177-0. URL: https://doi.org/10.1038/s41524-019-0177-0.

[20] *Jmol: an open-source Java viewer for chemical structures in 3D*. URL: http://www.jmol.org/.

[21] J. R. Kermode. "f90wrap: an automated tool for constructing deep Python interfaces to modern Fortran codes." In: *J. Phys. Condens. Matter* (Mar. 2020). ISSN: 0953-8984, 1361-648X. DOI: 10.1088/1361-648X/ab82d2.

[22] Z. Li, Q. Xu, Q. Sun, Z. Hou, and W.-J. Yin. "Thermodynamic Stability Landscape of Halide Double Perovskites via High-Throughput Computing and Machine Learning." In: *Advanced Functional Materials* 29.9 (2019), p. 1807280. ISSN: 1616-301X. DOI: 10.1002/adfm.201807280. URL: https://doi.org/10.1002/adfm.201807280.

[23] S. Lu, Q. Zhou, Y. Guo, Y. Zhang, Y. Wu, and J. Wang. “Coupling a Crystal Graph Multilayer Descriptor to Active Learning for Rapid Discovery of 2D Ferromagnetic Semiconductors/Half-Metals/Metals.” In: *Advanced Materials* n/a.n/a (). 2002658, p. 2002658. ISSN: 0935-9648. DOI: 10.1002/adma.202002658. URL: https://doi.org/10.1002/adma.202002658.

[24] J. Mercer and A. S. Forsyth. “XVI. Functions of positive and negative type, and their connection the theory of integral equations.” In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 209.441-458 (1909), pp. 415–446. DOI: 10.1098/rsta.1909.0016. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.1909.0016.

[25] J. E. Moussa. “Comment on "Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning".” In: *Phys. Rev. Lett.* 109 (5 2012), p. 059801. DOI: 10.1103/PhysRevLett.109.059801. URL: https://link.aps.org/doi/10.1103/PhysRevLett.109.059801.

[26] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. “An Introduction to Kernel-Based Learning Algorithms.” In: *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council* 12 (Feb. 2001), pp. 181–201. DOI: 10.1109/72.914517.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. “Scikit-learn: Machine Learning in Python.” In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[28] M. Rupp, A. Tkatchenko, K.-S. Muller, and O. A. von Lilienfeld. “Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning.” In: *Physical Review Letters* 108.5 (2012), p. 058301. DOI: 10.1103/PhysRevLett.108.058301. URL: https://link.aps.org/doi/10.1103/PhysRevLett.108.058301.

[29] M. Rupp, A. Tkatchenko, K.-S. Muller, and O. A. von Lilienfeld. “Rupp et al. Reply:” in: *Phys. Rev. Lett.* 109 (5 2012), p. 059802. DOI: 10.1103/PhysRevLett.109.059802. URL: https://link.aps.org/doi/10.1103/PhysRevLett.109.059802.

[30] K. M. Saravanan, H. Zhang, T. Hossain, S. Reza, and Y. Wei. “Deep Learning-Based Drug Screening for COVID-19 and Case Studies.” In: (May 2021). DOI: 10.1007/7653_2020_58.

[31] K. T. Schutt, H. Glawe, F. Brockherde, A. Sanna, K. R. Muller, and E. K. U. Gross. “How to represent crystal structures for machine learning: Towards fast prediction of electronic properties.” In: *Phys. Rev. B* 89 (20 2014), p. 205118. DOI: 10.1103/PhysRevB.89.205118. URL: https://link.aps.org/doi/10.1103/PhysRevB.89.205118.

[32] M. Segall. “Can we really do computer-aided drug design?” In: *Journal of Computer-Aided Molecular Design* 26.1 (2012), pp. 121–124. ISSN: 1573-4951. DOI: 10.1007/s10822-011-9512-3. URL: https://doi.org/10.1007/s10822-011-9512-3.

[33] R. D. Shannon. “Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides.” In: *Acta Crystallographica Section A* 32.5 (1976), pp. 751–767. DOI: 10.1107/S0567739476001551. URL: https://onlinelibrary.wiley.com/doi/abs/10.1107/S0567739476001551.

[34] *Structures of RS, WZ and ZB*. URL: http://www.aflowlib.org/prototype-encyclopedia/.

[35] C. Sutton, L. M. Ghiringhelli, T. Yamamoto, Y. Lysogorskiy, L. Blumenthal, T. Hammerschmidt, J. Golebiowski, X. Liu, A. Ziletti, and M. Scheffler. *NOMAD 2018 Kaggle Competition: Solving Materials Science Challenges Through Crowd Sourcing*. 2018. arXiv: 1812.00085 [cond-mat.mtrl-sci].

[36] C. Sutton, L. M. Ghiringhelli, T. Yamamoto, Y. Lysogorskiy, L. Blumenthal, T. Hammerschmidt, J. R. Golebiowski, X. Liu, A. Ziletti, and M. Scheffler. "Crowd-sourcing materials-science challenges with the NOMAD 2018 Kaggle competition." In: *npj Computational Materials* 5.1 (2019), p. 111. ISSN: 2057-3960. DOI: 10.1038/s41524-019-0239-3. URL: https://doi.org/10.1038/s41524-019-0239-3.

[37] *The NOMAD Repository and Archive, nomad_kaggle_vegards dataset*. URL: https://nomad-lab.eu/prod/rae/gui/dataset/id/nn56nNWzSOGmuf0ptRMymg.

[38] R. J. Tibshirani and J. Taylor. "The solution path of the generalized lasso." In: *The Annals of Statistics* 39.3 (2011). DOI: 10.1214/11-aos878. URL: https://doi.org/10.1214%2F11-aos878.

[39] X. Wang, S. Ramírez-Hinestrosa, J. Dobnikar, and D. Frenkel. "The Lennard-Jones potential: when (not) to use it." In: *Phys. Chem. Chem. Phys.* 22 (19 2020), pp. 10624–10633. DOI: 10.1039/C9CP05445F. URL: http://dx.doi.org/10.1039/C9CP05445F.

[40] T. Xie and J. C. Grossman. "Crystal Graph Convolutional Neural Networks for an Accurate and Interpretable Prediction of Material Properties." In: *Physical Review Letters* 120.14 (2018). ISSN: 1079-7114. DOI: 10.1103/physrevlett.120.145301. URL: http://dx.doi.org/10.1103/PhysRevLett.120.145301.