# Machine Learning in Matter at Different Scales

Master Thesis

Roberto Díaz Pérez

Monday 19th August, 2019

Advisors: Prof. Alexander Balatsky, Dr. Matthias Geilhufe, Dr. Johan Hellsvik, Prof. Eva Lindroth

Nordic Institute for Theoretical Physics, Department of Condensed Matter, Nordita

## Abstract

The steep increase in computational power and data storage capabilities available to researchers begets the need for tools able to analyze and interpret vast amounts of data. Machine learning (ML) is such a tool, able to obtain new insights from data and provide predictive capabilities for new samples.

The Organic Materials Database (OMDB) is an open access electronic structure database for 3-dimensional crystal structures hosted at the Nordic Institute for Theoretical Physics, Nordita. The OMDB was recently extended to include magnetic structures and properties, to meet the demand for novel magnetic materials with interesting properties brought about by the construction of the European Spallation Source (ESS). However, high throughput, ab initio calculation of these properties is a computationally demanding process. Thus, in this work we propose a ML workflow capable of parametrizing Heisenberg Hamiltonians for new materials, bypassing the most demanding part of the process.

From the OMDB we construct a dataset relating crystal structures and magnetization for each site in the unit cell. We then develop novel ML models for prediction of local properties in crystal structures, and apply them to predict site magnetization for new materials. The best performing model has a root mean squared error of $0.42\mu_B$, a precise prediction for a dataset with a standard deviation of $1.51\mu_B$. These models were then used to predict magnetic properties for $200,000$ materials, an infeasible task with traditional ab initio calculations for our computational resources.

Then, we use similar techniques to study nuclei stability, training models with the data from the NUBASE2016 dataset, composed of experimental measures of nuclear properties, showcasing the universality of ML as a tool. These models are then used to predict stability of nuclei outside the experimentally reachable range, showing a region of increased stability around Z=120, agreeing with previous predictions of the island of stability.

# Contents

Chapter 1

# Introduction

## 1.1 Machine learning

Machine Learning (ML) [1], a subset of Artificial Intelligence, is a technique that allows computers to use data to solve problems without being explicitly programmed to do so. This term was coined in 1959 [2]. However, the groundwork for Artificial Neural Networks (ANN or NN), one of the most influential ML methods, was laid in 1957 by Rosenblatt [3], a psychologist, inspired by biological neurons. Since then, ANNs utility as universal approximators [4] impulsed the scientific community and industry to invest a significant effort in investigating them.

The explosion in available computational power and the improvement of convolutional NNs, for image processing, and recurrent NNs, for text processing, brought an exponential increase in ML research. We show this in Figure 1.1, where we see that while the number of publications in arXiv per year increases in a polynomial fashion, the number of those that contain *neural network* in their abstract increases exponentially.

The improvement of computational power and data storage capabilities has allowed us to obtain an ever increasing amount of data, be it from experiments or computer simulations.

ML methods have proven their ability to use this data to extract information and uncover exciting physics, or to make accurate predictions that bypass otherwise expensive calculations and reach systems that are unavailable experimentally. It is being used in physics in a number of ways, e.g. at the Large Hadron Collider (LHC) for analyzing massive amounts of highly dimensional data generated experimentally [5], or to characterize phases of matter in quantum systems [6].

Multiple initiatives have been developed to handle the huge amount of data generated nowadays, to facilitate sharing results and increase the repro-

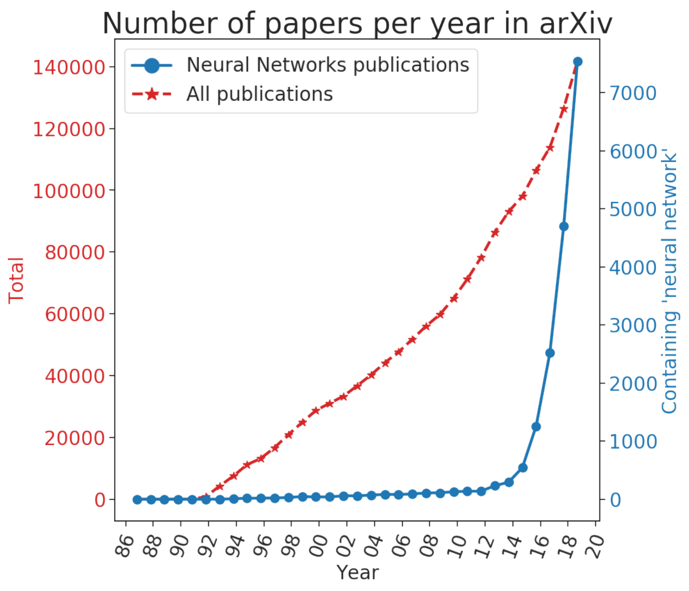


Figure 1.1: Number of papers published in arXiv.org (red, left axis) and number of papers containing the expression *neural network* in their abstract (blue, right axis).

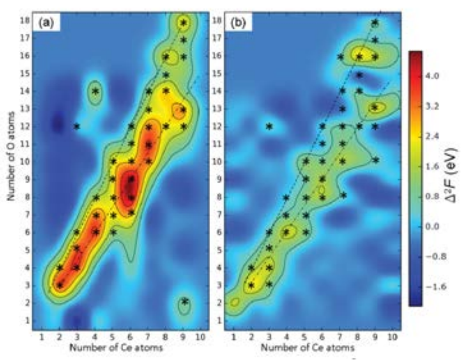
ducibility of experiments among different scientific groups. The NOMAD Repository, based in the EU, and Materials Genome Initiative, in the US, stand as such initiatives for materials science.

Specialized open databases such as Materials Project (`https://materialsproject.org/`) [7], or the Organic Materials Database (OMDB, `https://omdb.mathub.io`) [8], allow researchers to access and analyze vast amounts of data. The OMDB in particular is an electronic structure database with data that has been obtained in a consistent, ab initio, way; with multiple available advanced data mining tools, such as an online electronic structure pattern search feature [9].

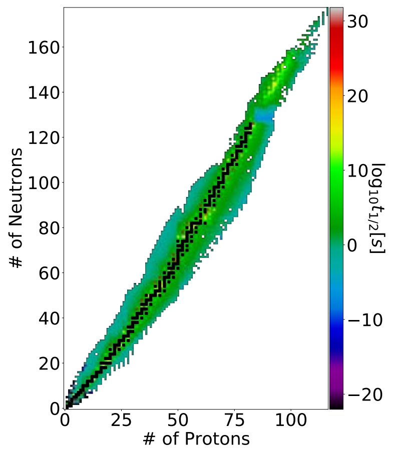
These databases make the perfect target for ML, having an extensive amount of data, with multiple possible applications, e.g. electronic band gap [10] or formation energies prediction[11].

As such, we will be using magnetization data from ab initio calculations in the OMDB to develop a ML model able to predict whether a crystal displays magnetic properties or not, and to parametrize Heisenberg's Hamiltonian for that material, allowing us to characterize its magnetic excitations, known as magnons.

Furthermore, we will apply similar techniques to nuclear data to explore nuclei stability, including the location of the fabled island of stability [12].



(a) Stability of $Ce_mO_n$ clusters determined from second formation energy differences, figure from Yu et al. [15]. Left: $\Delta^2 F_m = F(m-1,n) + F(m+1,n) - 2F(m,n)$, right: $\Delta^2 F_n = F(m,n-1) + F(m,n+1) - 2F(m,n)$



(b) Stability of Nuclei according to experimental data from NUBASE2016 [16]. Stable nuclei are represented by black squares.

Figure 1.2: Stability of crystal structures and nuclei.

## 1.2 Unifying Principles of Nature

In science, and physics in particular, a constant theme in time is that discoveries made in one field have the potential to be useful in vastly different fields, and, importantly, in applications. Crookes tubes were invented around 1875 [13], however it was not until after Röntgen's study [14] 15 years later, that their ability to visualize skeletons was discovered, bringing significant advances to medicine. The advent of machine learning follows a similar pattern, several decades after its invention it is now that we can take full advantage of its potential.

Recently, similarities were discovered between the stability of oxide clusters and nuclei stability. In Figure 1.2b we plot the order of magnitude of the half-life of a given nucleus, defined by the number of neutrons and protons. In Figure 1.2a we see a measure of stability of oxide clusters, $Ce_mO_n$, as second energies with respect to the number of cerium atoms, $m$, and oxygen atoms $n$. Both figures display a main stability diagonal; magic numbers, certain numbers of components for which the structure is substantially more stable than their neighbors; and stability islands, regions of increased stability outside the main diagonal [15, 12].

Nuclei stability is governed by high energy physics, while oxide clusters belong to condensed matter physics, a branch of physics characterized by much lower energies and much larger size. However, the results from Figure 1.2 seem to suggest a universal principle for determining which structures are stable. In this work we will predict magnetization for stable crystal structures, and then use similar techniques to study nuclei stability.

## 1.3 Dirac Materials

In 1928 Dirac explained the origin of the fine structure of atomic spectra [17], by successfully combining both the quantum and relativistic behavior of electrons, which show a linear energy-momentum relation,

$$E = \hbar v_D |\mathbf{k}|, \tag{1.1}$$

instead of the standard quadratic expression for the classical kinetic energy,

$$E = \frac{\hbar^2 \mathbf{k}^2}{2m}. \tag{1.2}$$

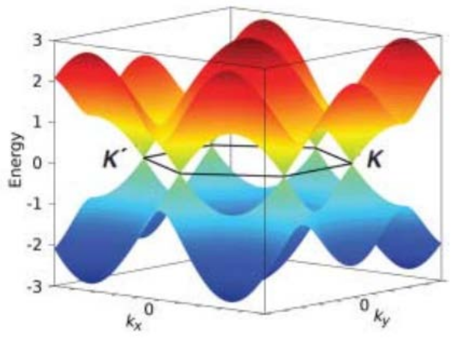
While Dirac's equation (Equation 1.1) was developed in the context of high energy physics, several condensed matter systems display low energy excitations that behave as Dirac fermions. These systems are referred to as Dirac materials [18], and include graphene, which has a linear dispersion around the Fermi level (see Figure 1.3a), topological insulators and Dirac semimetals.

Even more recently, examples of Dirac magnons were found in honeycomb structures [19], showing behavioral similarities between fermions (electrons) and bosons (magnons). Whilst in the fermionic case Dirac points are placed along the fermi level, bosons are not subject to the Pauli exclusion principle, and so, in the absence of excitations they all lie in the lowest energy state, placing Dirac points further from the ground state.

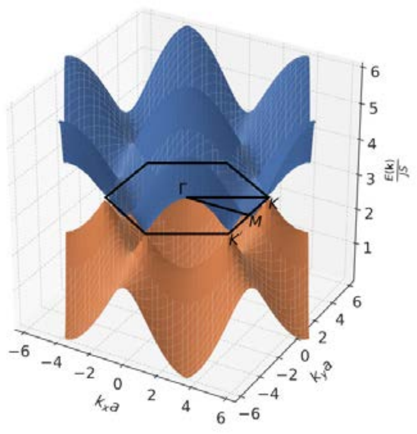
In Figure 1.3 we see the two aforementioned examples. Figure 1.3a displays the nearest neighbors tight-binding electronic band structure for graphene, with a Dirac crossing at the $K$ point at the fermi level. Figure 1.3b displays the nearest neighbors ferromagnetic magnon dispersion for a honeycomb structure, displaying a Dirac crossing at the $K$ point at an excited state.

Advanced computational features of the OMDB have been successfully used to discover new Dirac materials [20], by studying the electronic degrees of freedom in organic crystals. A similar effort has now been done to establish a magnetic database that studies bosonic excitations [21], with the objective of identifying magnetic materials with fascinating properties, such as the aforementioned Dirac magnons.

We observe, yet again, how a development in one field of physics, the Dirac equation from high energy physics, shows its utility in a disparate field,



(a) Electronic band structure for graphene with nearest neighbors tight-binding. Figure from [18].



(b) Magnon dispersion for a honeycomb structure with nearest neighbor ferromagnetic exchange.

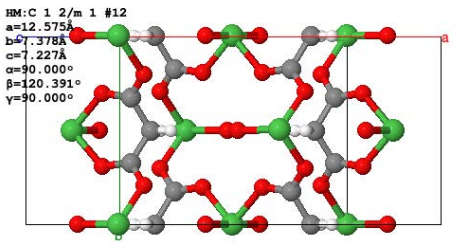
Figure 1.3: Dirac Matter.

Dirac matter in condensed matter physics. To this date, the most cited context for the Dirac equation is coming from the condensed matter community.

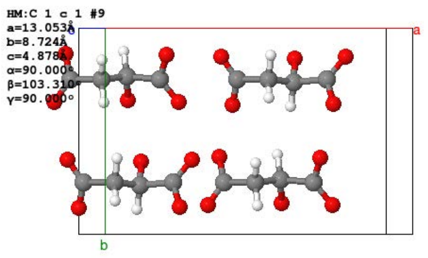
## 1.4 Magnetic features

With the European Spallation Source (ESS) construction ongoing there has been an increasing demand for frameworks capable of identifying novel functional materials with interesting magnetic properties. Organic materials are excellent candidates for this due to their following properties:

- Metal-organic frameworks (MOFs) can be constructed by linking organic molecules with metallic ions. This allows for an infinite space configuration with strong tunability.
- Due to their high complexity and bigger unit cell size than inorganic crystals they are sparsely investigated, making them an excellent research target.
- Organic materials tend to be soft and flexible, making them ideal for certain engineering purposes.



(a) Metal-organic material with formula $C_3H_6NiO_6$, available at https://omdb.mathub.io/material/id/18715. Magnetic interaction mediated by nickel (green site).



(b) Pure organic magnet with formula $C_4H_6O_5$, available at https://omdb.mathub.io/material/id/36872

Figure 1.4: Examples of magnetic organic materials present on the OMDB.

There are two classes of magnetic organic materials, those where magnetism arises by the mediation of transition or rare earth metal, known as metal-organic materials, and those in which magnetic behavior arises directly in the organic components, known as pure organic magnets [22]. In Figure 1.4 we see examples of both classes of magnets.

The Swedish QuEST for BIFROST is an international project that aims to sample metal-organic compounds computationally through the OMDB and later synthesize promising candidate materials to study their properties. It is in this context that we extended the OMDB capabilities to display the adiabatic magnon dispersion, $E(\omega)$, and dynamic structure factor, $S(q,\omega)$. This data was obtained using high throughput ab initio methods to characterize the magnetic properties of the material.

The methodology used to obtain this data is explained in great detail in our preprint [21], and an introduction to Linear Spin Wave Theory (LSWT) is presented in Section 3.2.

However, the computation of this data is extremely expensive, over the course of the last three years, we have done VASP simulations for over 26,000 materials. At an average of 60 [23] core-hours per material, it amounts to over 150 core-years to calculate all the data in the OMDB. Furthermore, obtaining the Heisenberg exchange parameters is even more computationally intensive, taking us around $4,000$ hours per material for a current total of 100 materials. On the other hand, once a ML model is trained it can compute thousands of materials per hour, taking us 48 hours to obtain the properties calculated in Section 3.3 for $200,000$ materials, amounting to around $4,000$ materials per core hour. This speed advantage with respect to traditional ab initio methods is the biggest motivation for us to investigate ML methods for magnetic properties in crystals.

The remainder of this thesis is organized as follows:

- In Chapter 2 we introduce basic ML concepts and discuss possible representations of physical systems.
- In Chapter 3 we describe a high throughput workflow for calculating magnetic excitations in crystals and the use of ML to expedite this process.
- In Chapter 4 we apply the concepts previously discussed to predict nuclear stability.

Chapter 2

# Machine Learning

Machine learning [1] (ML) is a subset of Artificial Intelligence that provides computers with the ability to learn from experience (data) to discern properties of a certain system without being explicitly programmed to do so.

ML is divided into three categories: supervised, unsupervised and reinforcement learning. Supervised learning requires and input variable ($\boldsymbol{x}$) or descriptor, and an output variable ($\boldsymbol{y}$) or target, and learns the mapping function $\boldsymbol{y} = f(\boldsymbol{x})$ using training data, e.g. translation software and image recognition. Unsupervised learning only has the input variable ($\boldsymbol{x}$) and tries to learn about the underlying properties of the data, e.g. dimensionality reduction. In reinforced learning the objective is to maximize a fitness function, or reward function, e.g. self driving cars.

From now on, unless explicitly stated otherwise we will discuss *supervised learning*.

Suppose that we have a function, $\boldsymbol{f}_{target}(\boldsymbol{x})$, that maps an input space, $\mathbb{V}_I$, to an output space, $\mathbb{V}_O$:

$$\boldsymbol{y} = \boldsymbol{f}_{target}(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{V}_I, \boldsymbol{y} \in \mathbb{V}_O, \tag{2.1}$$

from which we obtain a number of samples, $\boldsymbol{x}_i$.

We also have a function, $\boldsymbol{\Theta}_{\boldsymbol{p}}(\boldsymbol{x}), \boldsymbol{\Theta} : \mathbb{V}_I \to \mathbb{V}_O$, that depends on $\boldsymbol{x}$ and on a set of parameters, $\boldsymbol{p} \in \mathbb{V}_P$. The machine learning algorithm will try to find the best parameters, $\boldsymbol{p}$, so that:

$$\boldsymbol{y} = \boldsymbol{f}_{target}(\boldsymbol{x}) = \boldsymbol{\Theta}_{\boldsymbol{p}}(\boldsymbol{x}). \tag{2.2}$$

However, it is not usually possible to obtain the exact relation, so we need to include an error term, $\boldsymbol{e} \in \mathbb{V}_O$:

$$\boldsymbol{y} = \boldsymbol{f}_{target}(\boldsymbol{x}) = \boldsymbol{\Theta}_{\boldsymbol{p}}(\boldsymbol{x}) + \boldsymbol{e}_{\boldsymbol{p}}(\boldsymbol{x}). \tag{2.3}$$

Machine learning models learn by choosing the parameters $\boldsymbol{p}_0$ that minimize a loss function $L_{\boldsymbol{p}}(\boldsymbol{x})$, such that:

$$\sum_i L_{\boldsymbol{p}_0}(\boldsymbol{x}_i) \leq \sum_i L_{\boldsymbol{p}}(\boldsymbol{x}_i) \quad \forall \boldsymbol{p} \in \mathbb{V}_P, \tag{2.4}$$

where $i$ iterates over our data samples.

The function and parameters, $\boldsymbol{\Theta}_{\boldsymbol{p}}(\boldsymbol{x})$ and $\boldsymbol{p}$, are referred together as the model. For this model to be a valid ML model it should be able to make accurate predictions on the value of $\boldsymbol{y}$ on new, unseen inputs. To ensure this happens, our samples are divided in two sets: the training set and the test set. The test set is excluded from the training process and is used to evaluate the models.

Supervised learning is split into two further categories:

- **Classification**: where the predicted variable is one of multiple discrete classes, e.g. characterizing a market transactions as profitable or unprofitable.
- **Regression**: the predicted variable can take any value from a continuous domain, e.g. calculating the amount of profit (or loss) from a market transaction.

In this chapter we will first analyze how a ML model is capable of predicting new data in unseen samples, and some pitfalls we need to be careful about. Then we will introduce some representations ($\boldsymbol{x}$) that allow us to describe physical systems. Following by describing some algorithms ($\boldsymbol{\Theta}_{\boldsymbol{p}}$), and metrics that allow us to estimate the performance of our models.

## 2.1 Bias-variance tradeoff

When using machine learning to draw conclusions from a dataset we want to ensure those conclusions are correct and capture real trends in the data and not anomalies from our limited training samples. There is an ever-present tradeoff between models with high bias—simple models unable to uncover the real relationship within the data—and models with high variance, complex models that fail to discover actual trends due to memorization of training data. We refer to the former as underfitting and the latter as overfitting.

We will demonstrate this by considering the following polynomial:

$$f(x) = 1 + 3x^2 - x^3, \tag{2.5}$$

from which we will sample ten points and add scaled random noise. We then proceed to fit polynomials by the least squares method [24], minimizing the squared error between the fit polynomial and the sampled data. This is shown in Figure 2.1, where we observe that a second degree polynomial underfits our data and a tenth degree polynomial overfits it, memorizing the position of our samples.

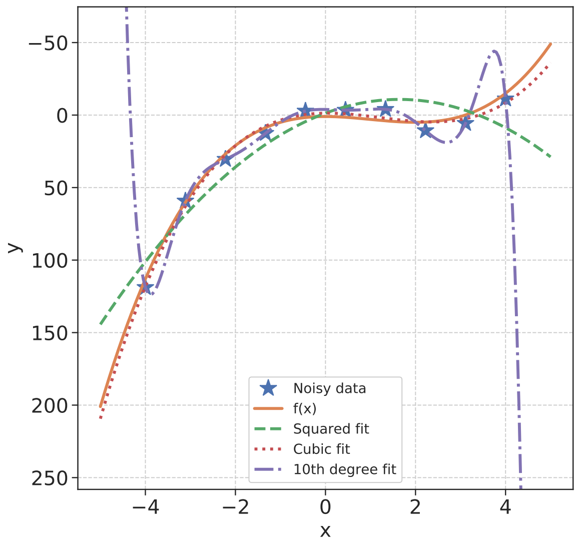


Figure 2.1: Polynomial fitting of different degrees to data sampled from a 3rd degree polynomial with random noise added.

In this case, it is obvious that the best choice is a third degree polynomial. However, in more complex datasets, with high-dimensional inputs and where the relationship is not so obvious, the choice of function to fit is not longer clear. To deal with this, we need to make a choice between either using a model with high complexity and consequently variance and try to counteract overfitting, e.g. neural networks with regularization or dropout, or using simpler models with higher bias and deal with underfitting, e.g. shallow decision trees ensembled into random forests.

Regardless of the method chosen, the dataset should always be divided into two parts: a training set in which we conduct the optimization process, and a test set which should stay untouched until after the model is trained and then used to measure the performance of the model. The performance of the model in data used for training is not indicative of how well the model would actually measure in new, unseen data, and shouldn't be interpreted as a performance metric. This is not always evident. For example, a model that uses temporal data requires a test set that is entirely in the future with respect to the training set. This is because when making real predictions in new data only data from the past will be available. Failure to correctly separate the train and test site is known as *data leakage*.

## 2.2 Representations

A representation, or descriptor, is a vector, $\boldsymbol{x}$, that contains the information necessary to represent a system.

Choosing these representations or descriptors of the system is of extreme importance for the performance of the model, and its generally beneficial to posses prior knowledge about the system and its behavior.

In this thesis, ML is applied to predict:

1. Macroscopic properties of crystals based on their structure
2. Local properties based on atomic clusters around crystal sites
3. Nuclear properties

Thus, we will introduce various representations for each of these physical systems.

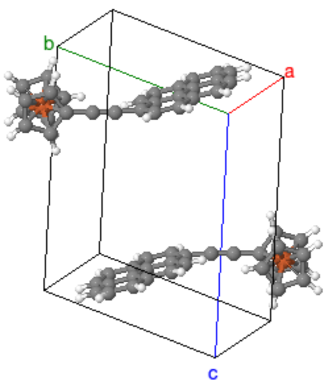
### 2.2.1 Representations of Crystals

Choosing the appropriate representation for a crystal is a hotly debated theme in the Materials Informatics community [25, 26, 11, 27, 28].

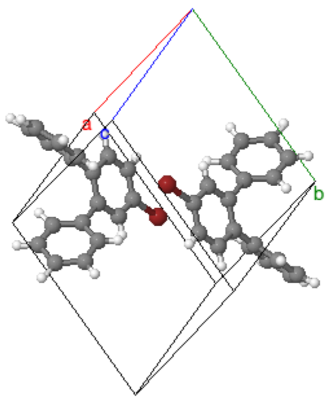
A crystal is identified by the lattice vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$, the position of its atoms $(\mathbf{r}_i)$ in the unit cell, and their atomic numbers, $(Z_i)$. However, there is an infinite number of $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \{\mathbf{r}_i, Z_i\}$ that describe any such system—due to the invariance with respect to choosing the unit cell—and ML algorithms would not be able to identify two crystals as the same if they have been represented differently.

A good descriptor for a crystal should be:

1. Translationally invariant
2. Rotationally invariant
3. Invariant with respect to the size of the cell
4. Invariant with respect to permutation of equivalent atoms in different cells



(a) Material 510, with formula $C_{28}H_{18}Fe$, available at https://omdb.mathub.io/material/id/510.



(b) Material 1583, with formula $C_{18}H_{13}Br$, available at https://omdb.mathub.io/material/id/1583.

Figure 2.2: OMDB materials represented in table 2.1

#### 2.2.1.1 Multi-Hot Encoding

We create a simple descriptor that only takes into account the species of the atoms in the crystal. The crystals are represented by a vector, $\boldsymbol{x}_i$, with length equal to the number of different chemical species in the entire dataset. The presence or absence of a species gets represented by a 1 or a 0 respectively.

We illustrate this representation with two materials from the OMDB (Figure 2.2), as seen in table 2.1 where the first material has iron but not bromine, but the second has bromine but not iron, determining the position of *hot* spots, ones instead of zeros, in the vector.

| **Formula** | C | H | Fe | Br | ... | $\boldsymbol{x}$ |
|---|---|---|---|---|---|---|
| $C_{28}H_{18}Fe$ | 1 | 1 | 1 | 0 | ... | [ 1 1 1 0 ...] |
| $C_{18}H_{13}Br$ | 1 | 1 | 0 | 1 | ... | [ 1 1 0 1 ...] |

Table 2.1: Schematic representation of Multi-Hot encoding.

This representation takes inspiration from One-Hot encoding, a widely used Ml representation for encoding binary features.

#### 2.2.1.2 Sine Matrix

In order to understand the Sine Matrix (SM) [11], we need to first introduce the Coulomb Matrix (CM) as the SM is a crystal structure representation that extends the Coulomb Matrix, from molecules to crystal structures.

The Coulomb Matrix (CM) [25] was originally developed to provide a ML representation of a molecule capable of predicting atomization energies, and has been shown capable of predicting other properties [29]. A molecule can be represented by a matrix $\mathbf{M}$, with components:

$$M_{ij} = \begin{cases} 0.5 Z_i^{2.4}, & i = j \\ \frac{Z_i Z_j}{|\mathbf{r}_i - \mathbf{r}_i|}, & i \neq j, \end{cases} \tag{2.6}$$

where the diagonal elements come from a polynomial fit of atomic energies to atomic number, and the off-diagonal elements represent Coulomb repulsion between the atoms in the molecule. $Z_i, \mathbf{r}_i$ represent the atomic number and the position of the atom $i$, respectively. This matrix is then flattened to create the input vector, $\boldsymbol{x}$.

The size of the CM still depends on the number of atoms in the molecule. In order to keep the representation's size constant we will pad the vector with zeros up to a fixed size. Furthermore, multiple CMs can be produced from the same molecule depending on the ordering of the atoms, we will address this ambiguity by sorting columns and rows according to their euclidean norm.

We can try, naïvely, to extend the Coulomb Matrix to support an infinitely repeating crystal structure as:

$$M_{ij} = \begin{cases} 0.5 Z_i^{2.4}, & i = j \\ \frac{Z_i Z_j}{N} \sum_{m,n} \frac{1}{|\mathbf{r}_{m,i} - \mathbf{r}_{n,j}|}, & i \neq j, \end{cases} \tag{2.7}$$

where $m$ and $n$ iterate over all $N$ closest unit cells. However, as $N \to \infty$, convergence issues arise.

One solution is to use any arbitrary simpler potential, $\Phi(\mathbf{r}_i, \mathbf{r}_j)$, than the electrostatic, that shares basic properties with the sum previously stated and doesn't have convergence issues:

$$M_{ij} = \begin{cases} 0.5 Z_i^{2.4}, & i = j \\ Z_i Z_j \Phi(\mathbf{r}_i, \mathbf{r}_j), & i \neq j. \end{cases} \tag{2.8}$$

Faber et al. [11] propose the interaction that defines the SM:

$$\Phi(\mathbf{r}_i, \mathbf{r}_j) = \left| \mathbf{B} \cdot \sum_{k=\{x,y,z\}} \boldsymbol{e}_k \sin^2 \left[ \pi \boldsymbol{e}_k \mathbf{B}^{-1} \cdot (\mathbf{r}_i - \mathbf{r}_j) \right] \right|^{-1}, \tag{2.9}$$

where $\boldsymbol{e}_k$ refers to the $k$ coordinate unit vectors, and $\mathbf{B}$ is the matrix formed by the unit vectors. This interaction is displayed in Figure 2.3.

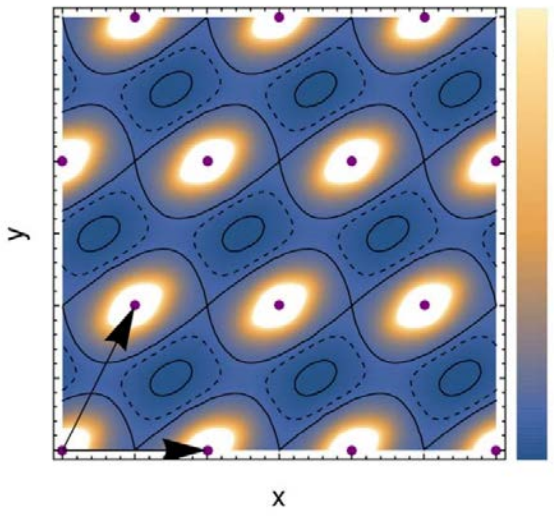


Figure 2.3: $\Phi(\boldsymbol{r}_i, \boldsymbol{r}_j)$ potential in a 2D crystal. The interaction strength between an atom situated at the origin and another one at $(x, y)$ is represented with colors. Figure from [11].

This interaction has the following properties:

1. The expression is periodic with respect to the crystal lattice.
2. The contribution from any pair of equivalent atoms is equivalent, that is, invariance with respect to the selection of unit cell.
3. As in the Coulomb potential, $\Phi(\mathbf{r}_i, \mathbf{r}_j) \to \infty$ as $|\mathbf{r}_i - \mathbf{r}_j| \to 0$.

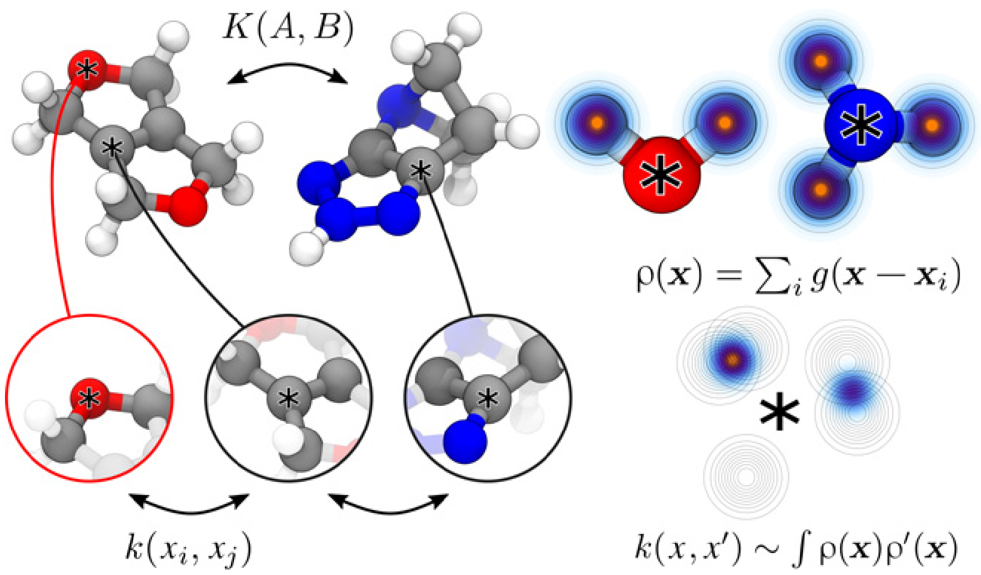
Which make the SM an appropriate crystal representation. In this thesis we used the implementation from DScribe [30].

#### 2.2.1.3 Smooth Overlap of Atomic Positions (SOAP)

The *SOAP average kernel* [27, 28], is defined as measure of similarity between two structures, constructed by the similarity between the environments of each atom in the structures. For an schematic representation see Figure 2.4.

The *local density*, $\rho_\chi(\mathbf{r})$, of an atom is constructed by placing a Gaussian on top of every atom within an environment, $\chi$, containing all atoms within a certain radius, $r_c$, of the central atom:

$$\rho_\chi(\mathbf{r}) = \sum_{i \in \chi} \exp\left(-\frac{(\mathbf{x}_i - \mathbf{r})^2}{2\sigma^2}\right), \tag{2.10}$$



$K(A,B)$
$\rho(\mathbf{x}) = \sum_i g(\mathbf{x}-\mathbf{x}_i)$
$k(x_i, x_j)$
$k(x,x') \sim \int \rho(\mathbf{x})\rho'(\mathbf{x})$

Figure 2.4: Figure from [28]. The SOAP kernel between two structures $K(A,B)$ is determined by comparing their environments $\rho(x)$. The function $g(\boldsymbol{x}-\boldsymbol{x_i})$ is the gaussian defined in 2.10.

where the standard deviation, $\sigma$, is a hyperparameter, set before the learning process begins, usually set to 1 Å, the size of an hydrogen atom.

The *SOAP kernel* is then defined as the overlap of the environments, integrating all the possible 3D rotations, denoted by $\hat{R}$:

$$k(\chi_i, \chi_j) = \int d\hat{R} \left| \int \rho_{\chi_i}(\mathbf{r})\rho_{\chi_j}(\hat{R}\mathbf{r})d\mathbf{r} \right|^2 . \tag{2.11}$$

Information about chemical species is introduced by modifying Equation 2.11 so that the species are matched separately:

$$k(\chi_i, \chi_j) = \sum_{\alpha,\beta} \int d\hat{R} \left| \int d\mathbf{r}\rho^{\alpha}_{\chi_i}(\mathbf{r})\rho^{\beta}_{\chi_j}(\hat{R}\mathbf{r}) \right|^2 , \tag{2.12}$$

where all pairwise combinations of species, $\alpha, \beta$ are considered.

Having two structures, $A$ and $B$, we now calculate the *pair wise similarity matrix* between them:

$$C_{ij}(A,B) = k(\chi_i^A, \chi_j^B), \tag{2.13}$$

with a dimensionality, $N_A \times N_B$, depending on the number of atoms in each structure.

The simplest way to convert this matrix into a single number, as a measure of similarity between the structures, is to calculate the *SOAP average kernel*:

$$K(A,B) = \frac{1}{N_a N_b} \sum_{ij} C_{ij}(A,B), \tag{2.14}$$

which can be used as input for kernel based methods, such as Kernel Ridge Regression or Support Vector Machines.

In this thesis we use the implementation from DScribe [30] and SOAP lite [31].

### 2.2.2 Local environment representations

In order to calculate local properties in a crystal, we need all the information needed to identify it: the lattice vectors ($\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$), the position of its atoms ($\mathbf{r}_i$) and the atomic numbers, ($Z_i$). Furthermore, to identify the site, we also require the index of the site we are analyzing, from now on referred as the central site $k$.

We will define the environment of a site by its nearest neighbors, determined using covalent radii information and a neighbor search algorithm implemented in the atomic simulation environment (ASE) [32]. The central site and its nearest atoms will be treated as if they were an independent molecule for representation purposes. In Figure 2.5 we draw this environment from a 2D crystal.

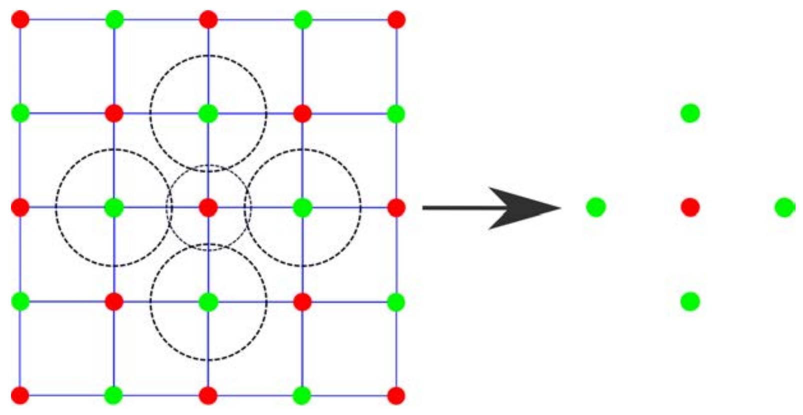


Figure 2.5: From a 2D square lattice with a chequerboard pattern we extract the environment from a central site (red) and a given covalent radius (black circles) for each site.

We need representations that focus on the central site and its environment, while respecting the criteria given in Section 2.2.1 for good crystal representations.

Kernel representations, such as SOAP, are not fit for this task, as the kernel matrix scales quadratically with the number of training samples. Due to each crystal in the OMDB having around 80 sites, the matrix is impossible to compute with the computational resources at our disposal.

The data on each site is highly correlated with the other sites on the same crystal, this means that the split between test and training dataset should be done at the crystal level, not the site level, to avoid data leakage (see Section 2.1).

#### 2.2.2.1 Coulomb Matrix

We will apply the Coulomb Matrix (CM), as described in 2.2.1.2, to the site environment as if it were an independent molecule.

#### 2.2.2.2 Coulomb Vector

We propose the Coulomb Vector, $\boldsymbol{C}$, as a modification of the CM where only the row corresponding to the central site, $k$, is considered, disregarding interactions between non central sites,

$$C_i = M_{ik}. \tag{2.15}$$

The ordering of the vector is also changed, the first element is the diagonal term in the CM, $M_{kk}$, as it is the central site's self interaction term, and the remaining elements are sorted by magnitude.

The purpose of this representation is to focus on the information relevant to the central site, compared to the full Coulomb Matrix.

#### 2.2.2.3 Radial Multi-Hot

The CM representation fails to capture the similarities between atoms of distant atomic numbers, neglecting the influence of periodic table groups on chemical properties, e.g. if in an organic molecule we change a Carbon atom (Z=6) for Nitrogen (Z=7) the CM will not vary significantly, however, by changing it to Silicon (Z=14) the CM will vary greatly, even though both C and Si belong to group 16 in the periodic table and have the same outer electronic layer, resulting in similar oxidation states and similar covalent binding properties.

To introduce the concept of similarities of different chemical species, in this work we propose encoding the species of the central atom in a One-Hot representation, using a vector of length equal to the number of different species in our dataset, with only one component different to 0. In table 2.2 we see an example of a dataset containing only C, H and O. We can define this vector it in a compact way:

$$V_i^{\text{One-Hot}} = \delta_{ik}, \tag{2.16}$$

| **Central site** | C | H | O | $\boldsymbol{V}_{\text{One-Hot}}$ |
|---|---|---|---|---|
| C | 1 | 0 | 0 | [ 1 0 0 ] |
| H | 0 | 1 | 0 | [ 0 1 0 ] |
| O | 0 | 0 | 1 | [ 0 0 1 ] |

Table 2.2: One-Hot encoding.

where $k$ refers to the species of the central atom, and $i$ is an index that unequivocally identifies each specie.

Then, the distance of the atoms in the molecule is encoded in a similar fashion, with the inverse of the distance instead of a binary representation:

$$V_{i,l}^{\text{distance}} = \frac{1}{|\boldsymbol{r}_l - \boldsymbol{r}_{\text{central}}|}\delta_{is_l}, \tag{2.17}$$

where $l$ iterates over all other atoms in the environment, and $s_l$ is the species of said atom.

The dimension of the vector, still depend on the number of sites in the environment around the central atom. To ensure a constant representation size we propose as the simplest solution to sum over the environment:

$$V_i^{\text{distance}} = \sum_l \frac{1}{|\boldsymbol{r}_l - \boldsymbol{r}_{\text{central}}|}\delta_{is_l}. \tag{2.18}$$

This vector is then concatenated with $\boldsymbol{V}_{\text{One-Hot}}$ resulting in of length equal to twice the size of the total number of different chemical species in our dataset:

$$\boldsymbol{V} = [\boldsymbol{V}_{\text{One-Hot}}\boldsymbol{V}_{\text{distance}}]. \tag{2.19}$$

This model has the same information as in the previous Coulomb Vector. However, by introducing a vectorial representation of the species instead of using the atomic numbers, the model should be able to identify element similarities, that is, discovering the periodic table that better describes our property.

We will give an example for the carbon sites of two simple molecules: methane $CH_4$ and carbon dioxide $CO_2$. We will index the One-Hot vectors in the same order as in table 2.2: C, H, O. Then, for both molecules:

$$\boldsymbol{V}_{\mathrm{C}}^{\text{One-Hot}} = [1,\ 0,\ 0]. \tag{2.20}$$

In methane, the carbon-hydrogen bond length is of 1.09Å [33] and for carbon dioxide the carbon-oxygen bond length is of 1.16Å. Then the distance vectors are:

$$\boldsymbol{V}_{\mathrm{CH_4,C}}^{\text{distance}} = [0,\ 3.67,\ 0], \tag{2.21}$$

$$\boldsymbol{V}_{\mathrm{CO_2,C}}^{\text{distance}} = [0,\ 0,\ 1.72], \tag{2.22}$$

Making the input vectors:

$$\boldsymbol{V}_{\mathrm{CH_4,C}} = [1,\ 0,\ 0,\ 0,\ 3.67,\ 0], \tag{2.23}$$

$$\boldsymbol{V}_{\mathrm{CO_2,C}} = [1,\ 0,\ 0,\ 0,\ 0,\ 1.72]. \tag{2.24}$$

### 2.2.3 Nuclear Representations

In this work we also will study the stability of a nucleus with a given number of neutrons and protons.

Recall that a chemical element, or species, is defined by the number of protons (Z), and an isotope is determined by the number of neutrons (N) in the nucleus.

Our objective is to predict the stability of nuclei that are impossible to synthesize with the current technology, so the only information available a priori is the numbers Z and N. We will extend this representation with extra information from nuclear models.

#### 2.2.3.1 Magic Number Representation

The simplest possible representation is a vector comprised of the number of protons and neutrons:

$$(Z, N). \tag{2.25}$$

However, as shown by Costiris et al. [34] and Niu et al. [35] in their studies on $\beta$-decay, an even or odd numbers of protons and neutrons has a significant effect in the stability of the nucleus, e.g. there are 147 stable nuclei with even number of both neutrons and protons, 101 with an even odd combination and 5 with both an odd number of protons and neutrons. Adding an extra discrete parameter, $\delta$, that encodes this parity will result in improved models:

$$\delta = \begin{cases} +1, & \text{for even-even nucleus} \\ \ \ 0, & \text{for even-odd nucleus} \\ -1, & \text{for odd-odd nucleus} \end{cases} \tag{2.26}$$

Furthermore, we will also add the distance—$\delta Z, \delta N$— to the closest magic numbers, regions of increased stability [12], resulting in a vector composed of 5 components:

$$(Z, N, \delta, \delta Z, \delta N) \tag{2.27}$$

#### 2.2.3.2 Yukawa representation

In 1935 Yukawa [36] proposed that the strong force was mediated by an exchange particle and had the following form:

$$V_{\text{Yukawa}} = -g^2 \frac{e^{-\alpha m r}}{r}, \tag{2.28}$$

Where $g$ is a scaling constant, $m$ is the mass of the particle that mediates the interaction and $\alpha$ determines the range of the interaction. In the case of a massless interaction particle, e.g. photons, we obtain a Coulomb like potential.

Nowadays, quantum chromodynamics (QCD) is the theory that explains the strong force, however, models were built upon Yukawa's potential that successfully reproduced experimental results. One such model is the Reid potential [37], which, in its simplest form, reads as a central potential:

$$V(x) = -h\frac{e^{-x}}{x} - 1650.6\frac{e^{-4x}}{x} + 6484.2\frac{e^{-7x}}{x}, \tag{2.29}$$

with $h = 10.463\,\text{MeV}$ and $x = \mu r, \quad \mu = 0.7F^{-1}$.

This model was obtained by parametrizing a Yukawa interaction with soft core repulsion and fitting to experimental results. We will be using in to obtain representations similar to those used for crystals and molecules.

By setting up a system of neutrons and protons that interact with this potential, and in the proton-proton case, adding Coulomb repulsion, we can proceed with a global minimization process to obtain the most stable structure.

The minimization is done using basin hopping [38], a method based in Monte Carlo minimization, that deforms the potential energy surface to the local minima closest to each point as shown in Figure 2.6.

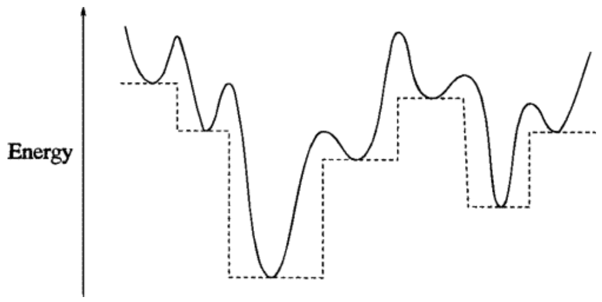


Figure 2.6: Potential energy surface deformation in basin hopping. The Monte Carlo simulation explores the local minima, as if it were hopping the basins, where the method takes it names from. Figure from [38].

We then treat the resulting structures as molecules for representation purposes.

## 2.3 Algorithms

### 2.3.1 Artificial Neural Networks

Artificial Neural Networks (ANNs or NNs) are one the of most versatile machine learning methods. They were designed by Rosenblatt [3], a psychologist, loosely inspired by the neurons in our brains.

The base element of a NN is a neuron, an object that transforms an input, $\boldsymbol{x} \in \mathbb{R}^m$, into an output $h \in \mathbb{R}$, and it is composed of the following parts:

- A set of weights, $\boldsymbol{w} \in \mathbb{R}^m$;
- A bias, $b \in \mathbb{R}$;
- An activation function, $f : \mathbb{R} \to \mathbb{R}$.

The neuron weights the output vector and adds a bias to it:

$$Y = \boldsymbol{w} \cdot \boldsymbol{x} + b, \tag{2.30}$$

and then applies the activation function to calculate the output:

$$h = f(Y). \tag{2.31}$$

Multiple activation functions can be used: the step function, the hyperbolic tangent, softmax, rectified linear units (ReLu), are just some examples. The main reason to use activation functions is that they introduce non-linearities in our model that help to learn arbitrarily complex mappings.

Neurons are connected to each other akin to the neural connections present in brains. The connections can be arbitrary, but the simplest example is a feedforward neural network, where all the neurons on one layer connect to the next one. Such network is sketched in Figure 2.7.

During the training process, the weights and biases are optimized to find the ones that minimize a defined loss function for the train set, e.g. mean squared error. Usually this minimization is done by a stochastic gradient descent process, such as ADAM [39]

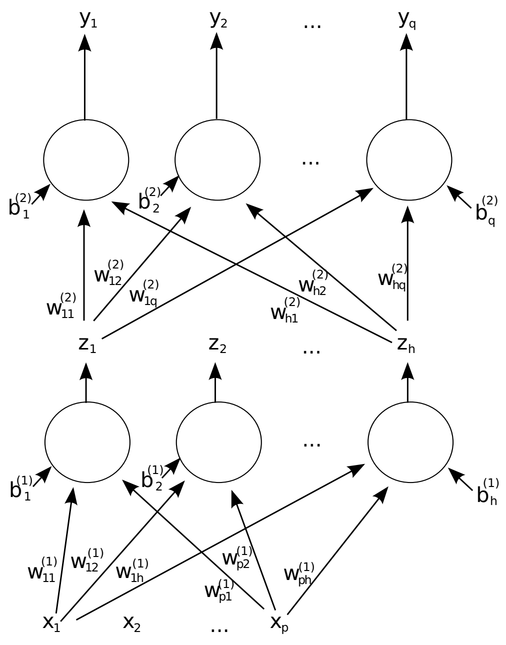


Figure 2.7: Two layer feedforward neural network with multiple inputs and outputs. Figure by Mcstrother [40].

In this thesis we use the NN implementation from scikit-learn [41].

### 2.3.2 Random Forests

A Random Forest (RF) is a ML algorithm that ensembles multiple simple models called decision trees [42]. A decision tree divides the data at each branch bifurcation according to features of the representation chosen for the data itself.

In Figure 2.8 we display a simple decision tree that performs binary classification (magnetic or non-magnetic) of materials. Assuming that the representation contains explicit information on the chemical formula of the material, there are three possible paths, one for a material with iron, another one for a material without iron but with chromium, and the last one for material without either of them, this one being the only path that results in a not magnetic prediction. This is an example of a really shallow decision tree that suffers from high bias and low variation (underfitting).

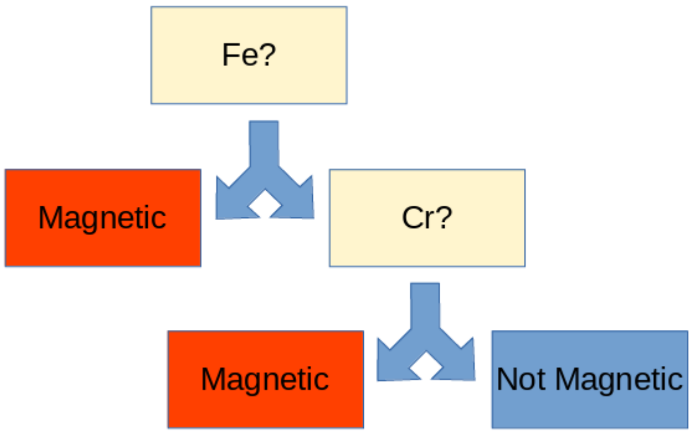


Figure 2.8: Simple decision tree that asks yes (left) or no (right) questions to classify a material as magnetic or not according to the presence of Fe or Cr in it.

Deep decision trees suffer from the opposite problem, their complexity brings high variance and low bias. A popular method consists of using a high number of shallow decision trees trained differently, e.g. using different subsets of the training data (bagging). This method is called Random Forest (RF), as it consists of multiple trees.

We use the RF implementation of scikit-learn [41].

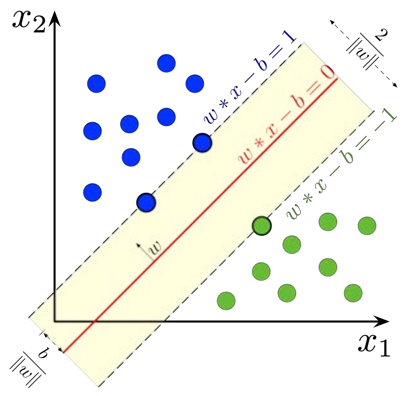


Figure 2.9: Hyperplane separating samples in two classes, support vectors lie on the margin. Figure by Larhmam [44].

### 2.3.3 Support Vector Machines

Support-vector Machines (SVM) [43] are a machine learning method used for binary classification. It separates the data into two classes, $y = \pm 1$, by separating the input space into two regions with an hyperplane, which can be written as:

$$\boldsymbol{w} \cdot \boldsymbol{x} - b = 0. \tag{2.32}$$

We want to obtain the maximum margin hyperplane, that is, the one that correctly separates our data and maximizes the distance to the closest points, which are called support vectors, giving the method its name. This plane is shown in Figure 2.9.

We force all positive samples to lie in one side of the hyperplane and all negatives in the other side,

$$\begin{aligned} \boldsymbol{w} \cdot \boldsymbol{x}_i - b &\geq 1, \quad \text{if } y_i = 1; \\ \boldsymbol{w} \cdot \boldsymbol{x}_i - b &\leq -1, \quad \text{if } y_i = -1, \end{aligned} \tag{2.33}$$

which can be simplified to:

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i - b) \geq 1. \tag{2.34}$$

Maximizing the distance between the hyperplane and the support vectors corresponds to maximizing $2/||\boldsymbol{w}||$, however, this is a non convex function, so we will instead minimize the squared inverse, converting our problem to:

$$\begin{aligned} &\min(||\boldsymbol{w}||^2/2), \\ &\text{subject to } y_i^{sv}(\boldsymbol{w} \cdot \boldsymbol{x}_i^{sv} - b) = 1, \end{aligned} \tag{2.35}$$

where the superscript *sv* refers to the support vectors. Note that only the support vectors affect the determination of the hyperplane. This is a constrained minimization problem, which can be solved using Lagrange multipliers:

$$L = \frac{||\boldsymbol{w}||^2}{2} - \alpha_i(y_i^{sv}(\boldsymbol{w} \cdot \boldsymbol{x}_i^{sv} - b) - 1). \tag{2.36}$$

However, the data is not usually perfectly separable, so we will introduce the *hinge loss*:

$$\mathcal{L}_i = \max(0, 1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i - b)). \tag{2.37}$$

Often, it is not the hinge loss that is used (L1-SVM), but its square (L2-SVM), as L2 is diferentiable [45] but L1 isn't. Then the problem reduces to minimizing the loss while maximizing the margin:

$$\min\left(C\left(\frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i^2\right) + \frac{||\boldsymbol{w}||^2}{2}\right), \tag{2.38}$$

where $C$ is a regularization parameter that determines the tradeoff between correctly classifying the samples, and increasing the margin size of the hyperplane. A small $C$ value will be too general and fail to adapt to the data (underfitting) but a large $C$ value will memorize our training data (overfitting), as explained in Section 2.1.

In order to deal with imbalanced data, we will use a different weight for $C$ for each class, inversely proportional to the frequency of appearance of each class in the training data:

$$C_{y_i} = C * \frac{n}{n_{y_i}}, \tag{2.39}$$

where $n$ is the number of training samples and $n_{y_i}$ the number of those with class $y_i$.

Thus, the optimization problem reduces to:

$$\min\left(\left(\frac{1}{N}\sum_{i=1}^{N}C_{y_i}\mathcal{L}_i^2\right) + ||\boldsymbol{w}||^2\right), \tag{2.40}$$

and this minimization can be done numerically. We will be using sklearn's [41] implementation, which in turn uses LIBLINEAR's [11].

## 2.4 Metrics

In this section we will define commonly used metrics [24, 46], indicators of performance, for evaluating ML models, both for classification and regression.

### 2.4.1 Classification

The most used metric in classification is *accuracy*. It is defined as the ratio of correct predictions over total predictions:

$$\text{Accuracy} := \frac{\text{Correct predictions}}{\text{Total predictions}}. \tag{2.41}$$

From this point on, we will suppose that we are in a binary classification problem, that is, only two classes are present, denoted as positive and negative. If one of the classes is present much more frequently than the other one, that is, we have an overrepresented class, we have an *unbalanced* dataset. In this case, accuracy is not the best metric, because a model that always predicts the overrepresented class it will have good accuracy, even if its predictive capabilities are null, e.g. if 99% of the data is negative, a model that always predicts negative will achieve 0.99 accuracy.

In order to define new metrics, better suited for unbalanced problems, let us introduce the *Confusion Matrix*, which, as its name indicates, informs us of where our model is *confused*—where it makes wrong predictions. Then, by looking at the predicted class and the real one, we can construct the matrix displayed in table 2.3.

| | **Predicted: Positive** | **Predicted: Negative** |
|---|---|---|
| **Actual: Positive** | True Positive (TP) | False Negative (FN) |
| **Actual: Negative** | False Positive (FP) | True Negative (TN) |

Table 2.3: Confusion Matrix in a binary classification problem.

With these concepts, we can define:

- Accuracy, equivalent to Equation 2.41:

$$\text{accuracy} := \frac{TP + TN}{TP + FP + TN + FN}. \tag{2.42}$$

The fraction of correct predictions over total predictions made.

- Precision:

$$\text{precision} := \frac{TP}{TP + FP}. \tag{2.43}$$

The fraction of correct positive predictions.

- Recall:

$$\text{recall} := \frac{TP}{TP + FN}. \tag{2.44}$$

The detection ratio for positive samples.

- F1:

$$\text{F1} := \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}. \tag{2.45}$$

The harmonic mean of recall and precision.

When classifying an unbalanced dataset, it is important to analyze both accuracy and F1 in order to evaluate a model. Furthermore, looking at the Confusion Matrix allows us to know where the errors are.

How good a model is depends on its purpose, e.g. if the model is used for cancer detection, missing a patient with cancer can have drastic consequences, so a model that minimizes the number of FNs, or maximizes recall, is desired.

### 2.4.2 Regression

The most straightforward metric for regression is Mean Absolute Error (MAE), defined as the arithmetic mean of the absolute value of the errors in the predictions,

$$\text{MAE} = \sum_{i=1}^{n} \frac{\left| y_i^{\text{pred}} - y_i^{\text{target}} \right|}{n}, \tag{2.46}$$

where $y^{\text{pred}}$ refers to the values predicted by our model and $y^{\text{target}}$ to the target values our model tries to reproduce.

However, the metric most often optimized is Mean Squared Error (MSE),

$$\mathrm{MSE} = \sum_{i=1}^{n} \frac{\left(y_i^{\mathrm{pred}} - y_i^{\mathrm{target}}\right)^2}{n}, \tag{2.47}$$

as it punishes large mistakes more which provides better results for outliers. The squared root of the MSE is defined as Root Mean Squared Error (RMSE) has the same units as the target variable, therefore being easier to interpret.

However, these values are meaningless without prior knowledge of a dataset. In order to know how good our model actually is we need to compare such metrics with values such as the mean or standard deviation of the target values. Consequently, we will use a metric that can be analyzed on its own.

The variance (VAR) of the target values corresponds to the MSE of a model that always predicts the mean value, independently of the input, $y_i^{\mathrm{pred}} = \bar{y}$:

$$\mathrm{VAR} = \sum_{i=1}^{n} \frac{\left(\bar{y} - y_i^{\mathrm{target}}\right)^2}{n}. \tag{2.48}$$

By comparing the variance with the mean squared error, we define r-squared, $r^2$, as:

$$r^2 = 1 - \frac{\mathrm{MSE}}{\mathrm{VAR}} \tag{2.49}$$

This metric is restricted to the interval $(-\infty, 1]$, where a negative value corresponds a model that performs worse than simply predicting the mean, and a value of 1 corresponds to a perfect model. By comparing our model to a baseline, mean prediction, we can give an easy to interpret performance metric that does not require extra knowledge about the dataset.

Chapter 3

# Magnetism

## 3.1 Introduction

We recently extended the OMDB to display magnon excitations [21], and this feature is available publicly at https://omdb.mathub.io/material?type=magnon.

The process used for high throughput calculations of magnetic excitations [21] is extremely expensive, so we aim to bypass the costliest parts of the process, the computation of the parameters of the Heisenberg Hamiltonian: the exchange interaction parameters $J_{ij}$ and the magnetization per site $\boldsymbol{m}_i$.

In this chapter we will first introduce the theoretical background needed to understand magnetic excitations and later apply the ML concepts discussed in Chapter 2 to facilitate more efficient high throughput calculations.

## 3.2 Linear Spin Wave Theory

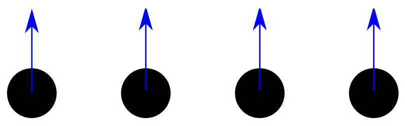
A particle that has a spin $\boldsymbol{S}$ has an associated magnetic moment $\boldsymbol{B}$,

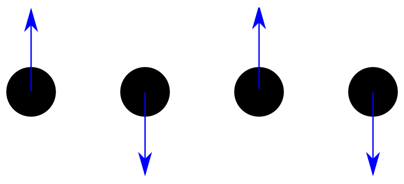
$$\boldsymbol{B} = g\frac{q}{2m}\boldsymbol{S}, \tag{3.1}$$

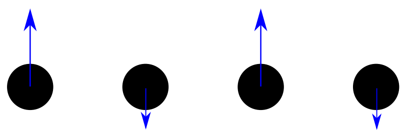
where $g$ is a dimensionless factor, and for an electron $g \approx 2$; $q$ denotes the charge and $m$ the mass. The orbital motion of the electron also contributes to the magnetic moment, but we will work in the limit of weak spin-orbit interaction so we will ignore it.

The spin configuration that minimizes energy is called the ground state, and is denoted as the vacuum state. In Figure 3.1 we see some of the most significant ground states, here listed:

- **Ferromagnetism**: All spins have the same direction and magnitude. Named after iron, it is also present in cobalt, nickel and many more materials.
- **Antiferromagnetic**: All spins have alternating directions, resulting on no net magnetization. Chromium is a well known example.
- **Ferrimagnetic**: All spins are collinear but different magnitudes, resulting in net magnetization—e.g. magnetite, $Fe_3O_4$.
- **Spin Spiral**: The direction of the spins is rotated a constant amount per unit cell as we move in one direction, e.g. $\gamma$-iron [47].


(a) Ferromagnetic


(b) Antiferromagnetic


(c) Ferrimagnetic

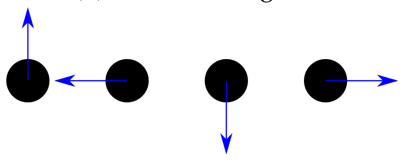

(d) Magnetic spiral

Figure 3.1: Schematic examples of magnetic groundstates on a 1D chain.

Spin waves transmit disturbances on the magnetic ground state of a material with infinite periodicity. This excitation can be quantized. Which, from a second quantization point of view, corresponds to quasiparticles, named magnons. Magnons are spin-1 so they display bosonic behavior.

Linear Spin Wave Theory (LSWT) studies low energy magnetic excitations in crystal structures. First introduced by Bloch [48] and independently by Slater [49], it provides a framework that allows one to obtain the magnetic dispersion relation, or energy momentum relation.

### 3.2.1 Ferromagnetic case

In this section we will assume that the groundstate of our crystal has Ferromagnetic ordering (see Figure 3.1a), meaning that all spins point in the same direction on a global magnetization axis, which we will choose to be the z-axis.

We start from the Heisenberg Hamiltonian:

$$H = -\sum_{mi,nj} J_{mi,nj} \boldsymbol{S}_{mi} \cdot \boldsymbol{S}_{nj}, \tag{3.2}$$

where the indexes $m$ and $n$ sum over the cells, and $i$ and $j$ sum over the sites inside the cells. $J_{mi,nj}$ is the exchange interaction parameter, and has units of energy. $\boldsymbol{S}_{mi}$ is a unitless vector with the direction and magnitude of the spin of the particle $m, i$.

We now introduce the spin ladder operators, that either raise or lower the eigenvalue of $S^z_{mi}$:

$$S^{\pm}_{mi} = S^x_{mi} \pm i S^y_{mi}. \tag{3.3}$$

Performing the scalar product in Equation 3.2 we get:

$$H = -\sum_{mi,nj} J_{mi,nj} \left( S^z_{mi} S^z_{nj} + \frac{S^+_{mi} S^-_{nj} + S^-_{mi} S^+_{nj}}{2} \right). \tag{3.4}$$

We now map the spin ladder operators to boson creation ($a^\dagger_{mi}$) and annihilation ($a_{mi}$) operators by means of the Holstein-Primakoff transformation [50], were we use the **linear approximation**, that is $\frac{<a^\dagger_{mi} a_{mi}>}{2S} \ll 1$, assuming a low excitation state. Then:

$$\begin{aligned} S^+_{mi} &= \sqrt{2S_i} \sqrt{1 - \frac{a^\dagger_{mi} a_{mi}}{2S}}\, a_{mi} \approx \sqrt{2S_i}\, a_{mi}\ , \\ S^-_{mi} &= \sqrt{2S_i}\, a^\dagger_{mi} \sqrt{1 - \frac{a^\dagger_{mi} a_{mi}}{2S}} \approx \sqrt{2S_i}\, a^\dagger_{mi}\ , \\ S^z_{mi} &= S_i - a^\dagger_{mi} a_{mi}\ . \end{aligned} \tag{3.5}$$

Due to the translational symmetry of our system these operators can be Fourier transformed, going from the real space (position), to the reciprocal space (momentum), with the definitions:

$$\begin{aligned} a_{mi} &= N^{-1/2} \sum_{\boldsymbol{k}} e^{i\boldsymbol{k}\boldsymbol{r}_{mi}} a_i(\boldsymbol{k}), \\ a^\dagger_{mi} &= N^{-1/2} \sum_{\boldsymbol{k}} e^{-i\boldsymbol{k}\boldsymbol{r}_{mi}} a^\dagger_i(\boldsymbol{k}). \end{aligned} \tag{3.6}$$

The commutator of the spin ladder operators obeys the following relation,

$$\left[S^+_{mi}, S^-_{nj}\right] = \delta_{m,n}\delta_{i,j}, \tag{3.7}$$

then, by taking into account Equation 3.5,

$$\left[a_{mi}, a^\dagger_{nj}\right] = \delta_{m,n}\delta_{i,j}, \tag{3.8}$$

allowing us to prove the following bosonic commutation relation:

$$\begin{aligned}
\left[a_i(\boldsymbol{k}), a^\dagger_j(\boldsymbol{k})\right] &= N^{-1}\sum_{nm} e^{-i\boldsymbol{k}\boldsymbol{r}_{mi}} e^{i\boldsymbol{k}\boldsymbol{r}_{nj}} \left[a_{mi}, a^\dagger_{nj}\right] \\
&= N^{-1}\sum_{nm} e^{-i\boldsymbol{k}\boldsymbol{r}_{mi}} e^{i\boldsymbol{k}\boldsymbol{r}_{nj}} \delta_{mn}\delta_{ij} \\
&= N^{-1}\sum_{m} e^{-i(\boldsymbol{k}-\boldsymbol{k}')\boldsymbol{r}_{mi}} \delta_{ij} \\
&= \delta_{\boldsymbol{k},\boldsymbol{k}'}\delta_{ij}.
\end{aligned} \tag{3.9}$$

We rewrite the Heisenberg Hamiltonian 3.2 in terms of the $a_i(\boldsymbol{k})$ and $a^\dagger_i(\boldsymbol{k})$ operators:

$$\begin{aligned}
H = -\sum_{mi,nj} J_{mi,nj} \Bigg[ &\left( S_i - N^{-1}\sum_{\boldsymbol{k},\boldsymbol{k}'} a^\dagger_i(\boldsymbol{k}) a_i(\boldsymbol{k}') e^{-i(\boldsymbol{k}-\boldsymbol{k}')\boldsymbol{r}_{mi}} \right) \\
&\left( S_j - N^{-1}\sum_{\boldsymbol{k},\boldsymbol{k}'} a^\dagger_j(\boldsymbol{k}) a_j(\boldsymbol{k}') e^{-i(\boldsymbol{k}-\boldsymbol{k}')\boldsymbol{r}_{nj}} \right) \\
&+ \frac{\sqrt{S_iS_j}}{2} N^{-1} \sum_{\boldsymbol{k},\boldsymbol{k}'} \left( e^{i(\boldsymbol{k}-\boldsymbol{k}')\boldsymbol{r}_{mi}} a_i(\boldsymbol{k}) a^\dagger_j(\boldsymbol{k}') e^{-i\boldsymbol{k}'\boldsymbol{d}} + e^{-i(\boldsymbol{k}-\boldsymbol{k}')\boldsymbol{r}_{mi}} a_j(\boldsymbol{k}) a^\dagger_i(\boldsymbol{k}') e^{i\boldsymbol{k}'\boldsymbol{d}} \right) \Bigg],
\end{aligned} \tag{3.10}$$

where we have defined $\boldsymbol{d}_{mi,nj}$ as $\boldsymbol{r}_{nj} = \boldsymbol{r}_{mi} + \boldsymbol{d}$ and dropped the indexes on $d$ for convenience. Due to the crystal's translational symmetry:

$$J_{mi,nj} = J_{i,j}(\boldsymbol{r}_{mi} - \boldsymbol{r}_{nj}) = J_{i,j}(\boldsymbol{d}). \tag{3.11}$$

After simplifying Equation 3.10, using the commutation relation from Equation 3.9 and keeping terms up to second order in the magnon operators, we obtain the following terms:

- Zeroth order, corresponding to the magnetic ground state:

$$H_{GS} = -\sum_{mi,nj} J_{mi,nj} S_i S_j. \tag{3.12}$$

- Second order, corresponding to the magnetic excitations:

$$\begin{aligned} H_2(\boldsymbol{k}) = \sum_{i,j} \Bigg[ & J_{i,j}(\boldsymbol{0}) \left( S_i a_j^\dagger(\boldsymbol{k}) a_j(\boldsymbol{k}') + S_j a_i^\dagger(\boldsymbol{k}) a_i(\boldsymbol{k}') \right) \\ & - J_{i,j}(\boldsymbol{k}) \frac{\sqrt{S_i S_j}}{2} \left( a_i(\boldsymbol{k}) a_j^\dagger(\boldsymbol{k}) + a_j(\boldsymbol{k}) a_i^\dagger(\boldsymbol{k}) \right) \Bigg], \end{aligned} \tag{3.13}$$

where we have Fourier transformed $J_{i,j}(\boldsymbol{d})$ as:

$$J_{ij}(\boldsymbol{k}) = \sum_{\boldsymbol{d}} J_{i,j}(\boldsymbol{d}) e^{-i\boldsymbol{k}\boldsymbol{d}}. \tag{3.14}$$

This Hamiltonian can be written in matrix form:

$$H_2(\boldsymbol{k}) = \boldsymbol{x}(\boldsymbol{k})^{T,\dagger} h(\boldsymbol{k}) \boldsymbol{x}(\boldsymbol{k}), \tag{3.15}$$

with:

$$\boldsymbol{x}(\boldsymbol{k}) = (a_1(\boldsymbol{k}), a_2(\boldsymbol{k}), ..., a_N(\boldsymbol{k})), \tag{3.16}$$

$$h_{i,j}(\boldsymbol{k}) = \delta_{ij} \sum_l S_l J_{i,l}(\boldsymbol{0}) - J_{i,j}(\boldsymbol{k}) \frac{\sqrt{S_i S_j}}{2}. \tag{3.17}$$

An explicit example of the ferromagnetic case (honeycomb ferromagnet) is given in Section 3.2.3.

### 3.2.2 Single Q case

In this section we will follow the methodology detailed by Haraldsen [51] and later expanded by Toth [52] to calculate the magnetic excitations of non-collinear crystal structures, such as spin spirals.

The introduction of magnetic moments might disturb the symmetries of the crystal, defining a magnetic unit cell (MU) bigger than the chemical unit cell (CU), as seen in Figure 3.2. We introduce the magnetic ordering vector, $\boldsymbol{Q}$,

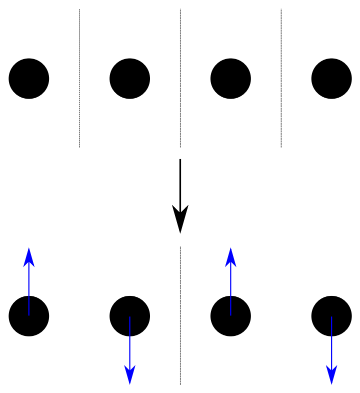


Figure 3.2: For a 1D chain, the introduction of an antiferromagnetic spin alignment gives a magnetic unit cell (MU), bottom, with double the size of the chemical unit cell (CU), top.

in reciprocal space, such that the spin in the m:th CU can be obtained by applying a rotation $R_m$ to the spin in the 0:th CU. With $R_m$ defined by the angle $\varphi = \boldsymbol{Q} \cdot \boldsymbol{r}_m$ around a given axis.

By applying $R_m$ over the spins $S_{m,i}$ our system is transformed into one with the same periodicity as in the CU:

$$\boldsymbol{S}_{mi} = R_m \boldsymbol{S}_{0i} = R_m \boldsymbol{S}_i. \tag{3.18}$$

We can define another rotation to transform our system into ferromagnetic ordering with the matrix $R_i$, such that $\boldsymbol{S}_i = R_i \boldsymbol{S}_i^{'}$, where $i$ iterates over the sites in the unit cell. These rotations are shown in Figure 3.3 for an antiferromagnetic spiral.

The matrix $R_i$ can be expressed by its components:

$$S_i^{\alpha} = \sum_{\mu} R_i^{'\alpha\mu} S_{0i}^{'\mu}. \tag{3.19}$$

As we will need them later, we define the useful vectors:

$$\begin{aligned} u_i^{\alpha\pm} &= R_i^{'\alpha 1} \pm i R_i^{'\alpha 2}, \\ v_i^{\alpha} &= R_i^{'\alpha 3}. \end{aligned} \tag{3.20}$$

With the rotated spins now in a ferromagnetic configuration, we can apply the Holstein-Primakoff transformation. After some work, our spins take the form:

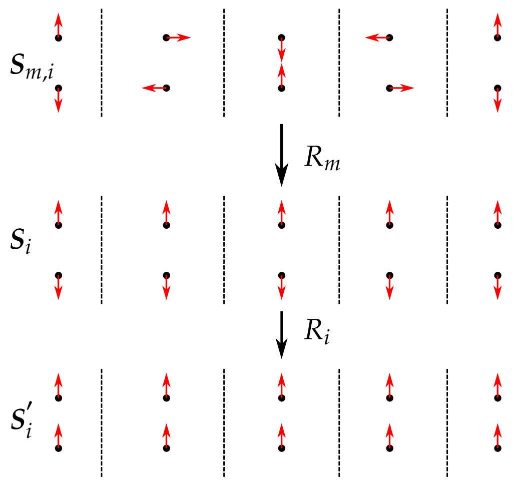


Figure 3.3: Schematic representation of how the rotation matrices $R_m$ and $R_i$ act on a antiferromagnetic spin spiral of $\boldsymbol{Q} = (\frac{1}{4}, 0, 0)$ written in terms of the reciprocal basis vectors.

$$\boldsymbol{S}_{mi} = R_m \left[ \sqrt{\frac{S_i}{2}} (\boldsymbol{u}_i^+ a_{mi} + \boldsymbol{u}_i^- a_{mi}^\dagger) + \boldsymbol{v}_i (S_i - a_{mi}^\dagger a_{mi}) \right]. \tag{3.21}$$

With this result, we proceed as in Section 3.2.1: we Fourier transform the $a_{mi}$ and $a_{mi}^\dagger$ operators; obtain the Hamiltonian; and separate by order in magnon operators, obtaining:

- Zeroth order: $H_0 = -\sum_{mi,nj} S_i S_j \boldsymbol{v}_i^T J'_{mi,nj} \boldsymbol{v}_j$, the ground state energy.

  Where we have used $J'_{mi,nj} = R_m^T J_{mi,nj} R_n$.

- Second order:

$$\begin{aligned} H_2 = -\sum_{i,j,k} \frac{\sqrt{S_i S_j}}{2} \Big( & \boldsymbol{u}_i^{T,+} J'_{ij}(\boldsymbol{k}) \boldsymbol{u}_j^+ a_i(\boldsymbol{k}) a_j(-\boldsymbol{k}) + \boldsymbol{u}_i^{T,+} J'_{ij}(-\boldsymbol{k}) \boldsymbol{u}_j^- a_i(\boldsymbol{k}) a_j^\dagger(\boldsymbol{k}) \\ & \boldsymbol{u}_i^{T,-} J'_{ij}(\boldsymbol{k}) \boldsymbol{u}_j^+ a_i^\dagger(\boldsymbol{k}) a_j(\boldsymbol{k}) + \boldsymbol{u}_i^{T,-} J'_{ij}(-\boldsymbol{k}) \boldsymbol{u}_j^- a_i^\dagger(\boldsymbol{k}) a_j^\dagger(-\boldsymbol{k}) \Big) - \\ & \boldsymbol{v}_i^T J_{ij}'(0) \boldsymbol{v}_j \left[ S_i a_j(\boldsymbol{k}) a_j^\dagger(\boldsymbol{k}) + S_j a_i(\boldsymbol{k}) a_i^\dagger(\boldsymbol{k}) \right], \end{aligned} \tag{3.22}$$

  with $J'_{ij}(\boldsymbol{k}) = \sum_{\boldsymbol{d}} J'_{ij}(\boldsymbol{d}) e^{i\boldsymbol{k}\boldsymbol{d}}$, as in Equations 3.11 and 3.14.

  This Hamiltonian can be written in matrix form as:

$$H_2(\boldsymbol{k}) = \boldsymbol{x}(\boldsymbol{k})^{T,\dagger} h(\boldsymbol{k}) \boldsymbol{x}(\boldsymbol{k}), \tag{3.23}$$

with:

$$x(\boldsymbol{k}) = \Big(a_1(\boldsymbol{k}), ..., a_N(\boldsymbol{k}), a_1^\dagger(-\boldsymbol{k}), ..., a_N^\dagger(-\boldsymbol{k})\Big)^T, \tag{3.24}$$

and $h(\boldsymbol{k})$ a 2N by 2N matrix:

$$h(\boldsymbol{k}) = -\begin{pmatrix} A(\boldsymbol{k}) - C & B(\boldsymbol{k}) \\ B(\boldsymbol{k})^\dagger & \bar{A}(-\boldsymbol{k}) - C \end{pmatrix}, \tag{3.25}$$

formed by the N by N Matrices:

$$\begin{aligned} A_{ij}(\boldsymbol{k}) &= \frac{\sqrt{S_i S_j}}{2} \boldsymbol{u}_i^{T,-} J'_{ij}(\boldsymbol{k}) \boldsymbol{u}_j^+, \\ B_{ij}(\boldsymbol{k}) &= \frac{\sqrt{S_i S_j}}{2} \boldsymbol{u}_i^{T,+} J'_{ij}(\boldsymbol{k}) \boldsymbol{u}_j^+, \\ C_{ij}(\boldsymbol{k}) &= \delta_{ij} \sum_l S_l \boldsymbol{v}_i J'_{il}(\boldsymbol{0}) \boldsymbol{v}_l. \end{aligned} \tag{3.26}$$

#### 3.2.2.1 Antiferromagnet

In this section we will apply single Q LSWT to a 2D-crystal along the XY plane with two sites per CU, with spins pointing in opposite direction along the z-axis and same magnitude, $S_1 = -S_2 = S, \quad J < 0$.

As our CU and MU coincide, $R_m = \mathbb{1}$, so $J'(\boldsymbol{k}) = \mathbb{1} J(\boldsymbol{k}) \mathbb{1} = J(\boldsymbol{k})$.

We will denote the particle with spin pointing upwards as $i = 1$, and downwards $i = 2$. In order to bring our system to ferromagnetic ordering we need to do a rotation of $\pi$ along the y-axis, giving:

$$R_1 = \mathbb{1} \quad , \quad R_2 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}. \tag{3.27}$$

Then the vectors $\boldsymbol{u}_i^\pm$ and $\boldsymbol{v}_i$ are:

$$\begin{aligned} \boldsymbol{v}_1 &= \boldsymbol{z}, \\ \boldsymbol{u}_1^\pm &= \boldsymbol{x} \pm i\boldsymbol{y}, \\ \boldsymbol{v}_2 &= -\boldsymbol{z}, \\ \boldsymbol{u}_2^\pm &= -\boldsymbol{x} \pm i\boldsymbol{y}. \end{aligned} \tag{3.28}$$

Hence, Equation 3.26 becomes:

$$\begin{aligned}
A &= \frac{S}{2}J(\boldsymbol{k})\begin{pmatrix} 0 & \boldsymbol{u}_1^{T,-}\boldsymbol{u}_2^{+} \\ \boldsymbol{u}_2^{T,-}\boldsymbol{u}_1^{+} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \\
B &= \frac{S}{2}J(\boldsymbol{k})\begin{pmatrix} 0 & \boldsymbol{u}_1^{T,+}\boldsymbol{u}_2^{+} \\ \boldsymbol{u}_2^{T,+}\boldsymbol{u}_1^{+} & 0 \end{pmatrix} = SJ(\boldsymbol{k})\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}, \\
C &= \begin{pmatrix} S\boldsymbol{v}_1 J(\boldsymbol{0})\boldsymbol{v}_2 & 0 \\ 0 & S\boldsymbol{v}_2 J(\boldsymbol{0})\boldsymbol{v}_1 \end{pmatrix} = SJ(\boldsymbol{0})\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix},
\end{aligned} \tag{3.29}$$

where we have dropped the index $J_{12} = J$. Thus Equation 3.25 takes the form:

$$H_2(\boldsymbol{k}) = -\begin{pmatrix} SJ^{'}(\boldsymbol{0}) & 0 & 0 & -SJ^{'}(\boldsymbol{k}) \\ 0 & SJ^{'}(\boldsymbol{0}) & -SJ^{'}(\boldsymbol{k}) & 0 \\ 0 & -SJ^{'}(-\boldsymbol{k}) & SJ^{'}(\boldsymbol{0}) & 0 \\ -SJ^{'}(-\boldsymbol{k}) & 0 & 0 & SJ^{'}(\boldsymbol{0}) \end{pmatrix}, \tag{3.30}$$

which can be diagonalized by taking into account the commutation relation of the bosonic operators, as detailed in Colpa's [53] and Toth's [52] work, giving the following degenerate eigenvalue:

$$E(\boldsymbol{k}) = \sqrt{SJ(\boldsymbol{0})^2 - S|J(\boldsymbol{k})|^2}. \tag{3.31}$$

### 3.2.3 Honeycomb Magnets: Ferromagnetic and Antiferromagnetic case

In this section we will use LSWT to analyze magnon excitations for both the ferromagnetic and antiferromagnetic case for the honeycomb lattice.

A honeycomb lattice (as seen in Figure 3.4) is a two site lattice in which the sites arrange themselves in an hexagonal pattern characteristic of bees' honeycombs. Such a lattice has been shown to display Dirac magnons in ferromagnets [19], such as chromium trihalides, $CrX_3$, where X is a halogen e.g. $CrF_3$.

The honeycomb lattice can be described by the lattice vectors:

$$\boldsymbol{a}_{\pm} = \frac{a}{2}\left(\sqrt{3}, \pm 1\right), \tag{3.32}$$

and an atomic basis of two atoms distanced $\frac{a}{\sqrt{3}}$ apart along the x-axis.

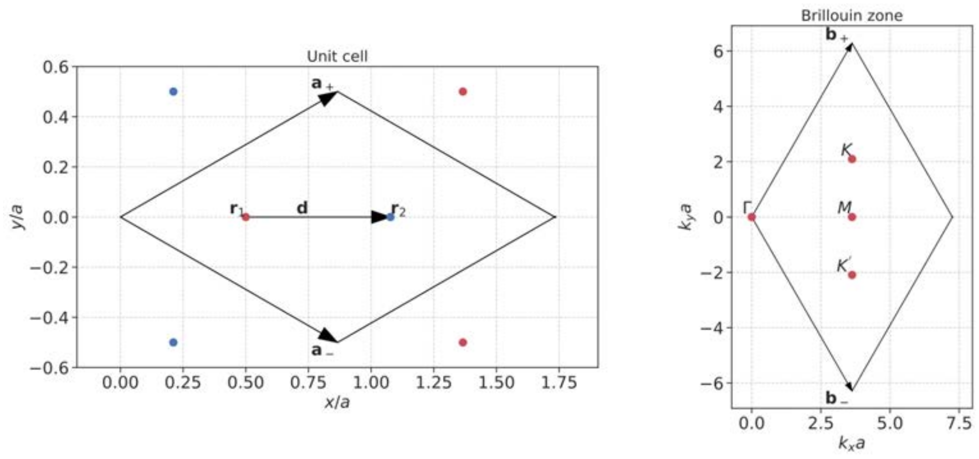


Figure 3.4: Honeycomb lattice (left) and reciprocal unit cell (right). The lattice is defined by the two sites, at $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, and the lattice vectors, $\boldsymbol{a}_\pm$. The vector $\boldsymbol{d}$ is the difference in the positions of the sites.

By taking into account only the nearest neighbors interactions, we obtain:

$$J(\boldsymbol{k}) = J\left(e^{i\boldsymbol{k}\boldsymbol{d}} + e^{i\boldsymbol{k}(\boldsymbol{d}-\boldsymbol{a}_-)} + e^{i\boldsymbol{k}(\boldsymbol{d}-\boldsymbol{a}_+)}\right), \tag{3.33}$$

with $\boldsymbol{d}$ being the vector from site two to site one. As it will be useful later, $J(\boldsymbol{0}) = 3J$ and

$$|J(\boldsymbol{k})| = J\sqrt{3 + 2\cos \boldsymbol{k}\boldsymbol{a}_+ + 2\cos \boldsymbol{k}\boldsymbol{a}_- + 2\cos \boldsymbol{k}(\boldsymbol{a}_+ - \boldsymbol{a}_-)}. \tag{3.34}$$

#### Ferromagnet

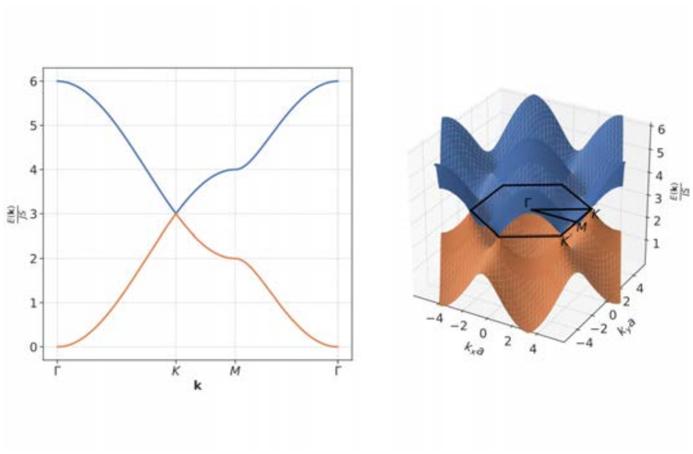
From Equation 3.15 and assuming a two site ferromagnet with only nearest neighbor interaction:

$$S_1 = S_2 = S, \quad J > 0, \tag{3.35}$$

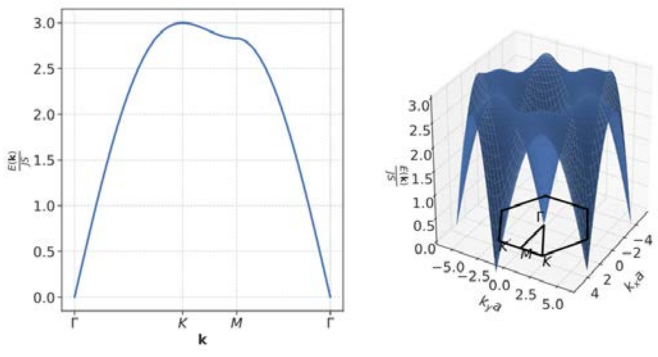
then, we obtain the following eigenvalues from the Hamiltonian in Equation 3.17:

$$E(\boldsymbol{k}) = SJ(\boldsymbol{0}) \pm S|J(\boldsymbol{k})|, \tag{3.36}$$

These eigenvalues are plotted, in 2D and 3D, in Figure 3.5a, where we can see linear crossings, or Dirac points, at the high symmetry points K and K$^{'}$.



(a) Magnon dispersion of a ferromagnetic honeycomb lattice in 2D (left) and 3D (right). Dirac points are present at the K and K$^{'}$ points, the latter only seen in the 3D plot.



(b) Magnon dispersion of an antiferromagnetic honeycomb lattice in 2D (left) and 3D (right). A linear crossing is present at the Γ point.

Figure 3.5: Magnon dispersions in a honeycomb lattice.

### Antiferromagnet

Using the two site antiferromagnet Equation 3.31, we can obtain the magnon dispersion for a honeycomb lattice in antiferromagnetic configuration, displayed in Figure 3.5b. It displays a linear crossing at the ground state, recognizable by the cone in the 3d plot at the Γ point.

## 3.3 Prediction of Heisenberg Hamiltonians

In Section 3.2 we have shown how to use LSWT to characterize low energy excitations of collinear and non-collinear magnets, given the full Heisenberg Hamiltonian. Now we will discuss how to formulate the Heisenberg Hamiltonian for an unknown complex organic crystal, using ML, for which we need to predict two quantities: the magnetic moment per site, $\boldsymbol{m}_i$ and the Heisenberg exchange parameters, $J_{ij}$.

However, we will predict the magnitude of this vector $|\boldsymbol{m}_i|$, as it is a simpler problem and the ground state determination [21] is not so computationally expensive.

In this section we will apply the methods introduced in Chapter 2 to predict these magnetic properties using the materials in the OMDB. We will divide the process into four steps:

1. Classify crystals as magnetic or not.
2. For the crystals that are magnetic, classify their sites as magnetic or not.
3. For the sites that are magnetic, obtain their magnetization $|m_i|$.
4. Obtain the exchange interaction parameters, $J_{ij}$, for every pair of sites.

Once these steps have been completed, we will have all the parameters necessary to calculate the magnetic ground state [21], and spin excitations as per Section 3.2, bypassing $|m_i|$ and $J_{ij}$ calculations that can take thousands of cpu-hours per material.

An schematic representation of the OMDB dataset is presented in Figure 3.6, from where we conclude that the first two steps are necessary because 99.3 % of the sites in the OMDB are non-magnetic, therefore, trying to directly predict site magnetization would make the model be extremely conservative, always predicting magnetization values close to zero.

### 3.3.1 Crystal Structures

In this section we aim to present a ML model able to predict if a crystal is magnetic or not in unseen data.

We are using data from the OMDB [8], containing over 22000 organic crystals, with properties calculated by DFT. Our dataset is composed by the 12991 of those that have magnetization per site calculated.

We will define magnetic materials as those with at least one site with magnetization larger than $0.1\mu_B$. According to this definition, 2048 out of the 12991 crystals are magnetic.

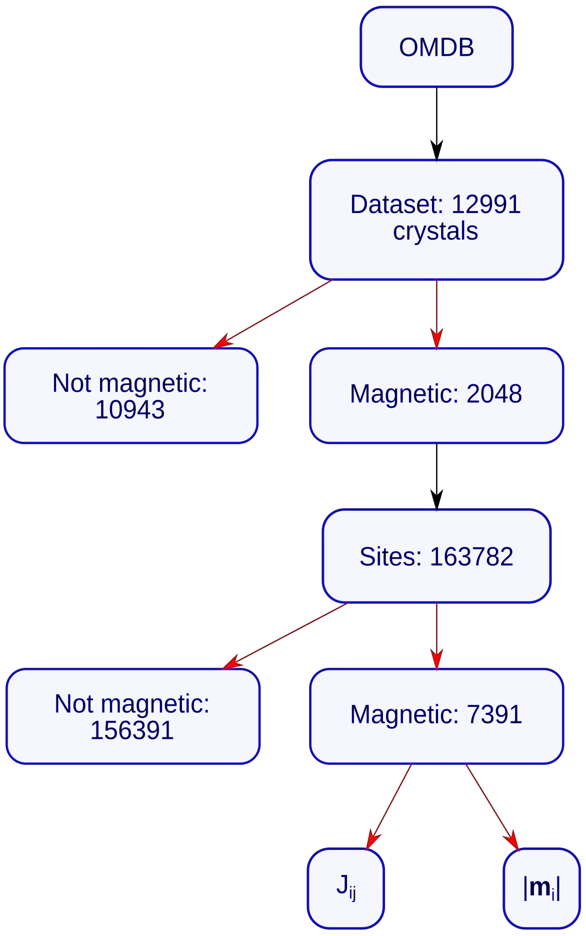


Figure 3.6: Schematic representation of the dataset. The red arrows indicate where ML can be used to classify (magnetic and not magnetic), or to do regression ($J_{ij}$, $|\mathbf{m}_i|$).

In order to evaluate our model's ability to make predictions in materials outside our dataset, we will divide our data into a training set and a test set, with sizes of 80% and 20% of the total dataset respectively. Which material was in which set was decided randomly.

#### 3.3.1.1 Baseline Models

Our training set is composed of 10392 materials, of which 1658 are magnetic, and the test set is formed of 2599 materials, 390 of them magnetic. The proportion of magnetic materials versus non-magnetic ones makes it an *unbalanced dataset problem*, which, as discussed in Section 2.4.1, requires us to examine different metrics to evaluate and discuss proposed models.

As a first step, we will analyze the *scalar model*, defined as always predicting the majority class, that is, **our model will always predict that a material is not magnetic**, no matter what crystal structure is used as input. Using this model we obtain an accuracy of 0.84, which in another problem could be an excellent model, however, by looking at F1, which evaluates to 0, we know that the scalar model has no actual predictive capabilities.

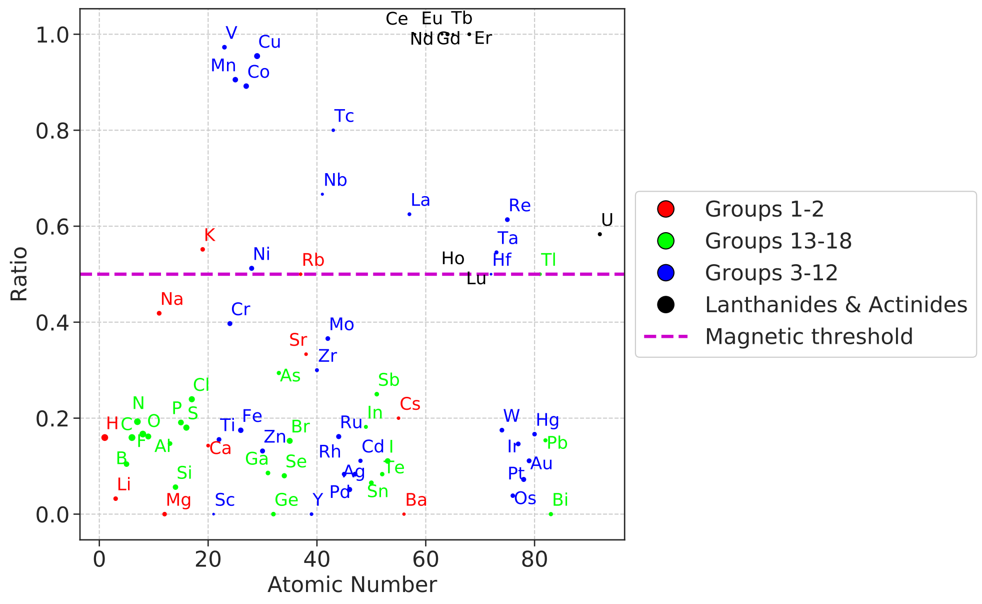


Figure 3.7: For a given chemical species we display the ratio of magnetic materials. Elements are colored according to their IUPAC (International Union of Pure and Applied Chemistry) group. To avoid cluttering, only elements that are present on more than 10 materials are displayed.

In Figure 3.7 we display the ratio of magnetic materials for all the materials in the training set that contain a given chemical species. We observe that the presence or absence of certain chemical species is a strong indicator of a material being magnetic or not. In fact, we can define a simple model that classifies a material as magnetic if it contains any element with a ratio over 0.5 (purple line on figure). By evaluating this model in the test set, we obtain the following metrics:

- Accuracy = 0.951;
- F1 = 0.831;
- Confusion Matrix:

| | **Predicted: Positive** | **Predicted: Negative** |
|---|---|---|
| **Actual: Positive** | 313 | 77 |
| **Actual: Negative** | 50 | 2159 |

We will consider this model as the *baseline model* due to its simplicity, and that we will discard any model that performs worse.

#### 3.3.1.2 Machine learning models

In this section we analyze the performance of two different representations, SOAP [Sec. 2.2.1.3] and Multi-Hot [Sec. 2.2.1.1], for predicting magnetic crystals. Which we combine with the following algorithms: for SOAP we employed Support Vector Machines (SVM) [Sec. 2.3.3] as it is ideal for a kernel based classification problem, and for Multi-Hot we will compare Random Forests (RF) [Sec. 2.3.2] and Artificial Neural Networks (ANN) [Sec 2.3.1], as they are algorithms widely used in classification tasks.

For all the models presented, the hyperparameters of the model have been optimized using random search with cross validation on the training set.

In Figure 3.8 we can see our models' performance as a function of the training set sample. We observe an increase in quality of our model with the increase in size of the training set, however, diminishing returns soon kick in.

By comparing the models we observe that the baseline model quality is almost independent of the number of training samples, performing well even for a small number of training samples.

Despite its simplicity, Multi-Hot representation proves to be a more than adequate descriptor of a crystal for this task, indicating that the chemical formula is the main determinant of magnetic presence in an organic crystal. However, it has a restricted input domain, so once the model has observed most variations of inputs it will keep making the same mistakes no matter how many more training samples, e.g. Multi-Hot ANN past 4000 materials stops improving.

Furthermore, there exists materials with the same chemical elements where one exhibits magnetic properties and other ones don't, for example: the materials with COD ID [54] 2218516 and 4021177 have the same elements present (C, H, Ni, O), and so the same Multi-Hot representation, but only the latter is magnetic. To illustrate this limit, in Figure 3.8 we include an horizontal line representing the model *Ideal Multi-Hot*, a model that for each possible Multi-Hot input representation on the test set always predicts the class which will maximize the accuracy. This model was trained and evaluated in the test set to indicate the maximum possible metrics Multi-Hot models could achieve, evaluating it in other sets of materials will give worse results.

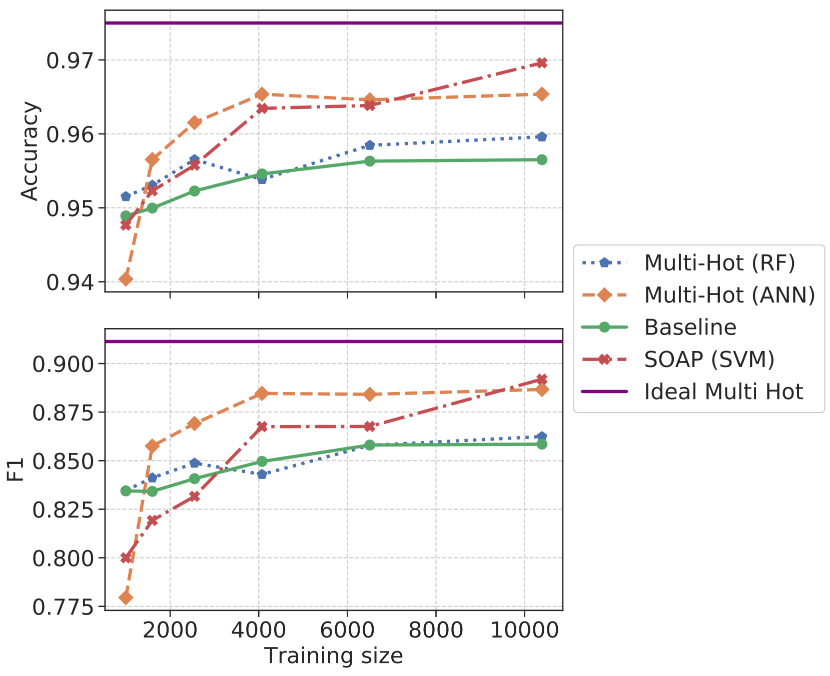


Figure 3.8: Performance of the model according to the number of training samples.

On the other hand, SOAP, a much more complex representation that has knowledge of the crystal structure, is able to obtain the best results for a large amount of training samples, with accuracy and F1 continuing to increase with the number of training samples, unlike the other descriptors. For the full training set, SOAP achieves the following metrics:

- Accuracy $= 0.97$;
- F1 $= 0.89$;
- Confusion Matrix:

| | **Predicted: Positive** | **Predicted: Negative** |
|---|---|---|
| **Actual: Positive** | 326 | 64 |
| **Actual: Negative** | 14 | 2194 |

Both metrics, accuracy and F1 can take values in the interval $[0, 1]$. By plotting $\log_{10}(1 - metric)$, Figure 3.9, we observe a roughly linear behavior in the logarithmic space as the number of training samples increases.

We can then fit a linear function and extrapolate it, which will allow us to evaluate the potential performance of our models as we increase our dataset's size. For F1:

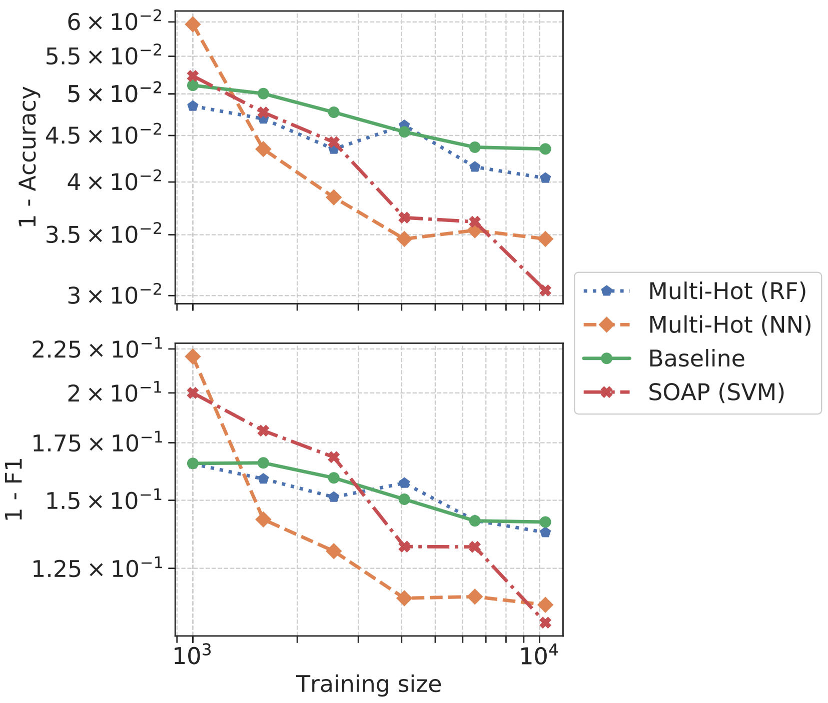


Figure 3.9: Logarithmic plot of the performance of the model according to the number of training samples.

$$\log(1 - F1) = m \log(N) + C \rightarrow F1 = 1 - 10^C N^m. \tag{3.37}$$

For the best model, SOAP, for accuracy we obtain the values $m = -0.23$ and $C = -0.59$; for F1 we obtain $m = -0.26$ and $C = -0.6$. Extrapolating to $10^5$ materials, a realistic number for the OMDB in the following years, would give a F1 score of 0.94 and an accuracy of 0.98.

### 3.3.2 Site Classification

After separating the magnetic materials from the non-magnetic ones we can proceed to the next step described in Figure 3.6: for those materials that are magnetic, we will try to determine which sites contribute to the magnetic Hamiltonian. We will say that a site is magnetic if it has a magnetization magnitude of at least $0.1\mu_B$.

This is a different problem, with different challenges to the previous one. There is no literature about how to represent specific sites in a crystal, nor there is information about how to use ML to predict a local property. We addressed this issue in 2.2.2 by introducing new representations, able to represent the central site and its environment.

In average, every crystal in the OMDB has 80 sites, meaning that our dataset's size has now increased by two orders of magnitude compared to the previous task of identifying magnetic materials, rendering kernel methods impractical due to the $N^2$ scaling in memory and time to generate the kernel matrix. For example, in order to create a $N \times N$ kernel of the whole site dataset (N=163782) using double precision floats, the matrix will require over 210 GB of RAM. While that is achievable with current computers, our limitations in computing power and available time renders kernel methods impossible to analyze in this work. Moreover, one of the objectives of using ML methods in this work is to reduce computational load, making it counterproductive to investigate kernel methods for this problem.

From the previous dataset we have 2048 magnetic materials, whose 163782 sites and magnetizations comprise the new dataset. Of these, only 7391 are magnetic, making it, as in the crystal case, an unbalanced dataset problem.

Adding to the complexity, in order to accurately characterize the Heisenberg Hamiltonian, it is important to correctly classify all sites in a material, as just one mistake in the predictions will completely change the predicted properties, e.g. missing one site in an antiferromagnetic material might lead to it being predicted as ferromagnetic.

#### 3.3.2.1 Baseline Models

We will proceed in a similar fashion to earlier by stating a baseline model that will serve as a minimum threshold for other models to improve.

In Figure 3.10 we display the ratio of magnetic sites per chemical species for the materials in the training set. There is a much clearer influence of the chemical group in the ratio of magnetic elements with respect to the crystal case, seen in Figure 3.7.

For our baseline model we will consider the elements with a ratio of magnetic sites over 0.5 as magnetic, and predict all the sites of that element as such. When this model is evaluated on the test dataset we obtain the following metrics:

- Accuracy = 0.975
- F1 = 0.642;
- Confusion Matrix:

| | **Predicted: Positive** | **Predicted: Negative** |
|---|---|---|
| **Actual: Positive** | 695 | 751 |
| **Actual: Negative** | 23 | 29722 |

This model has more errors—in particular with the False Negatives (FN)—which in turn means less accuracy, than a model that always predicts non-magnetic, demonstrating the biggest problem in unbalanced data problems:

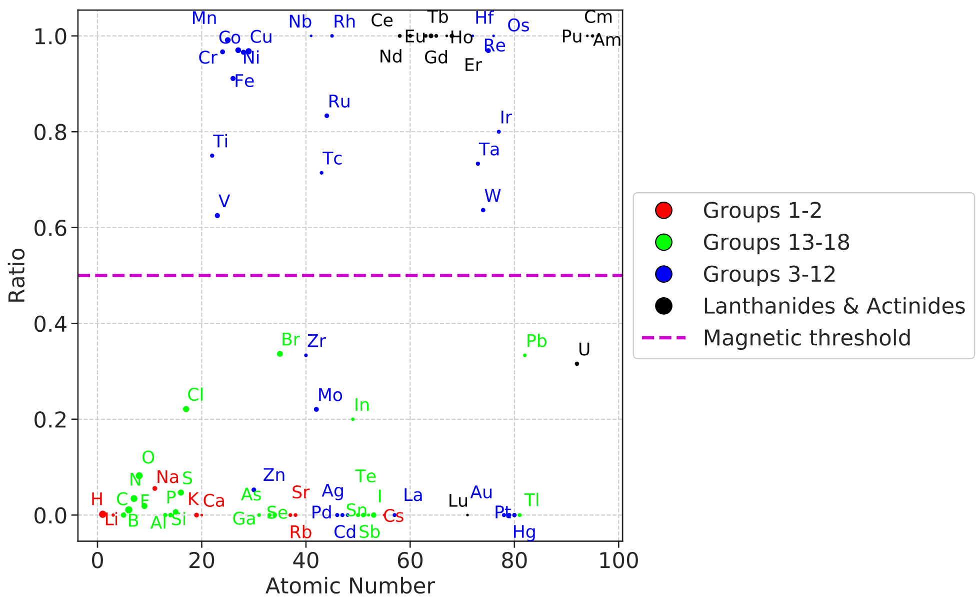


Figure 3.10: For a given chemical species we display the ratio of magnetic sites. Elements are colored according to their IUPAC group.

increasing the number of true positives might increase the number of false positives at a faster rate.

This baseline model performs worse than the equivalent one for crystals, in Section 3.3.1.2], pointing to this being a more challenging problem.

#### 3.3.2.2 Machine Learning Models

In this section we will analyze three representations, Coulomb Matrix (CM), Coulomb Vector (CM Vector) and Radial Multi-Hot (RMH), described in Section 2.2.2, for predicting magnetic sties. Each of these representations is then used as an input for two ML algorithms, Artificial Neural Networks (ANN) and Random Forests (RF), from Section 2.3.

In Figure 3.11 we display the performance metrics of the representations with its best performing algorithm and the dependence on the number of training samples. As seen in the figure, ML models quickly outperform the baseline model, particularly for F1, as the baseline has a high number of false negatives. Furthermore, every ML model achieves better accuracy than always predicting non-magnetic for the maximum training size.

RMH outperforms the other models, achieving an accuracy of 0.983 and a F1 of 0.797, highlighting the importance of not biasing our representation to assume similarity based on the atomic number, but instead letting our model figure out which chemical species are similar by itself.

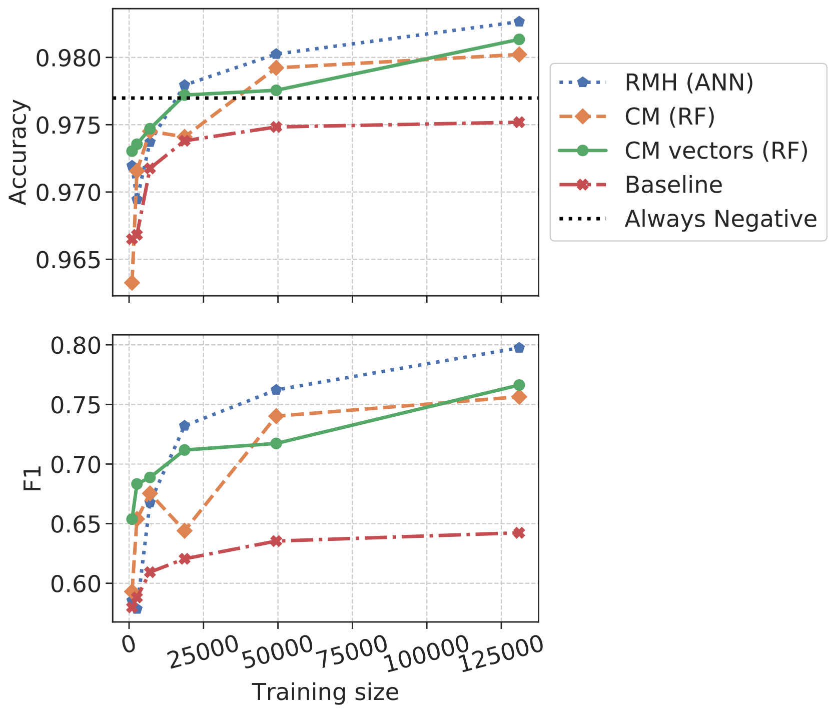


Figure 3.11: Comparison of the performance of the models according to the number of training samples.

These metrics were evaluated in a test set comprised of the sites from the same materials as the magnetic materials from the test set from Section 3.3.1. However, if we were processing a completely new material, according to Figure 3.6, the false negatives from classifying a material as magnetic or not, that is, materials which we incorrectly predicted as non-magnetic, will not propagate into this step. By removing them from the test set, we obtain an accuracy of 0.988 and a F1 of 0.862—as seen in figure 3.12—improving both our metrics, revealing a correlation between mistakes made in the first step and the ones made in the second step. In fact, our RMH model predicts all sites as non-magnetic for all the false positive crystals.

From now on, all the results will have this false positives removed.

In table 3.1 we can see the confusion model for our RMH at the maximum number of training samples, and in Figure 3.12 we observe that the performance of all our models improves after the removal of the false negatives.

Following the previous logic, we again analyze the performance of our model in logarithmic space, as seen in Figure 3.13, revealing again linear behavior.

We then extrapolate our metrics to $10^6$ sites, one order of magnitude more

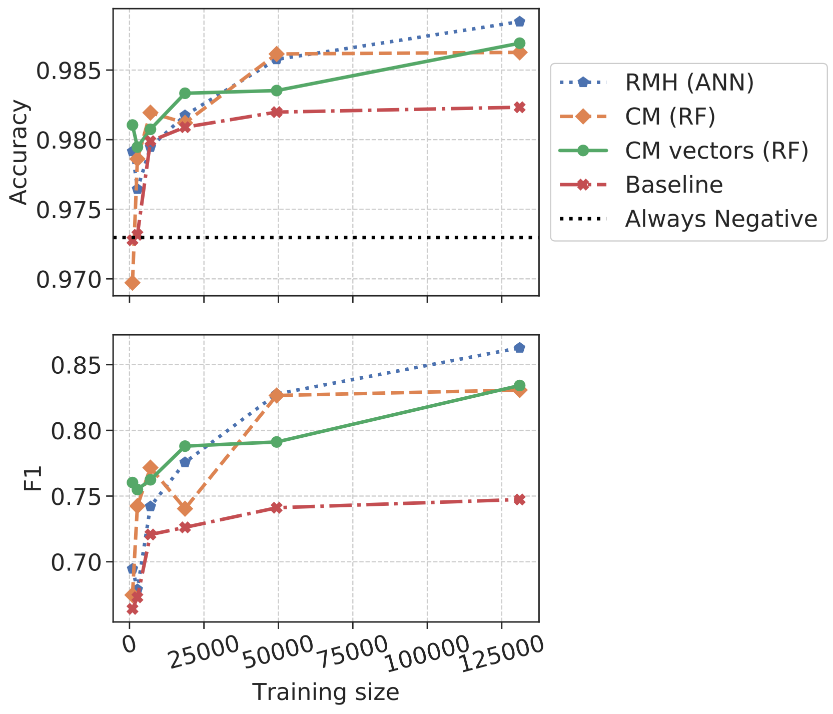


Figure 3.12: Performance of the model according to the number of training samples after removing false negatives from the previous problem.

| | **Predicted: Positive** | **Predicted: Negative** |
|---|---|---|
| **Actual: Positive** | 937 | 173 |
| **Actual: Negative** | 125 | 24624 |

Table 3.1: Confusion Matrix for RMH model.

than the current number, giving a F1 score of 0.89 and an accuracy of 0.99, giving an idea of achievable performance in the near future.

The errors in site predictions are concentrated in a few materials, such that 81 % of the materials have all their sites predicted correctly. However, we need to take into account the false positives from crystal classification, which do propagate into this section, resulting in 76 % of the materials displayed as magnetic having all their sites correctly classified as magnetic or not.

### 3.3.3 Magnetization Regression

Now that we are able to discern magnetic sites from non-magnetic ones we will develop models capable of predicting the magnetization of the magnetic sites, following the flowchart in Figure 3.6.

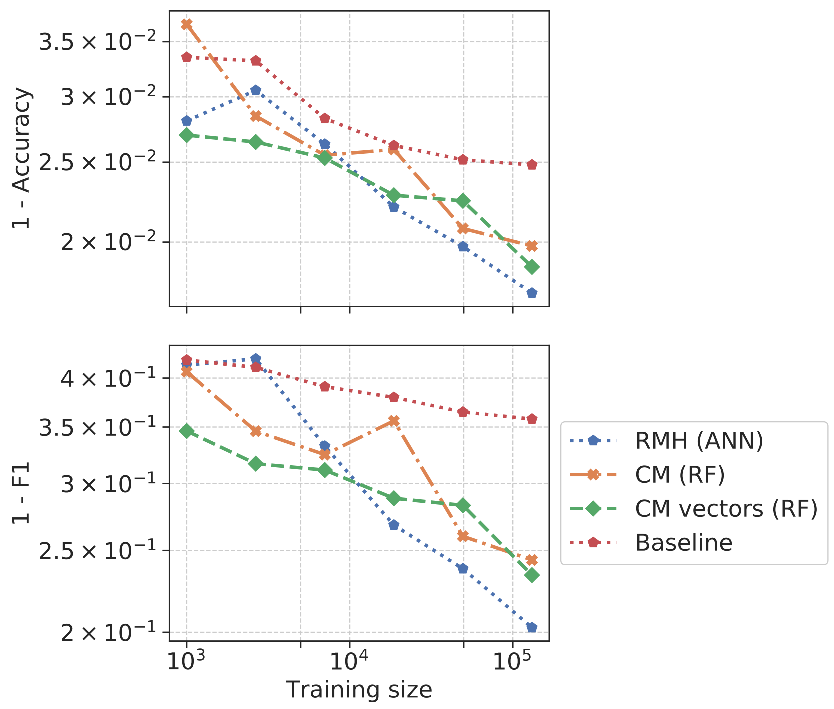


Figure 3.13: Logarithmic plot of the performance of the model according to the number of training samples.

As this is a regression problem, we will use the metrics defined in Section 2.4.2, such as $r^2$, mean absolute error (MAE), and root mean squared error (RMSE).

The dataset consists of 7391 materials, with an average magnetization of $1.01\mu_B$ and a standard deviation of $1.34\mu_B$. This dataset is then randomly split into training and testing sets, with 80 % of the samples in the training set. The distribution of magnetizations and number of sites can be seen in Figure 3.14.

#### 3.3.3.1 Baseline Model

We will define our baseline model by only considering the chemical species of the central site, the one we are predicting magnetization for, and predicting the mean magnetization of all the training samples from this chemical species. In Figure 3.15, we show the mean values and standard deviation of the train set species by species.

We can see that transition and rare earth metals tend to have higher magnetizations—the latter with the highest values—than other element types. The materials with higher standard deviation have multiple mag-

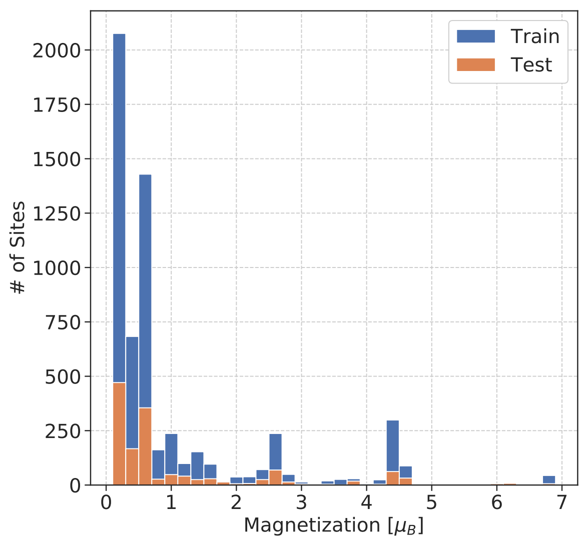


Figure 3.14: Magnetization histogram, only magnetic ($|\boldsymbol{m}_i| > 0.1\mu_B$) sites are displayed. The width of each bar is $0.2\mu_B$

netization values present in the dataset, which our model is not able to differentiate, due to it only knowing the species of the central site.

As in the previous chapter, for evaluating our model in the test dataset, we will discard the false negatives from the crystal classification as they wouldn't have propagated to this step, and wouldn't be representative of the predictions on an unknown crystal.

In Figure 3.16 we show the performance of our baseline model in the test dataset. We plot the predicted magnetization on the y-axis and our ground truth, VASP magnetization, on the x-axis. In an ideal model, without any error, all the samples would lie over the identity line, which we represent as a black continuous line. However, due to the simplicity of our model, the points are clustered in horizontal lines, each line corresponding to the mean magnetization value of a given element.

By applying this model on the test dataset, we obtain the following metrics:

- $r^2 = 0.858$;
- MAE $= 0.294$;
- RMSE $= 0.55$.

This $r^2$ value means that we explain just over 85% of the variance of our dataset, a remarkable performance for such a simple model. However, our model struggles the most with species with high standard deviation (see Figure 3.16).

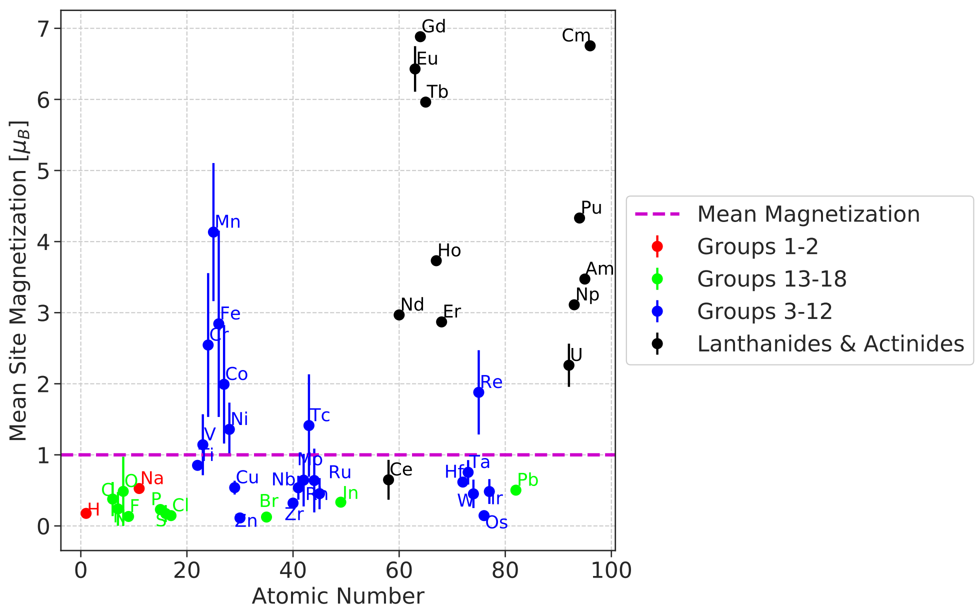


Figure 3.15: Mean and standard deviation of magnetization per site for each chemical species in the train set. Elements are colored according to their IUPAC group.

In the right panel of Figure 3.16 we plot the predicted versus real (VASP) plot for Cobalt, Manganese and Oxygen. We observe that the real magnetization values tend to be around a number of clusters, which correspond to sites with a different number of unpaired electrons, from different oxidation states. These species were selected as they had high variation in magnetization and enough samples to be statistically significant and of quite general interest.

### 3.3.4 Machine Learning models

In this section we will analyze the same three representations as in Section 3.3.2, this time in a regression context.

In Figure 3.17 we observe that the models that promote site information (CM vectors, RMH) outperform those that don't (Baseline, CM). Furthermore, all of our models outperform the baseline, and RMH is the best performing for most sizes of the training set, confirming that the atomic number, $Z$, is not an adequate descriptor for chemical species compared to the less biased approach of letting the model figure out which species are related to each other.

For our best model (RMH), the following metrics are obtained:

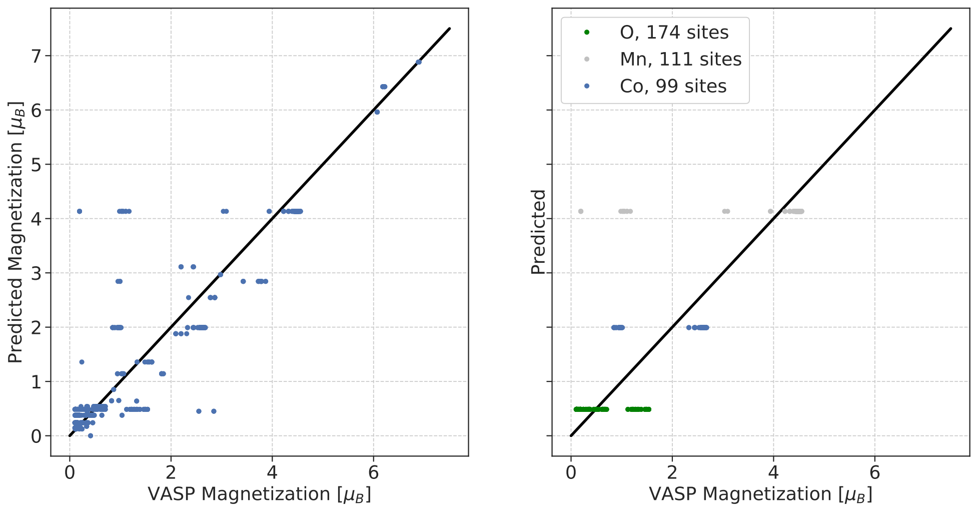


Figure 3.16: Predicted versus real plot of the baseline model for the test dataset. In the right plot we only plot manganese, cobalt and oxygen.

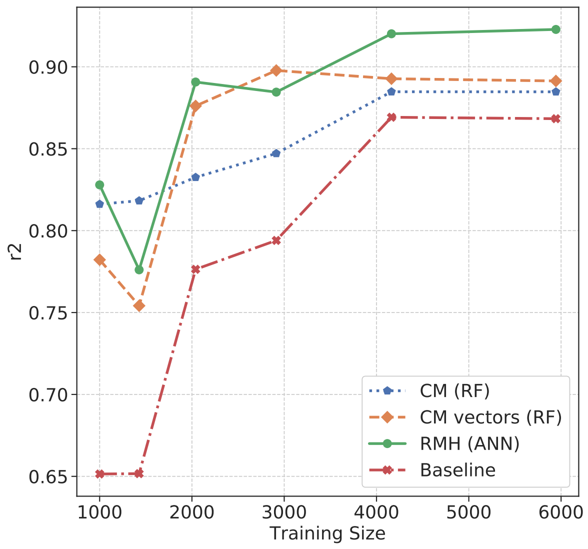


Figure 3.17: Comparison of the performance ($r^2$) of the models according to the number of training samples.

- $r^2 = 0.923$;
- MAE $= 0.21$;
- RMSE $= 0.42$.

In order to understand where our model performs worse and why, in Figure 3.18, we plot predicted versus real (VASP) magnetization for our test dataset.

Three regions of this graph stand out due to poor performance, and they are marked by transparent vertical bands. These bands relate to the clusters from different oxidation states, discussed in the context of figure 3.16, not being correctly identified, e.g. the overestimated values in the red band are due to our model predicting values from the yellow band.

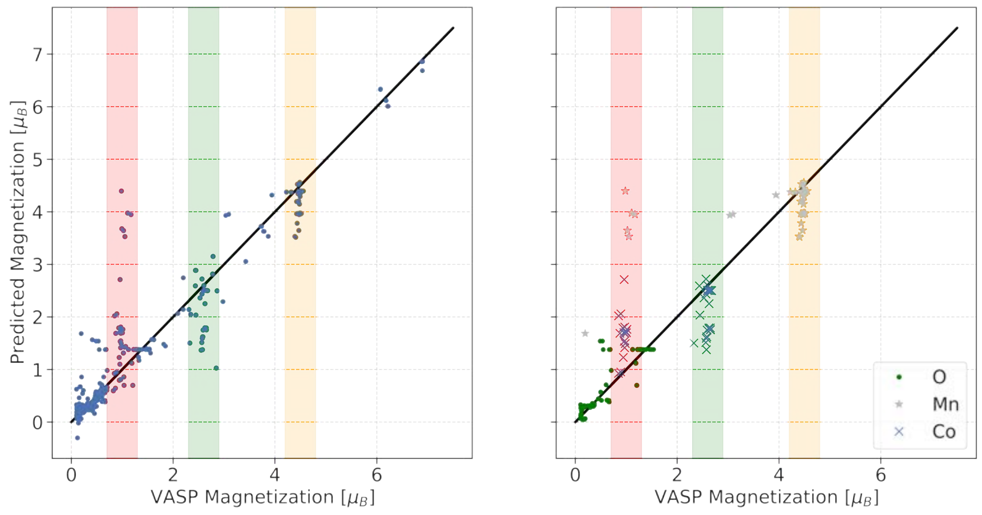


Figure 3.18: Predicted versus real plot of the best model for the full test dataset (left) and only O, Mn and Co . The vertical bands indicate the regions where the model performs the worst.

Even though this is a regression problem, the clustering of target values around different points lets us consider predicting points in different clusters as classification errors. In order to study this, three species were selected for the central atom: O, Mn and Co. To analyze them, we will define a new metric, $r_s^2$ which takes the same form as $r^2$, considering only a certain specie, $s$, instead of the whole set:

$$r_s^2 = 1 - \sum_{i \in \text{test}} \frac{(y_{i,s}^{\text{pred}} - y_{i,s}^{\text{target}})^2}{(y_{i,s}^{\text{target}} - \bar{y}_s^{\text{target}})^2} , \tag{3.38}$$

giving:

- O: $r_O^2 = 0.70$;
- Mn: $r_{Mn}^2 = 0.44$;
- Co: $r_{Co}^2 = 0.22$.

We observe that the $r_s^2$ values are smaller than the $r^2$ value for the whole dataset, $r_2 = 0.921$, due to most of the variance being explained with just the species of the central atom. However, a positive value of $r^2$ shows that our model is able to explain some of the remaining variance.

### 3.3.5 Magnetization Density

We define the magnetization density as the net magnetization per unit of volume a material would have if it was ferromagnetic,

$$\rho_m = \sum_i \frac{|\boldsymbol{m}_i|}{V}, \tag{3.39}$$

where $i$ iterates over all the sites in the unit cell of volume $V$. $\rho_m$ is a global property of the crystal that can be expressed in terms of local properties of the sites, so it makes an ideal target to test the performance of our model.

We ran predictions for the test dataset following the flowchart in Figure 3.6, this results can be seen in Figure 3.19 as a predicted vs real plot.

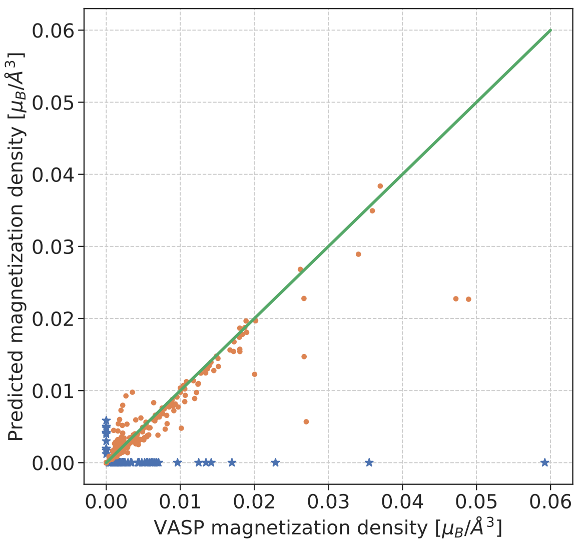


Figure 3.19: Predicted versus real plot on the test dataset of magnetization density. Blue asterisks represent missclassified materials.

This prediction has an $r^2$ value of 0.67, however, the biggest contributor to the error is the missclassification of materials as magnetic or not (see Section 3.3.1), by removing this errors, 78 materials out of 2510 (blue asterisks in Figure 3.19), we obtain $r^2 = 0.89$, a significant improvement, showcasing the importance of improving the performance of material classification.

In conclusion, most of the materials have a reasonable magnetization, compared to VASP calculations. However, 3% of them are missclassified, having a high contribution to the error metrics. Furthermore, we have demonstrated that predictions of local properties able to reconstruct global properties in the crystal, even though the models are trained to predict local properties of a site and have no knowledge about the crystal's structure.

## 3.4 Predicted data

In this section we apply the previously defined model to the organic materials in the Crystallographic Open Database (COD, http://www.crystallography.net/cod), obtaining results for 196, 471 materials. Of these, 35, 724 are predicted as magnetic, and in Figure 3.20 we observe the magnetization density of these materials and the VASP calculated ones from the OMDB as logarithmic histograms.

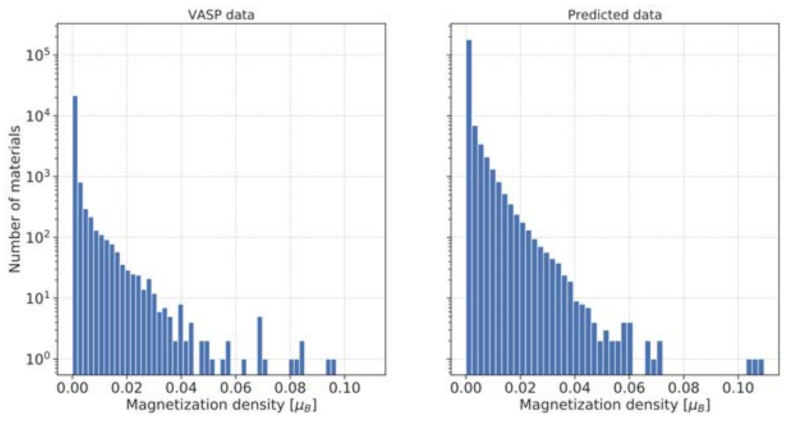


Figure 3.20: Magnetization density histogram. In the left we observe the distribution for the 23, 486 materials from VASP calculations available at the OMDB. On the right, the distribution for 196, 471 materials is displayed.

This data will be made available in the OMDB, allowing users to browse predictions as a first step to identify interesting magnetic materials.

### 3.4.1 Exchange Interaction Parameters

In order to fully characterize the Heisenberg Hamiltonian using ML, we need to predict the exchange interaction parameters, $J_{ij}$. This was not completed in the time frame of this thesis, however it is planned to be done in the near future. In this section we will present a short summary of some of the anticipated difficulties associated to this problem and possible approaches to solve them.

We have shown how to represent crystals and sites, however in order to predict $J_{ij}$ we need to represent two sites and their interaction. The simplest solution would be to represent both sites using the Radial Multi-Hot representation, seen in Section 2.2.2.3, and include the distance between them as an extra variable. This representation has some pitfalls:

- It is not invariant with respect to the permutation of the sites. There is no obvious condition available to order the two sites.
- We lack information about what other atoms lie between the two sites.

In order to address the second point we will suggest considering three objects: two environments representing the sites, and one representing the connections between them, as sketched in Figure 3.21. In a 3D example the circles would become spheres and the rectangle a cylinder.

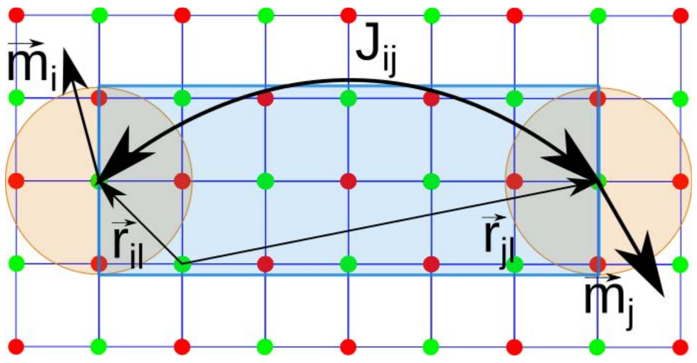


Figure 3.21: Interaction between two sites, i and j, on a 2D square lattice. Site environments are represented as orange circles and the space between the environments is represented as a blue rectangle.

The site's environments can be represented with Radial Multi-Hot representation vectors, $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$. Then we might define the two connection vectors, in a Multi-Hot way, according to chemical species:

$$V^{\text{conn}}_{i,s_l} = \sum_l \frac{1}{|\boldsymbol{r}_{il}|}\delta_{i,s_l},$$

where $S_l$ identifies one chemical specie, $l$ iterates over all the atoms in the cylinder of such species and $i = 1, 2$ identifies one of the central sites.

The aforementioned ordering issue can now be be solved by duplicating the training samples so that we have two input vectors, one for each possible ordering, associated with one target value:

$$\begin{aligned} \boldsymbol{X}_1 &= [\boldsymbol{V}_1 \boldsymbol{V}^{\text{conn}}_1 \boldsymbol{V}_2 \boldsymbol{V}^{\text{conn}}_2], \\ \boldsymbol{X}_2 &= [\boldsymbol{V}_2 \boldsymbol{V}^{\text{conn}}_2 \boldsymbol{V}_1 \boldsymbol{V}^{\text{conn}}_1], \end{aligned} \tag{3.40}$$

both representing the same two site interaction.

Chapter 4

# Machine learning for nuclei stability

## 4.1 Introduction

In recent years, the synthesis of new superheavy elements and their confirmation has expanded the periodic table up to $Z = 118$ (Og, Oganesson) [55]. These accomplishments have revitalized scientific investigation around superheavy nuclei and their stability [56].

In this chapter we will use ML methods to estimate the lifetime of nuclei currently unattainable experimentally.

### 4.1.1 Nuclei Stability

Since the establishment of the nuclear shell model in 1949, which was awarded a Nobel prize in 1963, we have known about the concept of *magic numbers* of nucleons which give more tightly bound and stable nuclei.

The experimental magic numbers for protons are **2, 8, 20, 28, 50, 82** and **2, 8, 20, 28, 50, 82, 126** for neutrons. The shells for neutrons and protons are independent, and so are the magic numbers—that is, the nuclear shells for protons and neutrons are filled independently of each other because Pauli's exclusion principle applies separately, as they are different fermions.

The possible existence of higher magic numbers leads to the **island of stability**, a concept introduced in the 1960s [57], a region of superheavy atoms of relatively long lifetime, disconnected from the main stability region.

Another factor that influences stability is the parity in the number of protons and neutrons. An even-even nucleus usually has a higher half-life than an even-odd or odd-even which in turn has a higher half-life than an odd-odd nucleus. Take as an example the number of stable nuclei: 147 for even-even, 101 for even-odd or odd-even and 5 for odd-odd.

## 4.2 Results

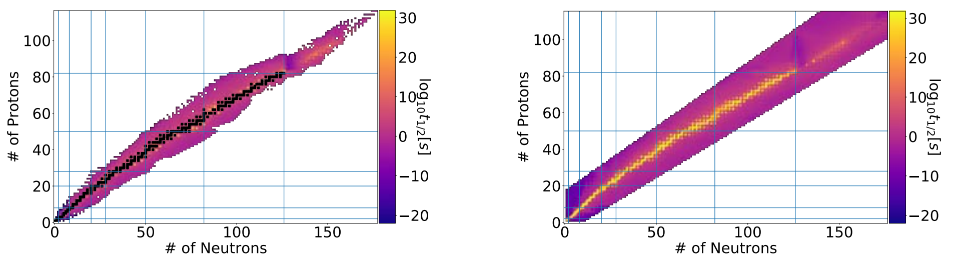
In this section we will try to estimate half-lives of atomic nuclei via ML with the objective of corroborating the existence of the island of stability using the representations from Section 2.2.3.

### 4.2.1 Magic Number Representation

We start by using the representation described in Section 2.2.3.1. We train a fully connected feedforward Artificial Neural Network (ANN) to perform regression on the order of magnitude of the half-lives of the nuclei ($\log_{10}(t_{1/2})$) obtained from the NUBASE2016 [16] dataset of experimentally measured properties of atomic nuclei, split into a training and testing dataset as usual.

Current models do not agree on predicted magic numbers for superheavy atoms, however, most of them predict them around $N \approx 172 - 184$ and $P \approx 112 - 126$ [56], so we chose $N = 178$ and $P = 120$ as the extra magic numbers for our predictions, as representative even midpoints of these ranges.

In Figure 4.1 we plot the prediction of the model for an extended region of nuclei and compare it with experimental results. This model seems to be able to predict the main stability features for currently known nuclei. For example, an increase of stability around magic numbers, and the sharp decrease of half-lives after the double magical $^{208}Pb$, as seen in Figure 4.1b.



(a) Experimental order of magnitude of half-lives. Stable atoms are denoted by black dots.

(b) ANN half-life predictions for nuclei.

Figure 4.1: Experimental versus predicted nuclei stability. The horizontal and vertical lines denote the magic numbers for protons and neutrons respectively.

This ML model obtains an $r^2$ value of $r^2 = 0.84$ and a RMSE of 3.77 orders of magnitude of half-life. Both metrics were evaluated in the test dataset.

#### 4.2.1.1 Decay Channels

Nuclei usually decay by more than one channel, and the lifetime of each channel can be measured independently. For superheavy atoms, alpha emission and spontaneous fission are the dominant decay channels, while in lighter atoms $\beta_{\pm}$ decay dominates,as seen in Figure 4.2.

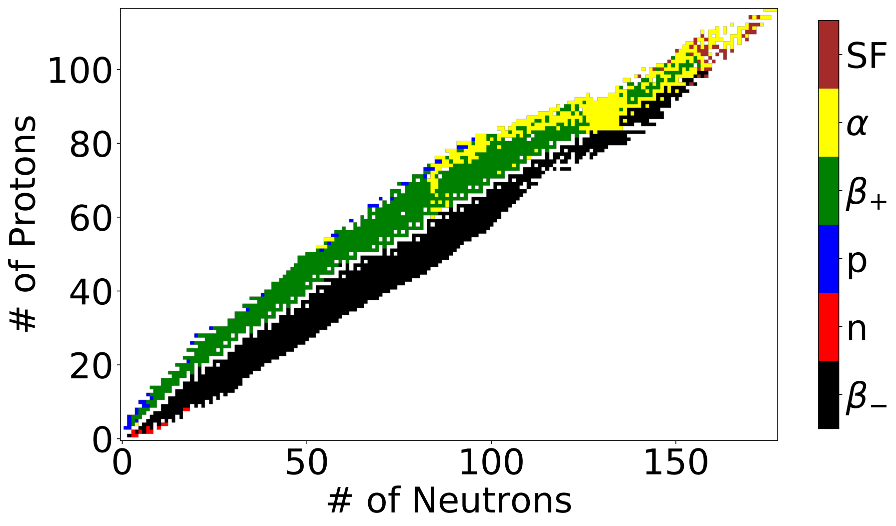


Figure 4.2: Dominant decay channels according to experiments [16]. SF is spontaneous fission; $\alpha$ represents the emission of He nucleus; $\beta_+$ and $\beta_-$ denotes the process of emiting a positron or electron respectively, transforming a proton into a neutron or vice versa; p and n are the emission of a proton or neutron respectively and occur in nuclei that are proton or neutron heavy, away from the stability diagonal. SF and $\alpha$ emission dominate in the heavier atoms.

This ANN was then trained separately for each of the main decay channels. Each model's performance on the test set can be seen in Table 4.1, with RMSE in orders of magnitude of half-life.

| **Decay Channel** | **RMSE** | $r^2$ |
|---|---|---|
| All combined | 3.77 | 0.84 |
| $\beta_+$ | 1.39 | 0.78 |
| $\beta_-$ | 2.99 | 0.44 |
| $\alpha$ | 2.54 | 0.92 |
| Spontaneous Fission | 2.80 | 0.86 |

Table 4.1: Performance metrics on the test set for the magic number representation.

We observe that the model performs better if trained in a particular decay channel than globally, which is expected, as each channel obeys different physics. Each of the decay channels' subset of data has a smaller variance than the whole dataset, that is why smaller values of RMSE correspond to similar values of $r^2$.

#### 4.2.1.2 Stability of the superheavy region

So far, all the predictions have been done in a region close to the training data (Figure 4.1), a problem known as interpolation.

However, the aim of this section is to use ML to provide predictions of lifetime in the superheavy region of atomic nuclei. ML methods have issues when performing extrapolation, defined as making predictions outside of the domain the training data is from. Extrapolation is a complex problem, subject to study from ML researchers, and has been successfully dealt with for certain problems, such as solving mathematical equations [58]. However, this is far from a solved topic in the general case.

In a NN there are multiple optimization methods [59] that can be used to obtain the optimal weight and biases for each problem. In this case, we used ADAM [39], based in stochastic gradient descent.

These methods have a degree of randomness included in them in order to avoid local minima in the parameter space. Another source of randomness is batching, a technique in which the parameters are updated each time a certain number, batch size, of training data is used to calculate the gradient, instead of using the whole set. This makes the final weights and biases sensitive to the order of the data set.

In Figure 4.3 we show the effect randomness has in a NN by training multiple NNs with the same architecture, one hidden layer with a hundred neurons and a hyperbolic tangent the as activation function, but varying the random seed. We train these NNs in data sampled from a sine function with added random noise. Due to limited range of the training data, the NNs fail to recognize the periodicity of the sine function, displaying wrong trends outside of the training domain. Furthermore, due to the randomness on the weights and biases, the predictions from the NNs start diverging outside of the training domain.

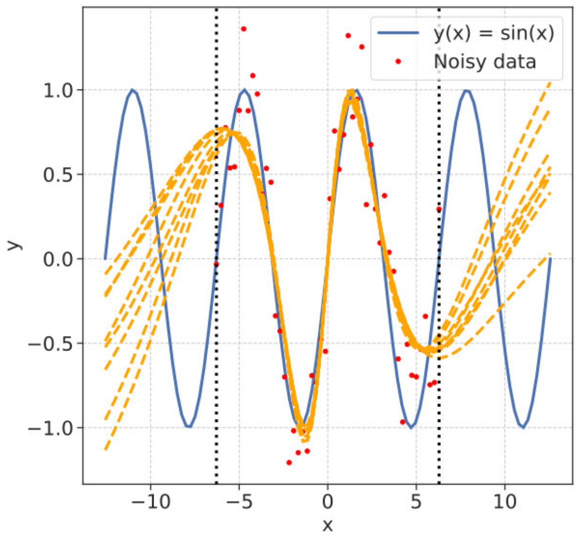


Figure 4.3: Extrapolation with a NN. The blue line represents the sine function, from which the red points were sampled with added random noise. The black dotted lines mark the boundaries of the domain in which the training samples are located. The orange lines are the predictions of NNs trained to the data initialized with different random seeds. Outside of the training region the predictions deviate considerably from the original sine function and there is a divergence between the predictions of each NN.

Coming back to the nuclear context, we will attempt to deal with the divergence of predictions outside of the training data by training 200 NNs with the same architecture, but different random seeds and different train and test dataset splits. In Figure 4.4 we display the mean, standard deviation and error of the predictions—considering only predictions on the test sets—of the 200 NN ensemble. It shows that the model performs better around the borders of our data, where all the nuclei are consistently unstable. Furthermore, our model struggles to predict the magnitude of the dip in stability after lead (Z=82)

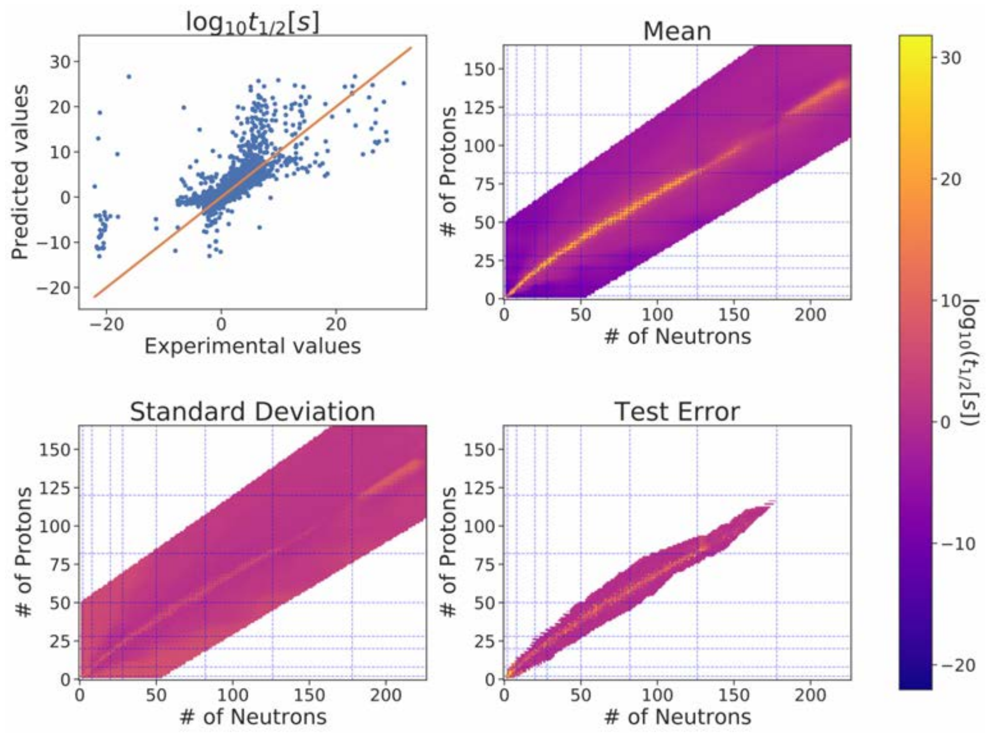


Figure 4.4: Statistical analysis of NN ensemble. A predicted versus real plot is displayed in the top left, and the rest of the figures correspond to the mean, standard deviation and error of the half-lives predicted for the test set. Blue lines represent the magic numbers.

This ensemble of NNs predicts a stability island soon after the predicted magic number for neutrons 178. However, this region also displays an increased standard deviation, meaning our models disagree on some level in their predictions, as expected in an extrapolation context. Furthermore, as shown in Figure 4.3, one has to be extremely careful when drawing conclusions outside if the training region. Nevertheless, our model did predict the stability island agreeing with the theoretical models [56].

### 4.2.2 Yukawa Representation

Continuing with the intention of avoiding extrapolation—and to use the knowledge gained from Chapter 3 on how to use ML in crystals, molecules and local enviroments—we developed a nuclear representation based in Reid's potential [37], as described in section 2.2.3.2, that identifies a nucleus as a structure similar to a molecule's, with each nucleon having a fixed position. In Figure 4.5 we plot the structures achieved by this method for two nuclei: $^{16}O$ and $^{40}Ca$.

A structure based representation has an advantage with respect to the one presented in section 2.2.3.1, as in an unseen, larger structure, the environment around each nucleon might be similar to one of a smaller nucleon present in the training data, possibly reducing extrapolation errors.

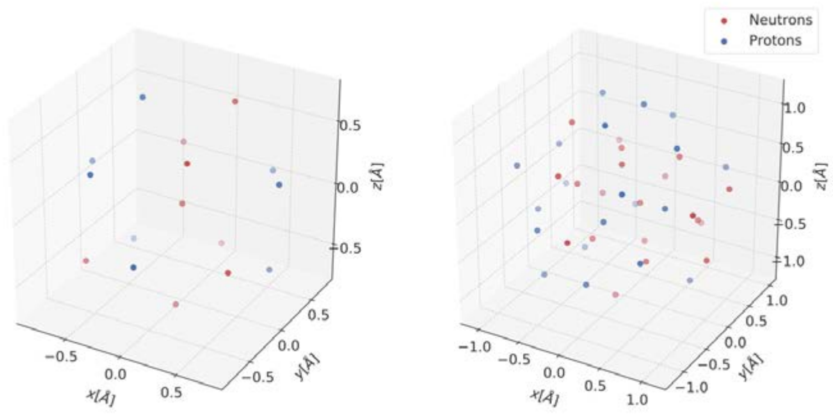


Figure 4.5: Nucleon positions for $^{16}O$ (left, P=8, N=8) and $^{40}Ca$ (right, P=20, N=20).

In order to keep computational times short, we will first restrict ourselves to nuclei with a total number of nucleons equal to or lower than 50. By plotting a histogram of nucleons and their distance from the center of mass (Figure 4.6), we realize that the nucleons are distributed in layers and that protons tend to be present in the outer layers of the nucleus, presumably due to Coulomb repulsion.

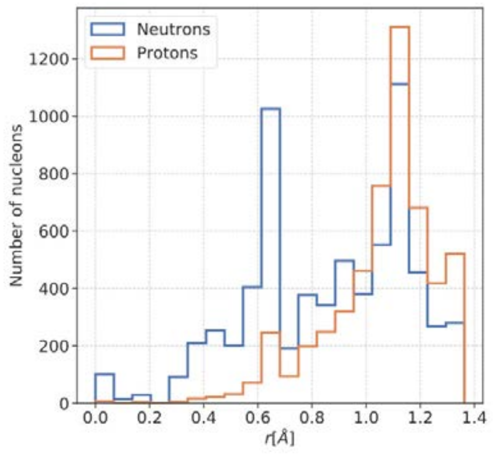


Figure 4.6: Histogram of distance of nucleons to the center of mass of the nucleus.

Once we have these structures, we can define the similarity between two nucleons $i, j$—that are either a proton or a neutron, $\alpha \in \{p, n\}$, belonging to two different nuclei $m, n$ to all other nucleons that are either a proton or a neutron $\beta \in \{p, n\}$—by considering the overlap of radial Gaussians placed on top of each nucleon:

$$A^{\alpha\beta}_{ij,mn} = \sum_{s\in\alpha_m}\sum_{t\in\beta_n}\int_0^\infty dr e^{-\frac{(r-r_{is})^2}{2\sigma^2}} e^{-\frac{(r-r_{jt})^2}{2\sigma^2}}. \tag{4.1}$$

The matrix $A^{\alpha\beta}_{mn}$ contains the similarity information between nucleus $m$ and nucleus $n$ by comparing the distances between all nucleons of type $\alpha$ and the ones of type $\beta$. In order to obtain a single number that represents the similarity between the two nuclei, we propose using the Frobenius norm:

$$K^{\alpha\beta}_{mn} = \mathrm{Tr}\left[(A^{\alpha\beta}_{mn})^\dagger A^{\alpha\beta}_{mn}\right], \tag{4.2}$$

which leaves us with four similarity kernels between nuclei: p-p, p-n, n-p and n-n.

Now we can perform Kernel Ridge Regression [60], a kernel method similar to Support Vector Machines, Section 2.3.3, but with a focus on regression and not classification, on the lifetime data.

| **Kernel** | **RMSE** | $r^2$ |
|---|---|---|
| p-p | 11.91 | 0.17 |
| n-n | 12.11 | 0.14 |
| p-n | 7.98 | 0.63 |
| n-p | 8.02 | 0.61 |

Table 4.2: Performance of the kernels in the training set. Positive values of $r^2$ indicate that the models are able to differentiate between nuclei with the information from the similarity kernel.

The p-n and n-n kernels were able to achieve some success in the training dataset. The performance metrics are displayed in Table 4.2.

However, in the test set the results were unsatisfactory, obtaining $r^2$ values smaller than 0, as seen in Table 4.3. The model therefore has no predictive capabilities and is outperformed by the scalar model (always predicting a mean value).

| **Kernel** | **RMSE** | $r^2$ |
|---|---|---|
| p-p | 10.39 | -0.04 |
| n-n | 10.32 | -0.02 |
| p-n | 10.58 | -0.08 |
| n-p | 10.44 | -0.05 |

Table 4.3: Performance of the kernels in the test set. Negative values of $r^2$ indicate that our model is unable to make meaningful predictions in previously unseen samples.

Significantly superior performance on the training set than on the test set is a clear indicator of overfitting, as described in Section 2.1, signaling that our model is not able to detect real trends in data and instead memorizes the training set.

Two other kernels, one comprised of the sum of all the kernels and the other one the sum of p-n and n-p kernels, were also tried as a way of including all the data into one model. However, this didn't improve the metrics.

Further investigation of how ML methods can be used to predict nuclear stability and detect stability island would take a more focused effort. Here we summarized the approach and results we obtained to this date.

Chapter 5

# Conclusions

In this thesis I have deployed state of the art machine learning tools for investigations of research questions arising in modern physics. ML proves to be statistical tool with great potential, due to its ability to analyze data and obtain new insights and predictions. In Chapter 2, we introduced many statistical concepts, useful not only for creating ML models, but also to give accurate measurements of their performance in realistic scenarios, and their ability to address the challenges of unseen data.

Then, in Chapters 3 and 4, we applied these concepts to two different problems: prediction of magnetic characteristics and nuclei lifetime estimation.

The aim of the study of nuclear stability was to corroborate the existence of the island of stability, already predicted by nuclear physics models [56], with novel ML techniques. This was achieved by using a ML model with information about the number of nucleons and the location of nuclear magic numbers [12], however, this ML model is not well suited for extrapolation. It was in this context that we introduced another ML model—using the experimentally based Reid's potential [37]—less susceptible to extrapolation issues. Nonetheless, this model had no predictive capabilities outside of its training data. ML is a potent and popular tool, however, it is important to recognize its limitations and extrapolation is still a challenge [58].

In Chapter 3 we first used Linear Spin Wave Theory [52, 51] to understand magnetic excitations in crystals, applying them to honeycomb magnets. Then, we used the OMDB [8] data to construct a ML model able to identify magnetic properties and with the aim of parametrizing magnetic Hamiltonians of organic crystals. This model showed the capabilities of ML for predicting the magnetic properties of crystals, obtaining accurate results when classifying a crystal as magnetic or non-magnetic. We then introduced ML as a tool to analyze local properties of crystals, including the creation of several site representations. Of those, Radial Multi-Hot—a representation developed by us—achieves the best performance, both when classifying magnetism in sites and when performing regression for their magnetization. Moreover, these local property predictions are able to reconstruct global properties successfully, such as magnetization density. This model was then applied to around $200,000$ organic crystals—previously synthesized and present in the Crystallographic Open Database [54]—and this predicted dataset will be soon available publicly in the OMDB. Finally, we proposed a possible representation for pairwise site interactions, such as Heisenberg exchange parameters, paving the way for ML to be used for modelling and prediction of magnetic ground states and excitations.

**Acknowledgements**

I would like to thank Matthias Geilhufe and Johan Hellsvik for their guidance and patience during this thesis. They provided invaluable support and always welcomed my inquiries. This thesis would not have been the same without the multiple ML discussions with Bart Olsthoorn, Kirsty Dunnett's exhaustive review of my writing, and Prof. Eva Lindroth's role as a link between SU and Nordita. I would also like to thank all the members of the Condensed Matter group of Nordita, and specifically to Prof. Alexander Balatsky for offering me the opportunity to work in his group.

# Bibliography

[1] K. P. Murphy, *Machine Learning: A Probabilistic Perspective.* MIT press, 2012.

[2] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM J. Res. Dev.*, vol. 3, no. 3, pp. 210–229, Jul. 1959.

[3] F. Rosenblatt, *The Perceptron, a Perceiving and Recognizing Automaton Project Para.* Cornell Aeronautical Laboratory, 1957.

[4] G. Cybenko, "Approximations by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, pp. 183–192, 1989.

[5] D. Guest, K. Cranmer, and D. Whiteson, "Deep learning and its application to LHC physics," *Annu. Rev. Nucl. Part. Sci.*, vol. 68, pp. 161–181, 2018.

[6] S. Das Sarma, D.-L. Deng, and L.-M. Duan, "Machine learning meets quantum physics," *Physics Today*, vol. 72, no. 3, pp. 48–54, Mar. 2019.

[7] S. P. Ong, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, D. Bailey, D. Skinner, K. Persson, and G. Ceder, *The Materials Project.* Lawrence Berkeley National Laboratory: Berkeley, CA, 2011.

[8] S. S. Borysov, R. M. Geilhufe, and A. V. Balatsky, "Organic materials database: An open-access online database for data mining," *PloS One*, vol. 12, no. 2, p. e0171501, 2017.

[9] S. S. Borysov, B. Olsthoorn, M. B. Gedik, R. M. Geilhufe, and A. V. Balatsky, "Online search tool for graphical patterns in electronic band structures," *Npj Comput. Mater.*, vol. 4, no. 1, p. 46, Aug. 2018.

[10] B. Olsthoorn, R. M. Geilhufe, S. S. Borysov, and A. V. Balatsky, "Band Gap Prediction for Large Organic Crystal Structures with Machine Learning," *Adv. Quantum Technol.*, vol. 0, no. 0, p. 1900023.

[11] F. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento, "Crystal structure representations for machine learning models of formation energies," *Int. J. Quantum Chem.*, vol. 115, no. 16, pp. 1094–1101, 2015.

[12] O. Sorlin and M.-G. Porquet, "Nuclear magic numbers: New features far from stability," *Prog. Part. Nucl. Phys.*, vol. 61, no. 2, pp. 602–673, 2008.

[13] Crookes William, "V. The Bakerian Lecture.—On the illumination of lines of molecular pressure, and the trajectory of molecules," *Philosophical Transactions of the Royal Society of London*, vol. 170, pp. 135–164, Jan. 1879.

[14] W. C. Röntgen, "ON A NEW KIND OF RAYS," *Science*, vol. 3, no. 59, pp. 227–231, Feb. 1896.

[15] X. Yu, A. R. Oganov, Q. Zhu, F. Qi, and G. Qian, "The stability and unexpected chemistry of oxide clusters," *Phys. Chem. Chem. Phys.*, vol. 20, no. 48, pp. 30 437–30 444, 2018.

[16] G. Audi, F. G. Kondev, M. Wang, W. Huang, and S. Naimi, "The NUBASE2016 evaluation of nuclear properties," *Chin. Phys. C*, vol. 41, no. 3, p. 030001, Mar. 2017.

[17] P. A. M. Dirac, "The quantum theory of the electron," *Proc. R. Soc. Lond. Ser. Contain. Pap. Math. Phys. Character*, vol. 117, no. 778, pp. 610–624, 1928.

[18] T. O. Wehling, A. M. Black-Schaffer, and A. V. Balatsky, "Dirac materials," *Adv. Phys.*, vol. 63, no. 1, pp. 1–76, 2014.

[19] S. S. Pershoguba, S. Banerjee, J. C. Lashley, J. Park, H. Ågren, G. Aeppli, and A. V. Balatsky, "Dirac Magnons in Honeycomb Ferromagnets," *Phys. Rev. X*, vol. 8, no. 1, p. 011010, Jan. 2018.

[20] R. M. Geilhufe, S. S. Borysov, A. Bouhon, and A. V. Balatsky, "Data mining for three-dimensional organic Dirac materials: Focus on space group 19," *Sci. Rep.*, vol. 7, no. 1, p. 7298, 2017.

[21] J. Hellsvik, R. D. Pérez, R. M. Geilhufe, M. Månsson, and A. V. Balatsky, "Spin wave excitations of magnetic metalorganic materials," *ArXiv190701817 Cond-Mat*, Jul. 2019. [Online]. Available: http://arxiv.org/abs/1907.01817

[22] J. Tuček, K. Holá, A. B. Bourlinos, P. Błoński, A. Bakandritsos, J. Ugolotti, M. Dubecký, F. Karlický, V. Ranc, K. Čépe, M. Otyepka, and R. Zbořil, "Room temperature organic magnets derived from $sp^3$ functionalized graphene," *Nat. Commun.*, vol. 8, p. 14525, Feb. 2017.

[23] B. Olsthoorn, R. M. Geilhufe, S. S. Borysov, and A. V. Balatsky, "Band gap prediction for large organic crystal structures with machine

learning," *ArXiv181012814 Cond-Mat Physicsphysics Stat*, Oct. 2018. [Online]. Available: http://arxiv.org/abs/1810.12814

[24] D. J. Hand, "Measurements and Their Uncertainties: A Practical Guide to Modern Error Analysis by Ifan G. Hughes, Thomas P. A. Hase," *Int Stat Rev*, vol. 79, no. 2, pp. 280–280, 2011.

[25] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Phys. Rev. Lett.*, vol. 108, no. 5, p. 058301, 2012.

[26] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, "SchNet – A deep learning architecture for molecules and materials," *J. Chem. Phys.*, vol. 148, no. 24, p. 241722, 2018.

[27] S. De, A. P. Bartók, G. Csányi, and M. Ceriotti, "Comparing molecules and solids across structural and alchemical space," *Phys. Chem. Chem. Phys.*, vol. 18, no. 20, pp. 13 754–13 769, 2016.

[28] A. P. Bartók, S. De, C. Poelking, N. Bernstein, J. R. Kermode, G. Csányi, and M. Ceriotti, "Machine learning unifies the modeling of materials and molecules," *Sci. Adv.*, vol. 3, no. 12, p. e1701816, 2017.

[29] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld, "Machine learning of molecular electronic properties in chemical compound space," *New J. Phys.*, vol. 15, no. 9, p. 095003, 2013.

[30] L. Himanen, M. O. J. Jäger, E. V. Morooka, F. Federici Canova, Y. S. Ranawat, D. Z. Gao, P. Rinke, and A. S. Foster, "DScribe: Library of Descriptors for Machine Learning in Materials Science," *ArXiv E-Prints*, p. arXiv:1904.08875, Apr. 2019.

[31] M. O. J. Jäger, E. V. Morooka, F. F. Canova, L. Himanen, and A. S. Foster, "Machine learning hydrogen adsorption on nanoclusters through structural descriptors," *npj Comput Mater*, vol. 4, no. 1, pp. 1–8, Jul. 2018.

[32] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dulak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. B. Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, and K. W. Jacobsen, "The atomic simulation environment—a Python library for working with atoms," *J. Phys.: Condens. Matter*, vol. 29, no. 27, p. 273002, Jun. 2017.

[33] L. E. Sutton, *Tables of Interatomic Distances and Configuration in Molecules and Ions: Supplement 1956-59*. Chemical Society, 1965, no. 18.

[34] N. J. Costiris, E. Mavrommatis, K. A. Gernoth, and J. W. Clark, “Decoding beta-decay systematics: A global statistical model for beta- half-lives,” *Phys. Rev. C*, vol. 80, no. 4, p. 044332, 2009.

[35] Z. M. Niu, H. Z. Liang, B. H. Sun, W. H. Long, and Y. F. Niu, “Predictions of nuclear $\beta$-decay half-lives with machine learning and their impacts on r process,” Oct. 2018.

[36] H. Yukawa, “On the interaction of elementary particles. I,” *Proc. Phys.-Math. Soc. Jpn. 3rd Ser.*, vol. 17, pp. 48–57, 1935.

[37] R. V. Reid, “Local phenomenological nucleon-nucleon potentials,” *Annals of Physics*, vol. 50, no. 3, pp. 411–448, Dec. 1968.

[38] D. J. Wales and J. P. Doye, “Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms,” *J. Phys. Chem. A*, vol. 101, no. 28, pp. 5111–5116, 1997.

[39] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *ArXiv14126980 Cs*, Dec. 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[40] Mcstrother, “Multilayer Feed Forward Artificial Neural Network, CC,” 2010. [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/7/7f/Two_layer_ann.svg

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[42] L. Rokach and O. Z. Maimon, *Data Mining with Decision Trees: Theory and Applications*. World scientific, 2008, vol. 69.

[43] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach Learn*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[44] Larhmam, “SVM margin, CC,” 2018. [Online]. Available: https://commons.wikimedia.org/wiki/File:Two_layer_ann.svg

[45] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan, “A dual coordinate descent method for large-scale linear SVM,” in *Proceedings of the 25th International Conference on Machine Learning*. ACM, 2008, pp. 408–415.

[46] P. Flach, “The Geometry of ROC Space: Understanding Machine Learning Metrics through ROC Isometrics,” in *Proceedings of ICML03*, vol. 1, Jan. 2003, pp. 194–201.

[47] K. Knöpfle, L. M. Sandratskii, and J. Kübler, “Spin spiral ground state of $\gamma$-iron,” *Phys. Rev. B*, vol. 62, no. 9, p. 5564, 2000.

[48] F. Bloch, “Zur theorie des ferromagnetismus,” *Z. Für Phys.*, vol. 61, no. 3-4, pp. 206–219, 1930.

[49] J. C. Slater, “Cohesion in monovalent metals,” *Phys. Rev.*, vol. 35, no. 5, p. 509, 1930.

[50] T. Holstein and H. Primakoff, “Field dependence of the intrinsic domain magnetization of a ferromagnet,” *Phys. Rev.*, vol. 58, no. 12, p. 1098, 1940.

[51] J. T. Haraldsen and R. S. Fishman, “Spin rotation technique for non-collinear magnetic systems: Application to the generalized Villain model,” *J. Phys. Condens. Matter*, vol. 21, no. 21, p. 216001, 2009.

[52] S. Toth and B. Lake, “Linear spin wave theory for single-Q incommensurate magnetic structures,” *J. Phys.: Condens. Matter*, vol. 27, no. 16, p. 166002, 2015.

[53] J. H. P. Colpa, “Diagonalization of the quadratic boson Hamiltonian,” *Phys. Stat. Mech. Its Appl.*, vol. 93, no. 3-4, pp. 327–353, 1978.

[54] S. Gražulis, D. Chateigner, R. T. Downs, A. F. T. Yokochi, M. Quirós, L. Lutterotti, E. Manakova, J. Butkus, P. Moeck, and A. Le Bail, “Crystallography Open Database–an open-access collection of crystal structures,” *J. Appl. Crystallogr.*, vol. 42, no. 4, pp. 726–729, 2009.

[55] P. J. Karol, R. C. Barber, B. M. Sherrill, E. Vardaci, and T. Yamazaki, “Discovery of the element with atomic number Z = 118 completing the 7th row of the periodic table (IUPAC Technical Report),” *Pure Appl. Chem.*, vol. 88, no. 1-2, pp. 155–160, 2016.

[56] S. A. Giuliani, Z. Matheson, W. Nazarewicz, E. Olsen, P.-G. Reinhard, J. Sadhukhan, B. Schuetrumpf, N. Schunck, and P. Schwerdtfeger, “Colloquium: Superheavy elements: Oganesson and beyond,” *Rev. Mod. Phys.*, vol. 91, no. 1, p. 011001, 2019.

[57] W. D. Myers and W. J. Swiatecki, “Nuclear masses and deformations,” *Nucl. Phys.*, vol. 81, no. 1, pp. 1–60, 1966.

[58] S. S. Sahoo, C. H. Lampert, and G. Martius, “Learning Equations for Extrapolation and Control,” *ArXiv180607259 Cs Stat*, Jun. 2018. [Online]. Available: http://arxiv.org/abs/1806.07259

[59] S. Ruder, “An overview of gradient descent optimization algorithms,” *ArXiv Prepr. ArXiv160904747*, 2016.

[60] M. Welling, “Kernel ridge regression,” *Max Wellings Classnotes Mach. Learn.*, pp. 1–3, 2013.