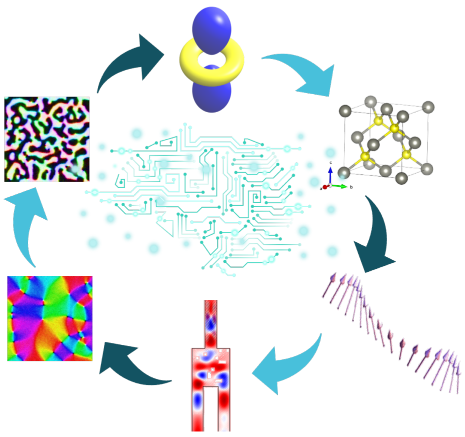
Università degli Studi di Torino

Doctoral School of the University of Torino

PhD Programme in Chemical and Materials Sciences XXXV Cycle

The Versatility of Machine Learning Techniques in Materials Science: A Multiscale Study



Udaykumar Gajera

******

Supervisor(s):

Dr. Silvia Picozzi

Prof. Paola Rizzi

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Udaykumar Gajera
2023

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Università degli Studi di Torino.

I would like to wholeheartedly dedicate this thesis to my caring parents, my cherished family, and all of my supportive friends and colleagues, without whom this incredible journey would not have been possible.

Acknowledgements

Time truly flies, doesn't it? Just three years ago, I arrived in Chieti, brimming with excitement for a new adventure. Little did I know, I'd soon face a Covid-19 lockdown, with each day bringing new restrictions. Despite language barriers, I was fortunate to have a supportive network of supervisors and peers to guide me. Fast forward to today, and I've experienced highs and lows, from exploring new places and attending conferences to enduring sleepless nights and the tedium of paper writing.

I'm profoundly grateful for the guidance and patience of my supervisor, Dr. Silvia Picozzi. Professors Sebastiaan van Dijken and Loriano Storchi also deserve recognition for their unwavering support. My advisory committee's expertise was invaluable, and the European Union's Horizon 2020 research and innovation program's Marie Skłodowska-Curie grant agreement no. 861145 provided crucial funding.

I extend my thanks to Drs. Amoroso, Delodovici, and Talapatra for their early support; friends Srdjan, Baishun, and Lei for their editing and critiquing; and the research assistants and study participants at the university. My family—including my parents, sister Disha, brother-in-law Hardik, and spouse Anjali—kept me motivated throughout this journey. I'm also grateful for the initial research assistance from the Max-Planck Institute's Kishan, Murali, Poulumi, Biswanath, Ankit, Omkar Lekshmi, and Dr. Hickel, as well as the support of friends Akarsh, Alessio, Prabin, Roberto, Maurine, Anna, Zubair, Nikolai, Weija, Huajun, Vairag, and Mital.

In conclusion, I'm deeply appreciative of everyone who contributed to this journey. Your encouragement and support have made all the difference. Through my PhD thesis, I hope to positively impact the scientific community and advance knowledge in our field. Thank you for believing in me and helping me turn my dreams into reality.

Abstract

Nowadays machine learning (ML) tools are severely impacting many fields in science and technology, due to their flexibility and overall reduced computational cost. This thesis explores the application of Machine Learning (ML) techniques in materials science across multiple length scales, focusing on three different topics related to various condensed matter subfields.

The first example focuses on predicting the energetic stability of binary semiconductors in the Zincblende and Rocksalt crystal structures. The proposed approach is based on simple and accessible atomic properties used to build complex descriptors employed in linear regression models. By using a dataset obtained via first-principles simulations, our study shows that one-dimensional formulas can efficiently describe the energetics of binary semiconductors. The approach also highlights the importance of various atomic properties in driving the stabilization of one crystal structure over the other, with spatial atomic properties playing a significant role.

This thesis also focuses on the design of an energy-efficient magnonic device that functions as a physical neural network. The proposed model uses spin waves as primary information carrier, allowing data processing with minimal energy dissipation. The research builds upon a benchmark problem from the literature, where the separation of two spin wave frequencies using a magnonic device was successfully simulated. In our case, we first performed micromagnetic simulations on a prototype device having a rectangular ferromagnetic area with square-shaped defects. Genetic Algorithms (GA) were used to optimize the position of defects by minimizing a defined cost function (*i.e.* maximizing frequency separation). Our results proved GA to be an extremely efficient approach. After successfully optimizing this kind of demultiplexer, other devices were simulated, such as classifiers of alphabet letter and numbers.

In addition, this study also explores image regression algorithms based on neural networks, to understand changes in magnetic properties (especially perpendicular magnetic anisotropy) resulting from external perturbations on a Co/Pd multilayer system. Our research was inspired by an experimental investigation focused on magnetic and structural modifications obtained upon ion-beam-irradiation. After validating the results using micromagnetic simulations, we generated about a thousand simulated domain images, to compensate for the lack of a significant number of experimental images, which were used to train and test a convolutional neural network (CNN). Our approach was shown to predict magnetic properties with high accuracy starting from domain images and demonstrate the potential of image regression methods based on CNN for understanding magnetic properties driven by microscopic features.

Overall, this Ph.D. thesis highlights the versatility and usefulness of ML techniques in materials science at different scales, ranging from predicting energetic stability to designing energy-efficient magnonic devices to understanding magnetic properties from domain patterns. These findings pave the way for further research aimed at developing AI tools for materials design and computing technologies.

Contents

List of Figures **x**

List of Tables **xviii**

Nomenclature **xx**

1 Introduction to Machine Learning in Materials Science: Exploring Multiple Scales from Atomic to Macroscopic **1**

2 Role of Atomic Properties in the Crystal Structure Stabilization: insights from ML **7**

2.1 Introduction 7

2.2 Methodology 10

2.2.1 Dataset preparation and materials 11

2.2.2 Formulas construction 12

2.2.3 Formulas optimization 13

2.2.4 Higher-dimensional features 15

2.2.5 Test of predictive power of ΔE formula for AB compounds outside the dataset 16

2.2.6 Validation of the linear regression 17

2.3 Results 18

2.3.1 Results from Formula Search 18

2.3.2 Effect of atomic radii on ΔE 19
2.3.3 Proof of concept using alloy systems 21
2.3.4 Higher dimensional features 24
2.3.5 Heat map for 1D, 2D and 3D formulas 26
2.3.6 Comparison with the reference 1D Formula 29
2.3.7 Formula optimization using automated optimization methods 30
2.3.8 Comparison with deep neural network 34
2.4 Summary 36

3 Optimizing the Design and Performance of Magnonic Devices through Modelling and Simulation Techniques Enhanced by Machine Learning Algorithms 38

3.1 Introduction 38
3.2 Methodology and Technical details 42
3.2.1 MuMax3 code 42
3.2.2 Micromagnetic simulations of demultiplexer 44
3.3 Results and Discussion of different ML methods 44
3.3.1 Binary Search Method 44
3.3.2 Encoder-Decoder Sequence Model (EDSM) 48
3.3.3 Genetic Algorithm 54
3.4 Additional GA-optimized magnonic devices for complex tasks . . . 59
3.4.1 Demultiplexer for three frequencies 59
3.4.2 Alphabet Identifier 60
3.4.3 Digit Identifier 62
3.5 Summary 65

4 Understanding the Modification of Magnetic Domains Using Machine Learning 67

4.1 Introduction 67
4.2 Methodology 69
4.2.1 Micromagnetic Simulations 69
4.2.2 Simulated Domain Configurations 70
4.2.3 Overview of the Convolutional Neural Network 73
4.2.4 Addition of Convolutional layers to the Neural Network . . 74
4.2.5 Building and Training of CNN 77
4.3 Results and Discussion 84
4.4 Test on experimental MFM image: discussion and summary 87

5 Conclusion and Outlook 89

References 92

Appendix A Additional Information 109
.1 Analysis of the best 1D, 2D and 3D formulas 109
.2 Script for the alphabet identifier 112
.3 Understanding Domain Modification Using Machine Learning . . . 115
.3.1 Atomic and Magnetic Force Microscopy 115
.3.2 Mumax3 script 116
.3.3 ResNet Models 117

List of Figures

2.1 Different components used in machine learning methods and detailed for the Linear regression. The steps include: Dataset preparation, Selection of machine learning method and Validation. 10

2.2 a) Basic atomic properties (APs) used to construct the material features. b) Crystal structures of RS and ZB (plot made using the VESTA tool) [1]. Grey (yellow) spheres represent A (B) atoms. c) Workflow for formulas construction, machine learning methodology, validation, and MF selection procedures. In the AB compounds, A is the atom with the lowest electronegativity. 11

2.3 Energy difference between rocksalt and zincblende, ΔE (in eV), as a function of $r_p(A)$ for different binary compounds (blue triangles). Data fit functions are also shown, using proportionality to $r_p(A)^{-2}$ and $r_p(A)^{-3}$ (cfr: $f(r_p(A))$) via green dashed line and red straight line, respectively in the main graph and insets. Insert shows the linear fit of ΔE for $f(r_p(A)) \in [0:1]$, as a function of $1/r_p(A)^2$ (green triangles) and $1/r_p(A)^3$ (red dots). Resulting slope (m) and intercept (q) are respectively: 1.120 (eV Å^2), -0.360 (eV) for $1/r_p(A)^2$, and 1.184 (eV Å^3), -0.240 (eV) for $1/r_p(A)^3$. 22

2.4 Dependence of ΔE on atomic features: a) r_s, b) r_d, c) EA, d) $HOMO$, e) $LUMO$ and f) IP. Orange dots (blue triangles) indicate values relative to the B (A) atoms. In panel-**a**, we perform a fit using a function $f(x)$ proportional to x^{-2} (dotted green line) and to x^{-3} (straight red line). 23

2.5 Total energy difference ΔE as a function of concentration(x) for $[Ca_xSr_{1-x}]Se$, $[Mg_xCa_{1-x}]Se$ and $[Be_xMg_{1-x}]Se$ alloys, highlighted in blue, green, and pink regions respectively. Energy differences are predicted using original and optimized 1D descriptors constructed using $GEN3$ and $GEN4$ and verified using DFT (black line with diamond points) within VCA. For an improved accuracy, the two asterisk-highlighted intermediate points in the $[Mg_xCa_{1-x}]Se$ region are calculated using the supercell approach rather than VCA. 25

2.6 Total energy difference ΔE as a function of concentration (x) for $Sr[S_xSe_{1-x}]$, see panel **a)**, and $B[P_xSb_{1-x}]$ alloys, see panel **b)**, predicted from original and optimized 1D descriptors constructed using $GEN3$ and $GEN4$. Model predictions are verified using energy differences calculated via DFT[2][3] (black-line with diamond points). 25

2.7 Comparison of actual (*i.e.* DFT) vs predicted total energy difference ΔE for **a)** 1D, **c)** 2D and **d)** 3D formulas constructed using $GEN3$. Panel **b)** shows the best 1D descriptors after formula optimization. Lower-right insets show a zoom in the relevant region where many compounds are concentrated. Red dotted lines correspond to $2\times avg(RMSE)$ value. The respective descriptors can be inferred from tables-2.1, 2.2, 2.4, 2.5 28

2.8 Nomenclature used in the heatmaps, taking as example a 2D formula constructed using two 1D formulas and Gen-2. Here, G represents the generator number followed by two numbers (1, 2 or 3): GIJ refers to the Jth 1D-formula used by Generator I when constructing higher dimensional formula. The complete formula can be then denoted as 2DG2 (*i.e.* 2D formula using generator 2). 29

2.9 Heatmap of Pearson correlation coefficient between different 1D (**a**) and elements of 2D (**b**) formulas, using different generators. (**c**) shows the colour bar for the heatmap. The nomenclature reported on the axes labels follows Fig-2.8. The red square inside panel (**b**) shows the correlation between 1D formulas inside each 2D formula. R1 represents the 1D formula obtained from the reference paper-[4]. 30

2.10 Heatmap of Pearson correlation coefficient between different elements in 3D formulas using different generators . The nomenclature reported on the axes labels follows Fig-2.8. The red squares show the correlation between 1D formulas inside each 3D formula. 31

2.11 Heatmap of Pearson correlation coefficient between different 1D, 2D and 3D complete formulas. The nomenclature for the axis label follows Fig-2.8. 1DR, 2DR and 3DR denote the 1D, 2D and 3D formulas as published in the reference-paper [4], respectively. 32

2.12 Panel **a** reports the predicted against actual (*i.e* DFT) ΔE values for 1D formula presented in Ref. 4; panel **b** reports those obtained by $GEN3$ as a comparison. Absolute error for the same formula for different compounds in the bar graph (panels **c** and **d**). The related formulas used to calculate the values can be inferred from the main text. 33

2.13 Evolution of different parameters at each minimization step (i.e. iteration) using different automated optimizing algorithms: Nelder-Mead (Blue lines), CG (orange lines), BFGS (green lines) and TNC (red lines) as shown in **f**. Here, we show the evolution of RMSE (eV), $a/b, c/d, m \times a/c$ and $m \times b/d$ in panels **a, b, c, d** and **e** respectively. Here, to show the evolution of different parameters, we use a GEN3 formula $\big((r_p(B) - \sqrt{|r_d(A)|})/(r_p(B)^3 + r_p(A)^3)\big)$. The ratios for grid search are -1.28, 1.00, 0.88, -0.67 for a/b, c/d, $m \times a/c$ and $m \times b/d$ respectively. 35

2.14 Conceptual visualization of approaches based on Neural Network (top) or Formula search method (bottom). 36

3.1 Evolution of logic devices from Vacuum tubes (a) to integrated circuits (c) to high efficiency processors (e). The figure is reproduced using the ref-[5]. 39

3.2 a) Structure for the design of the magnonic frequency demultiplexer suggested by Wang et al.[6]. b) Complete structure of the demultiplexer device. C) Visualization of spin waves inside the demultiplexer. 42

3.3 Simulations of different stripes of YIG to calculate the wavelengths associated to different frequencies. The size of the stripes is $10\mu m \times 0.4\mu m$. The vertical blue line in each of the stripes indicates the place of the antenna that we have used to excite the spin waves. The different frequencies are shown in the legend. Red and blue regions show the spin wave amplitude (high and low) in the Z direction. Here, panel a) shows the spin waves for different frequencies and panel b). shows the spin wave amplitude in the central section of the stripes for different frequencies. 45

3.4 (a) Algorithm of the Binary search method published in the paper-[6]; (b) schematic explanation, using different structures. The detailed explanation is provided in Section-3.3.1 46

3.5 Schematic difference between the CNN (right) and RNN (left) types of Networks. The figure shows the main difference between the two types of networks: RNN feeds the results back into the network, whereas CNN uses feed-forward architecture types. 49

3.6 Schematic difference between the BS (left) and EDSM (right) types of Networks. We can see that the BS will start from the initial structure and optimize from the first one. On the other hand, EDSM needs to be trained before generating results. Other differences are given in the text. The panel (b) is constructed using the ref-[7]. . . . 50

3.7 Histogram of relative intensity flowing through each arm. The blue bars indicate the 2.6GHz flowing in output guide 1 and orange bars indicate the 2.8GHz flows into the output guide 2 for any particular configurations. The Y-axis indicates the number of sample (configurations) resulting in the intensity. 51

3.8 Graphical representation of Many to Many sequence model. Here, (x_1, x_2, x_n), (y_1, y_2, y_n) represent the input and output vector respectively. $(h_{1,t}, h_{2,t}, h_{n,t})$, $(s_{1,t} s_{2,t}, s_{n,t})$ represent the hidden state vector for the time t for the encoder and decoder respectively. $C_{n,t}$ represents the objective function we try to predict at time t. 52

3.9 Examples of the configuration that we have used to train the Encoder part. On every configuration, the top line suggests the value of the intensity of f_1 in the output guide 1, similarly for the output guide 2. The yellow color shows the YIG material, whereas the black color indicates the vacuum. 52

3.10 Left (right) panel: results from the encoder (decoder) network. Here, we give the sequence of the configuration to the encoder to predict the intensity in two output guides. These intensities will transmit to the decoder network to predict the input configuration. Here "O" represent value of objective function3.1 on X and Y axis. In addition to that, accurate and prediction O in the encoder suggest the Mumax3 calculation and machine learning output for the provided sequences respectively. However, on the decoder side, accurate and prediction O suggest the Mumax3 simulations of actual sequence and predicted sequence from EDSM 53

3.11 Graphical representation of the concept of Genetic algorithm in nature (a) and its use to optimize the configuration (b). The panel a) was taken from the ref-[8] 54

3.12 Graphical representation of population generation. 55

3.13 An example of single point crossover and its new off springs. 55

3.14 An example of Mutation. The panel on the left shows the unmutated individual and the panel on the right shows the mutated version (see numbers in red). 56

3.15 Three types of devices. **a)** demultiplexer submitted in the reference article [6]. **b)** demultiplexer achieved by the GA. **c)** attenuator (reducing amplitude) created using the GA. The graphs on the right column show, for each device, the amplitude at different lengths. Here, we plot the green line for $2.6 + 2.8GHz$ at the input waveguide, whereas the orange and blue lines show the spin-wave signals in the two output waveguides. Moreover, Figure-**i,ii,iii** shows the amplitude of spinwave at the horizontal cross section of device for **a,b,c** respectively. 58

3.16 (a) Demultiplexer with one input and three output. (b) Amplitude of the device at the input (blue line) and output waveguide (orange and green lines). The red and blue colors inside the device represent the total magnetization in Z direction (positive and negative Z shown in red and blue, respectively). 61

3.17 Representation of 12 alphabet characters by a 3×3 matrix. 62

3.18 Workflow of the alphabet identifier. Panel a) shows three letters: L, N, X. Panel b) shows the binary conversion of the 3×3 pixels. Panel c) shows the complete device with nine inputs, representing the binary array created in panel-b. Panel d) shows the simulation result for the letter L. Here, we took the example of the letter L to show the Workflow, represented by the red lines. 63

3.19 Example of alphabet identifier. a) device simulation using input letter "L" at 50 ns. b) magnetic amplitudes, presented for three distinct cross sections located at the midpoint of each output waveguide along its length. c) zoom out of panel-(b) in the range from 34 to 44 μm. d) legend used for the different arms. 63

3.20 Concept of physical Neural Network devices where Digits from pixels can be classified using optimized structure defects. Panel a) shows that all digits can be described by seven pixels. Panel b) shows the example of digit zero conversion to related frequencies. Panel c) shows the physical magnonic device to classify each digit. . 64

3.21 The problem with the digit identifier is shown in the Figure above. The Left and right side panel shows the spin-wave propagation of frequencies representing number 1 and number 7, respectively. The badge in the output guides of both figures shows the waveguide with the highest amplitude, suggesting the input number. The two panels show that the waveguide with the highest amplitude is less distinguishable than in the demultiplexer case, due to the frequency similarity between the two numbers. The insets in the top left corner of each Figure with badge show that 4 frequencies are the same in both numbers. 65

4.1 Correspondance between the colorbar for magnetic moments projected on the XY plane and related arrows. We will use the same colorbar to describe the domain images in this chapter. In addition to that, we note that the black and white colors show the magnetisation in positive Z and negative Z direction, respectively. 70

4.2 Simulated domain images with different Ku_1 (repored on the side of the panel) with A_{ex} (reported on the top of the panel). The color bar is the same for all the images, consistently with Fig-4.1. 71

4.3 Schematic of an artificial neural network with three input and one output neurons. Here, w_i, x, and f_i are weights, input values, and threshold function respectively. 73

4.4 Convolutional layer expression between (a) the image and (b) the example kernel. The result of the activation map is shown in panel (c). 75

4.5 An example of image separation into the three primary colors: Red, Green, Blue 76

4.6 The two different pooling operations: a) the original image; b) the average pooling; C) the max pooling. 77

4.7 Simulated domain images with different Ku_1 (reported in the top of the panel) with A_{ex} (reported on the left of the panel). The scale bar is the same for all the images, as shown in Fig-4.1. 80

4.8 Details of custom neural network with different descriptions: a) schematic for the reduction in dimension of the input image in each layer of CNN, b) flow chart of a custom CNN model, c) qualitative working principles of CNN. 81

4.9 Evolution of images with the application of different filters on the input image and feature extraction from different layers: a) distribution of out-of-plane magnetization, b) domain boundaries, c) curvature or branching. 83

4.10 Evolution of loss function for different CNN models as a function of epochs. Here loss for training and validation images are shown by solid and dashed lines, respectively. The blue solid line represents a reference level at 5%. 85

4.11 Experimental and simulated domain images, obtained with the input parameters derived from the image regression algorithm, applied on the experimental domain image. **a**) shows the experimental image of the sample. **b**) shows the greyscale conversion of the image-a. **c**) shows the simulated image from the MuMax3 where different parameters were predicted using the trained CNN model on image-(b). 88

1 $Mg_{0.5}Ca_{0.5}Se$ rocksalt supercell: Mg is reported in orange, Ca in blue and Se in green. The supercell is obtained alternating layers of Mg and Ca in the cation sub-lattice along the c primitive vector. . . 110

2 Representation of 12 alphabets possible to visualize by the 3×3 matrix. 114

3 Series of normalized (a) AFM images with simultaneously captured corresponding (b) MFM images for the films irradiated with 100 *kev* of Ar^{+} ions at various fluences (mentioned on top of the image). The scale bar is same for all the images. 115

4 Lateral dimensions decrement in the ResNet-34 [9]. Picture taken from ref-[10] 117

List of Tables

2.3 foo 19

2.1 1D formulas, along with related statistics: $avg(RMSE)$ denotes the root mean squared error for average over 1000 random train-test splits of dataset. Instead, the RMSE is the root mean squared error for the entire dataset as training and test. Similarly, the R^2 values are calculated considering the entire dataset and they show the quality of fit between predicted and actual values. The success rate (in percent) shows how many RS or ZB phases out of 82 have been correctly identified by the descriptor. The “Generator type" column indicates the different generators used to produce the corresponding descriptor. RMSEs are in eV. 20

2.2 1D formulas after the optimization step, along with related statistics. Notation as in table-2.1. RMSEs are in eV. Here we use RMSE as target function 20

2.4 2D formulas, along with related statistics. Notation as in table-2.1. RMSEs are in eV. 27

2.5 3D formulas, along with related statistics. Notation as in table-2.1. RMSEs are in eV. 27

4.1 Comparison of R^2 scores, and time of training for 1000 epochs (500 epochs with each of frozen and unfrozen weights) for different CNN models. The time for training is based on an Intel i5 quad core processor. 86

1 Different verification parameters for 1D, 2D and 3D descriptors calculated in the present work and descriptors presented in Ref.[4]. Here, avg(RMSE train), avg(RMSE test) and RMSE indicate the root mean squared error for training data, test data and full dataset, respectively. R^2 and Pearson coeff are goodness parameters. Max_E and Min_E show the maximum and minimum absolute error in prediction. 110

2 Values related to 82 AB binaries: total energy difference between Rock-Salt and Zinc-Blende ($\Delta E = E^{RS} - E^{ZB}$) (calculated using DFT) and seven atomic properties of corresponding A and B atom[4]. 111

Nomenclature

AFM	Atomic Force Microscopy
ANN	Artificial Neural Network
AP	Atomic Properties
BS	Binary Search
CNN	Convolutional Neural Network
CPU	Central processing unit
DFT	Density Functional Theory
DL	Deep Learning
EA	Electron Affinity
EDSM	Encoder-Decoder Sequence Mode
EDSM	Encoder-Decoder Sequence Model
GA	Genetic Algorithm
GPU	Graphics processing unit
HOMO	Highest Occupied Molecular Orbital
I-P	In-Plane
IP	Ionisation potential

LDA	Local Density Approximation
LR	Linear Regression
LUMO	Lowest Unoccupied Molecular Orbital
MF	Material Features
MF	Material’s Feature
MFM	Magnetic Force Microscopy
ML	Machine Learning
NN	Neural Network
OOP	Out of Plane
PC	Personal Computer
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
RS	Rock Salt
SHAP	SHapley additive exPlanations
SOC	System On Chip
SRT	Spin Reorientation Transition
TPU	Tensor processing unit
VASP	Vienna Ab initio Simulation Package
YIG	Yttrium Iron Garnet
ZB	Zinc Blende

Chapter 1

Introduction to Machine Learning in Materials Science: Exploring Multiple Scales from Atomic to Macroscopic

Computational materials science has undoubtedly been a driving force in materials characterization, design and discovery in the last half-century[11–15]. Indeed, simulations are well-known to reduce time and cost in comparison to experiments, shortening the expensive trial-and-error process usually required in experimental materials science. Nowadays, to support simulations, scientists are increasingly benefiting from machine learning (ML) methods[16–18]. As well known, ML is currently revolutionizing various research and technology fields and materials science makes no exception. The use of ML in materials science provides several advantages over traditional methods. For example, ML techniques can analyze large and complex data sets, enabling researchers to identify patterns and correlations that may not be apparent with conventional approaches[19–22]. Additionally, ML techniques allow for the rapid screening of materials for specific applications and ML models can be trained to predict the properties of new materials based on existing data[4, 23]. Although ML offers numerous advantages, it shows limitations as well. In fact, the incorporation of ML into the "simulation box," - thereby including its combination with atomic and micro-scale simulations - poses several challenges:

1. From the ML side, interpreting complex ML models can be difficult, making it challenging to understand how the model makes predictions and what features and physical ingredients are most important[24].

2. ML models trained on data from one materials class may not be transferrable to other classes, making it challenging to generalize results across different sets of compounds.[25]

3. Training ML models require a large amount of high-quality homogeneous data, which can be time-consuming and expensive to generate, particularly within accurate quantum-mechanical techniques[26].

4. Sometimes *overfitting* occurs, *i.e.* referring to the situation in which a model - usually overly complex - fits the training data too closely, however leading to poor performance on new data (not present in the training dataset) [27].

ML can in principle be used for any material system at any scale, given its multifaceted nature[28]. The current thesis concentrates on three material science problems at three different material scales. These problems address diverse ML challenges and will be thoroughly examined, along with the employed ML techniques and the attained solutions. We will cover three topics related to materials science: *i*) we will present the application of linear regression[29–32] in predicting the most stable crystallographic phase in a specific class of materials (binary semiconductors). *ii*) we will demonstrate how Genetic Algorithms (GA)[33] can be used to construct a magnonic neural network. *iii*) we will showcase the Convolutional Neural Network(CNN) implementation in relation to magnetic domain patterns.

Linear Regression starting from Density Functional Theory data

In the last 50 years, Density Functional Theory (DFT) has revolutionized the field of materials science by enabling accurate predictions of materials' properties starting from their electronic structure[2, 34]. Being "ab-initio" in nature, DFT does not need any previous information or data, apart from the atomic number and the coordinates of the elements that are being considered. DFT applications range from designing new materials to understanding their behavior under different conditions. Kohn-Sham equations[3], at the basis of DFT, are transparent and transferrable: they can be equally applied to all kinds of materials (from binaries to ternaries, from metals to insulators, from bulk to nanostructures, etc.), without the need to build a new model every time one changes materials' class. However, despite its many benefits, DFT also has limitations. The accuracy of DFT is influenced by the specific approximation used for the exchange-correlation functional, making it challenging to treat excited states or correlated electrons or systems characterized by weak bonding[35]. Additionally, while DFT codes become more and more computationally efficient, large systems (with a number of atoms higher than a few thousands) still require significant computational resources. Anyhow, since DFT data are considered to be highly accurate, different ML methods are often exploited for data analysis based on the vast amount of data generated through DFT simulations.

On the ML side, various approaches exist nowadays, each characterized by advantages and disadvantages. Let's take, as a paradigmatic example, Deep Learning (DL), a type of ML algorithm relying on artificial neural networks to make complex predictions. Although DL algorithms can be very efficient in predictions, they are challenging to interpret due to their complex and often non-transparent structure (see specific points 1 and 2 from the previously reported list-1). Researchers therefore resort to alternative approaches, such as linear regression, logistic regression[36], and decision trees[37], to improve interpretability. In the first part of the thesis, the latter approach - privileging interpretability - was the one adopted. We started with a case study to predict the difference in total energy between two crystal structures in semiconductor binary compounds[4]. Linear regression was selected as the algorithm of choice, because it allows the creation of formulas linking the target energy difference to an interpretable set of basic atomic features. We used a dataset from a previous study conducted by Ghiringhelli et al. as a benchmark. We generated multiple combinations of fundamental atomic properties of the material constituents and employed these features to train a model able to predict the energy difference between two structures. The best-performing formulas were then analyzed to understand the role of specific atomic features in determining the stability of the crystal structure. We tested the predictive capability of the derived formulas on "new" compounds (*i.e.* which were not present in the training dataset) and found satisfactory agreement with first-principles results. Furthermore, we compared our results with those reported in a previous study using a different strategy for obtaining the descriptors. Our findings revealed a 40% improvement in accuracy compared to the previously reported results.

Genetic Algorithm for developing Magnonic devices

As previously mentioned, ML tools are not restricted to the atomistic scale in materials science and engineering. Rather, they have a wide range of applications throughout the materials world. Indeed, ML can be applied not only at the level of individual atoms but also to larger length scales, which is of interest for the micro/macroscopic characterization of materials and fabrication of devices[38].

On the technological side, magnonics is an innovative field that offers a new approach to low-power information processing[39]. Unlike conventional electronics, which rely on electrons to carry and process data, magnonics uses magnons, the quanta of spin waves, to perform these tasks, complementing modern electronic processors[39]. Despite the development of many magnonic devices, they are typically designed for specific functions and require specialized investigations. To overcome this limitation, a new method called “inverse-design[40] magnonics" was introduced by Wang et al.[6], a powerful tool in which a feedback-based computational algorithm is used to design magnonic devices for specific functions. For example, it was used to create devices that can separate two wavelengths of spin waves by exploiting the interaction of spin waves with defects. Micromagnetic simulations on a prototype device consisting of a rectangular ferromagnetic area with square-shaped defects have demonstrated this concept[6]. However, to optimize the configurations of defects more efficiently, a finer technique than the method used by Wang et al. is desirable. In addition, one of the challenges in developing novel magnonic devices is represented by the high amount of data needed to train ML models [41](cfr point 3 of the previously mentioned list).

To address this challenge, we tried two ML techniques, referred to as Binary search and encoder-decoder sequence–to–sequence model. Here, we faced problems similar to those reported in points number-3 and 4 of the list. Therefore, we optimized the structure configuration using the GA[33] . The GA is a search algorithm inspired by natural selection and can be used to create an efficient design by minimizing a defined cost function. This leads to a reduction in the number of simulations required to train the model, therefore minimizing the time and computational resources required and making the process far more efficient.

The combination of inverse-design magnonics and GA also allows for optimizing complex magnonic configurations with multiple inputs and outputs. We demonstrated this by creating a magnonic demultiplexer (as benchmark result), alphabetical letter, and numerical number identifier; the latter examples illustrate the universality of this method for linear, nonlinear, and nonreciprocal magnonic functionalities. The advancement of magnonic devices and the implementation of inverse-design magnonics can significantly impact various applications. Low-power information processing using magnons has the potential to transform the field of electronics, and the combination of inverse-design magnonics with GA provides a powerful tool to optimize complex magnonic configurations, as shown by our benchmark outcomes produced with fewer iterations.

CNN for understanding magnetic domains

Perpendicular magnetic anisotropy (PMA) thin films have revolutionized the field of high-tech applications, such as ultra-high-density magnetic storage, fast memory devices, and nanosensors[42–45]. These thin films exhibit a favorable atomic ordering in ultrathin stacking of Co with Pd or Pt, resulting in PMA[46, 47]. However, the magnetic microstructure of PMA thin films and multilayers is a critical factor in the magnetization reversal process[48, 49]. Structural manipulation, such as that obtained by Ion Beam Radiation (IBR), has proven to be effective in changing the magnetic microstructure and selectively improving the properties of materials[50–52]. Indeed, ion beam radiation is a powerful tool used to modify the properties of materials: by irradiating materials with high-energy ion beams, various changes can be induced in their structure, morphology, and properties, including the creation of defects, the formation of new phases, and the modification of surface properties[53]. In this respect, researchers have focused on understanding the effect of IBR on the images of magnetic domain patterns in Co/Pd bi-layers.

One of the challenges faced in this field is the lack of a sufficiently large number of experimental images to train any ML model. To address this issue, we used micromagnetic simulations to generate a sufficiently vast set of images by changing four critical parameters: exchange length, Anisotropy constant, damping constant, and temperature. In this respect, we note that some ML approaches also face challenges, such as overfitting[54] (cfr point 4. mentioned in the ML challenges list).

Convolutional Neural Networks (CNNs) offer an excellent solution to these challenges. CNNs are specifically designed to process and analyze visual data, using convolutional layers to extract features from images and pooling layers to reduce their size while maintaining essential information. As well known, CNNs are better suited for image recognition and computer vision tasks than traditional neural networks. In this work, we have used CNNs to study magnetic domains in perpendicularly magnetized multilayers subjected to ion-beam irradiation. We have compared the performance of different pre-trained libraries, such as ResNets[55] and VGG[56], with that of CNNs. The results showed that CNNs outperformed the other ML models, achieving an accuracy of almost 94% in predicting the critical parameters. The potential applications of this research are significant, paving the way for enhanced PMA thin films and multilayers.

The thesis is organized in three “self-contained" chapters, each of them being based on a different ML methodology, tackling a different problem and proposing different solutions. In closer detail, the thesis is organized as follows: Chapter-2 shows the use of ML for the prediction of a stable crystal structure starting from DFT data. Chapter-3 presents a compelling application of ML in materials science at the micro-scale, where GAs are used for in-plane micro-magnetic simulations. Chapter-4 showcases the tremendous potential of ML by implementing Convolutional Neural Networks (CNNs) in predicting multilayer perpendicular magnetic properties. By devoting a different chapter to each application of ML, we therefore aim at showing how ML can be proven to be a valuable and innovative tool in solving complex problems in various aspects of materials science.

Chapter 2

Role of Atomic Properties in the Crystal Structure Stabilization: insights from ML

2.1 Introduction

Discovering novel materials has always represented a challenging task for material scientists and engineers. Traditionally, it has been supported by intuition and previous experience of scientists, an approach being time and resource consuming though. Researchers have recently used advanced simulation tools to predict materials' properties even before experimental synthesis. For example, first principles methods have been shown to well describe the electronic properties of a given crystal structure or molecule, which can in turn be used to predict a wide range of physical response functions. The development of quantum mechanics provided a rigorous theoretical foundation for chemistry and material science. However, despite the exponential growth in the available computing power, first principles-based techniques for predicting the properties of materials remain rather time consuming. The combination of statistical methods with high-throughput datasets can help to identify potential candidates as novel materials for targeted applications. Although machine learning techniques in data analytics have drastically advanced, the process phase for transforming raw physical data into quantitative descriptions, as required for employing these algorithms, is yet to transpire.

Modeling material properties with high accuracy and low computational cost still represents one of the grand-challenges in materials science and engineering.

As for ab-initio methods, repeatedly shown to provide accurate tools for material properties prediction, the continuous growth of available computational power [57] has stimulated scientists to move in the direction of high-throughput simulations[58–66]. Along this line, open access databases, such as OQMD[67][68], NOMAD[69, 70], Aflowlib[71], C2DB[72, 73], QPOD[74], Materials Project[75], Materials Cloud[76] and related AiiDa[77, 78], provide researchers with a huge collection of basic first-principles results. A large amount of ab-initio data is thus available, which can be used for deeper analyses and studies, provided one can count on proper tools to extract relevant information out of them.

In the last years, materials scientists have developed different Machine Learning (ML) methods to rationalize the data analysis [79–89]. Each method has its own specific advantages and limitations. Methods like Random Forest [90] or Neural Networks (NN)[91], which are mainly behind the Deep Learning (DL), are very efficient[92] but not always transparent, partially blurring the comprehension of the role played by the input variables in the final results. Nonetheless, over the last decades, improvements towards the interpretability of such "black-box" ML models have been made through additional methodologies [93], such as model-agnostic methods, which in turn are divided into global and local interpretation techniques (see Ref.94 and references therein). For instance, out of various global methods, we can cite: the *permutation feature importance* [95, 96], which associates to each feature an importance values depending on how much the model error increases when its values are shuffled; the *functional decomposition* [97], which decomposes the complex prediction function into smaller parts; the *global surrogate* [98], which replaces the original model with a simpler one that can be more easily interpreted. On the other hand, among the local methods, we can cite: the *local surrogate models* (LIME) [99], which replaces the complex model with a locally interpretable surrogate model; the *SHapley Additive exPlanations* (SHAP) [100], which is based on Shapley values and computes the contribution of each feature to the prediction.

Also in the specific case of DL, which structures algorithms in multiple layers to create "artificial neural networks" (therefore enhancing the complexity in the prediction's interpretation), other specific interpretation methods have been proposed [101–103], in addition to the already cited model-agnostic ones.

However, when targeted case studies allow, the easiest way to achieve a deeper understanding of machine learning results is to rely on interpretable models, such as Linear regression (LR)[29–32], logistic regression [36] and Decision trees [37]. This situation can apply here to our target case, *i.e* - as detailed at length below - the prediction of the difference in total energy (ΔE) between rock-salt (RS) and zinc-blende (ZB) crystal structures in semiconductor binary AB compounds, a prototypical case in material science. Accordingly, being our goal the creation of formulas linking the target label (ΔE) to a set of basic atomic features, we selected the LR algorithm, both in its one-dimensional and multi-dimensional forms.

In closer detail, we here propose a ML-based approach to build sets of features (or descriptors) starting from a given set of basic variables (e.g., atomic properties), which are subsequently used to construct LR models (or formulas). The final outcome of our procedure is a transparent formula, not necessarily of easy mathematical formulation, but revealing which part of the input mostly affects the output[104], *i.e.* allowing the identification of the main driving physical features.

As anticipated, to test our method, we target a prototypical case in material science: indeed, inspired by the original work of Ghiringhelli *et al.* [4], we optimized our models to predict the difference in energy between rock-salt (RS) and zinc-blende (ZB), from that optimization a classification of the most stable crystal structure between RS and ZB for semiconductor AB binary compounds naturally derives. To identify useful features, we generate combinations of basic atomic properties (*i.e.* the independent variables in our approach) of the material constituents through a combinatorial approach[105]. We then carry out an analysis of the emerging best-performing formulas, identifying the role of specific atomic features in determining the final stabilization of the crystal structure. Finally, we test the predictive capability of the obtained formulas by applying them to “new" compounds (*i.e.* outside the dataset used for training the model), finding an overall satisfactory agreement with first-principles results. As already mentioned, our approach is similar to what originally proposed by Ghiringhelli *et al.* [4], though with some differences and further extensions, which will be carefully discussed in what follows. Let us also remark that large packages are nowadays available to the scientific community, already providing advanced and well-tested features for use in ML applied to materials science [106, 107]. However, we here strictly followed the spirit of Ghiringhelli *et al.* [4] and therefore chose their same atomic features, as detailed below. In the spirit

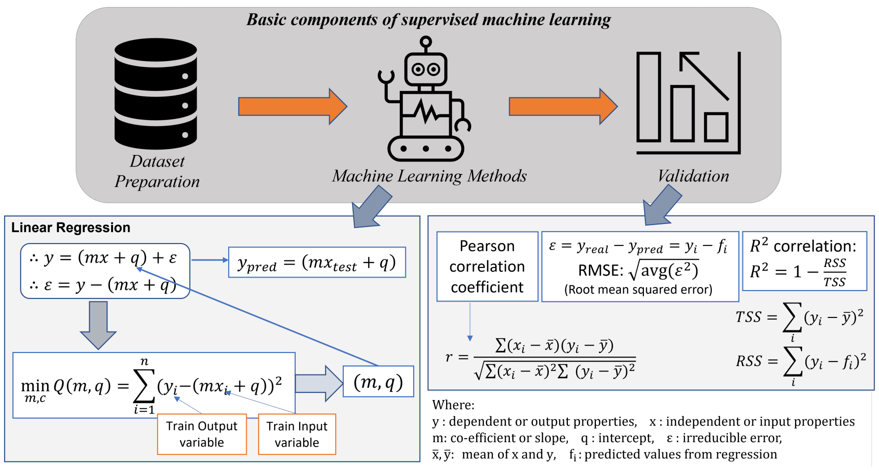

Basic components of supervised machine learning
Dataset Preparation
Machine Learning Methods
Validation
Linear Regression
$\therefore y = (mx + q) + \varepsilon$
$\therefore \varepsilon = y - (mx + q)$
$y_{pred} = (mx_{test} + q)$
$\min_{m,c} Q(m,q) = \sum_{i=1}^{n} (y_i - (mx_i + q))^2$
(m, q)
Train Output variable
Train Input variable
Pearson correlation coefficient
$\varepsilon = y_{real} - y_{pred} = y_i - f_i$
RMSE: $\sqrt{\mathrm{avg}(\varepsilon^2)}$
(Root mean squared error)
R^2 correlation:
$R^2 = 1 - \frac{RSS}{TSS}$
$TSS = \sum_i (y_i - \bar{y})^2$
$RSS = \sum_i (y_i - f_i)^2$
$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$
Where:
y : dependent or output properties, x : independent or input properties
m: co-efficient or slope, q : intercept, ε : irreducible error,
$\bar{x}, \bar{y}$: mean of x and y, f_i : predicted values from regression

Fig. 2.1 Different components used in machine learning methods and detailed for the Linear regression. The steps include: Dataset preparation, Selection of machine learning method and Validation.

of comparing the two approaches, we also compare our results with the reference paper[4], where the authors have used a different strategy to obtain the descriptors.

The work presented in this chapter is in collaboration with Prof. Loriano Storchi(a), Danila Amoroso(b) and Francesco Delodovici(b). (a. Univ. Chieti, (b. CNR-SPIN).

2.2 Methodology

The approach we present here can be regarded as a combinatorial machine-learning: a set of basic atomic properties (APs, listed in Table-2 (Appendix)) are randomly combined (though under certain initial constraints detailed below), to build a set of material features (MFs). The generated features are then used to train a LR model, where the energy difference between rocksalt and zincblende structures is the dependent variable (i.e., the label). Then, we select the best performing model according to standard performances metrics, such as the Root Mean Squared Error (RMSE). The final result of this procedure is a “formula", which is a concise and clear representation of the relationship between the used atomic properties and the energy difference between RS and ZB phases. In the following, we describe in detail the different steps of our approach which are described in the Fig-2.1.

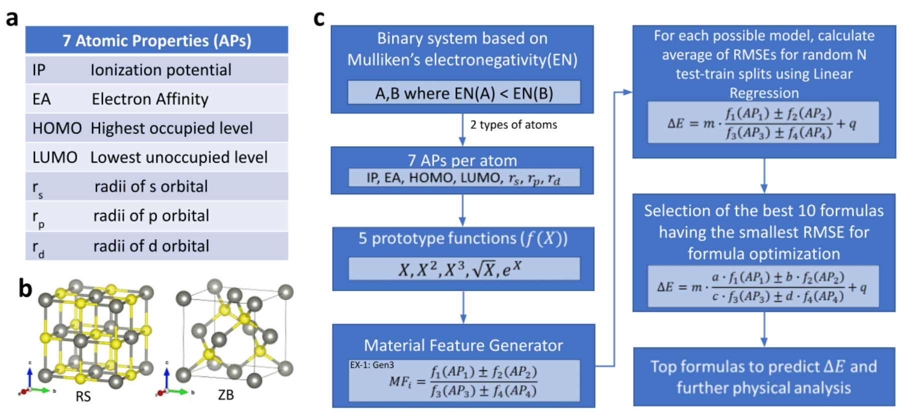

a
7 Atomic Properties (APs)
IP Ionization potential
EA Electron Affinity
HOMO Highest occupied level
LUMO Lowest unoccupied level
r_s radii of s orbital
r_p radii of p orbital
r_d radii of d orbital
b
RS
ZB
c
Binary system based on Mulliken's electronegativity(EN)
A,B where EN(A) < EN(B)
2 types of atoms
7 APs per atom
IP, EA, HOMO, LUMO, r_s, r_p, r_d
5 prototype functions ($f(X)$)
$X, X^2, X^3, \sqrt{X}, e^X$
Material Feature Generator
EX-1: Gen3
$MF_i = \frac{f_1(AP_1) \pm f_2(AP_2)}{f_3(AP_3) \pm f_4(AP_4)}$
For each possible model, calculate average of RMSEs for random N test-train splits using Linear Regression
$\Delta E = m \cdot \frac{f_1(AP_1) \pm f_2(AP_2)}{f_3(AP_3) \pm f_4(AP_4)} + q$
Selection of the best 10 formulas having the smallest RMSE for formula optimization
$\Delta E = m \cdot \frac{a \cdot f_1(AP_1) \pm b \cdot f_2(AP_2)}{c \cdot f_3(AP_3) \pm d \cdot f_4(AP_4)} + q$
Top formulas to predict ΔE and further physical analysis

Fig. 2.2 a) Basic atomic properties (APs) used to construct the material features. b) Crystal structures of RS and ZB (plot made using the VESTA tool) [1]. Grey (yellow) spheres represent A (B) atoms. c) Workflow for formulas construction, machine learning methodology, validation, and MF selection procedures. In the AB compounds, A is the atom with the lowest electronegativity.

2.2.1 Dataset preparation and materials

As mentioned above, we aim at predicting the total energy difference ($\Delta E = E^{RS} - E^{ZB}$) between RS and ZB phases of cubic crystal structures for 82 semiconductor binary AB compounds (the dataset is reported in table-2 in SI). We employed total energies reported in Ref. [4], which were calculated through density functional theory (DFT)[2][3] within the local density approximation (LDA [34]). The RS and ZB are two cubic crystallographic phases which are often found as ground state in binary semiconductors. One of the most important differences between the two phases stands in the bond coordination, which is octahedral for RS and tetrahedral for ZB.

The construction of the material features (MFs), is based on primary atomic properties of the constituents, also taken from Ref. [4]. To facilitate the physical interpretation of each MF, the APs are subdivided into two different kinds: (*i*) “energy" properties, including highest occupied Kohn-Sham level (HOMO), lowest unoccupied Kohn-Sham level (LUMO), Ionization Potential (IP), Electron affinity (EA); (*ii*) “spatial" properties, including r_s, r_p, and r_d, *i.e.* the radii where the radial probability density of the valence *s*, *p*, and *d* orbitals, respectively, reaches its maximum.

2.2.2 Formulas construction

We rely on the LR[29, 30] approach to obtain a direct interpretation of the dependent and independent variables. The construction of a useful LR model can become troublesome, requiring a linear dependence between features. In Ref.[4], the authors implemented an automated feature selection method employing the LASSO regression analysis method [108, 4]. In our work, we use a combinatorial approach to generate the dependent variable (material features) to be used within the linear equations, and thus to finally obtain the formulas.

In Fig. 2.2, we illustrate the workflow of the formula generation and selection using LR. The process starts with the selection of the APs to be combined. Afterwards, we choose prototype functions that are simple analytical operations applied to the APs. In our case, we selected 5 prototype functions, $f(x)$, namely $x, x^2, x^3, \sqrt{x}, e^x$. where x is an AP. Then, we obtain the final set of MFs by combining different prototype functions via the combinatorial approach (see for instance [105]), and applying the following additional set of rules:

- $GEN1$: combine two prototype functions in the numerator, forcing them to belong to the same kind of APs, that is both “spatial"-like or both “energy"-like; one prototype function is at the denominator with the only constraint to be non-zero, such as
$$MF = \frac{f_1(AP_1) \pm f_2(AP_2)}{f_3(AP_3)} \tag{2.1}$$
- $GEN2$: combine two prototype functions with same kind of APs at the numerator, and a single prototype function at the denominator with argument of a different kind with respect to the numerator ones. For instance considering equation-2.1, if AP_1 in $f_1(AP_1)$ and AP_2 in $f_2(AP_2)$ is an “energy" term (*i.e.* EA or $HOMO$), then AP_3 must be a “spatial" term, (*i.e.* r_p)
- $GEN3$: combine two prototype functions at both the numerator and denominator without any constraints
$$MF = \frac{f_1(AP_1) \pm f_2(AP_2)}{f_3(AP_3) \pm f_4(AP_4)} \tag{2.2}$$

- *GEN*4: combine two prototype functions with the same physical dimensions at both the numerator and denominator

$$MF = \frac{f_1(AP_1) \circledast f_2(AP_2)}{f_3(AP_3) \circledast f_4(AP_4)} \tag{2.3}$$

  where where we introduce a generic $\circledast$ symbol to indicate different mathematical operations (i.e.$\circledast = + - \times \div$)

Each one of these set of rules corresponds to a different MFs generator.

From the implementation point of view, each generator is a Python [109] function that produces a set of strings. Therefore, we can easily exploit the Python capability to parse a source code and run Python expression (code) within a program [110] to compute all the MFs' values starting from the generated sets of strings. This allows for an easy implementation and plugin of other generators, as well as to easily adopt different sets of Atomic Properties, leaving the workflow unchanged: a new generator can be introduced implementing a Python function returning a list of strings, each one being a valid MF.

Finally, in order to choose the optimal formula, we build a LR model for each of the generated MF. To practically select the best model, *i.e.* the "best formula", we randomly split the full dataset into : 90% as training set to train/initialize the model; 10% as a test set to check model's performance. We perform this random splitting N times (with $N = 150$) for each model, and we calculate the RMSE from the test set for each run. Afterwards, we again verify the top 10 resulting best formulas with a higher value of training set and test set splitting, with $N = 1000$. We average it over all N splitting, and we obtain $avg(RMSE)$, as reported in our Tables.

We mention that different metrics for evaluating regression model can lead to different formulas ranking. In this work, we rank the obtained models based on the lowest $avg(RMSE)$ for direct comparison with a previous work [4].

2.2.3 Formulas optimization

In order to further improve the performance of our models, we introduce an additional step, which we refer to as "formula optimization". In detail, we focus on the top 10 formulas obtained using each generator and the subsequent LR, as described in the previous section. After that, we use a grid search to find the relative weights of each prototype function of the atomic properties (i.e., each $f_i(AP_i)$) within the formula. A first grid search ranging between -1 to 1 with the increasing step of 0.1 is used simultaneously for all the weight coefficients (i.e. an exhaustive search through the specified subset of values for a, b, c, d coefficients is simultaneously performed). We multiply each $f_i(AP_i)$ of the formula by the weight coefficient and we optimize the final RMSE value. Once the procedure finds a set of optimal weight coefficients, two subsequent grid searches, with reduced incremental step values (0.01 and 0.001 respectively) and range of search are performed to obtain the final set of refined weight coefficients. Of note, for each set of weight coefficients generated during the grid search, we also run the linear regression. Thus, we are performing a proper formula optimization, as at each step of the grid search we are updating both the weight coefficients as well as the slope and intercept coming from the LR. In addition, it is important to underline that we made sure to find the global minimum solution when analysing the N-D maps over the phase space of the parameters (in the specific case of $GEN2$, we also double-checked the optimization results through an analytical minimization approach via Mathematica [111, 112]).

To further clarify the procedure, we show here an exemplary equation:

$$\Delta E = m \times \frac{a \times f_1(AP_1) \circledast b \times f_2(AP_2)}{c \times f_3(AP_3) \circledast d \times f_4(AP_4)} + q \tag{2.4}$$

where ΔE is the targeted material feature (MF), a, b, c, d denote the weight coefficients scanned during the grid search, $f_1(AP_1), f_2(AP_2), f_3(AP_3), f_4(AP_4)$ are the prototype functions build on the primary atomic properties AP_i, and m and q are the the slope or angular coefficient and intercept, respectively, recursively determined upon LR. In Table 2.2, we report the optimized, best performing formula from the different generators.

To benchmark our grid search, we also used automated coefficient-optimizing methods: Nelder-Mead[113], Conjugate Gradient (CG)[114], Broyden– Fletcher–Goldfarb– Shanno (BFGS) [115] and Truncated Newton method (TNC)[116]. Although the resulting sets of coefficients are different in terms of single values with respect to those obtained via the grid search, the ratios between them is almost preserved as well as the associated RMSE. In particular, for the case of $GEN1$ and

*GEN*2, the ratio between the numerator coefficients *a* and *b* is preserved; for *GEN*3 and *GEN*4 also the denominator coefficients ratio, between *c* and *d*, is preserved.

In Fig. S.3 of the Appendix, we show the evolution of the RMSE and different ratios for different methods using 1D feature generated by *GEN*3.

Finally, we would like to underline that the whole procedure, *i.e.* formula construction and its optimization, is not too expensive from a computational point of view. Indeed, as reported in Table S.4 of SI, for almost all the generators the whole computation can be performed in less than four hours on a standard PC. Only GEN3, where 1091200 different formulas are generated and evaluated, is more time consuming (*i.e.* almost 15 hours are needed). However, we underline that the 1D formula construction procedure can be easily parallelized, in order to drastically reduce its computational burden.

2.2.4 Higher-dimensional features

Following Ghiringhelli et al. idea (see Ref. 4), we also build 2D and 3D formulas, as follows: we combined in all possible ways two or three different 1D MFs extracted from the best 1000 ones, and checked the $avg(RMSE)$ using multiple LR for N test-train set splits. Thus the final equations, which relate the ΔE to the basic atomic features, are written as follows:

$$\begin{aligned} \Delta E \;=\;& m_1 \times \frac{a_1 \times f_1(AP_1) \circledast b_1 \times f_2(AP_2)}{c_1 \times f_3(AP_3) \circledast d_1 \times f_4(AP_4)} + \\ &+ m_2 \times \frac{a_2 \times f_5(AP_5) \circledast b_2 \times f_6(AP_6)}{c_2 \times f_7(AP_7) \circledast d_2 \times f_8(AP_8)} + q \end{aligned} \tag{2.5}$$

for the general 2D formulas , and:

$$\begin{aligned} \Delta E \;=\;& m_1 \times \frac{a_1 \times f_1(AP_1) \circledast b_1 \times f_2(AP_2)}{c_1 \times f_3(AP_3) \circledast d_1 \times f_4(AP_4)} + \\ &+ m_2 \times \frac{a_2 \times f_5(AP_5) \circledast b_2 \times f_6(AP_6)}{c_2 \times f_7(AP_7) \circledast d_2 \times f_8(AP_8)} + \\ &+ m_3 \times \frac{a_3 \times f_9(AP_9) \circledast b_3 \times f_{10}(AP_{10})}{c_3 \times f_{11}(AP_{11}) \circledast d_3 \times f_{12}(AP_{12})} + q \end{aligned} \tag{2.6}$$

for the 3D ones. The comparison between performances is discussed in the following Section.

2.2.5 Test of predictive power of ΔE formula for AB compounds outside the dataset

After obtaining the optimised 1D formulas for ΔE in the case of AB compounds, we aimed at further verifying their validity and predictive power, by considering additional AB systems (*i.e.* which were not originally included in the ML training set) and by comparing values obtained from ML-predicted ΔE formula with corresponding *ab-initio* calculated values. In closer detail, we focused on different alloys, obtained by changing respectively the concentration of A-site atoms, such as $[A_xA'_{1-x}]B$, and of B-site atoms, such as $A[B_xB'_{1-x}]$. Accordingly, one can test the efficiency of the formulas by checking the energy difference for intermediate concentrations as obtained from optimised 1D formulas and compare their trend with respect to first-principles results. To this end, ab-initio electronic-structure simulations were carried out within DFT

and LDA functional. Calculations were performed using the VASP[117–119] code, employing a $8 \times 8 \times 8$ k-mesh for the Brillouin zone sampling. We verified that the results obtained with the pseudopotential VASP code for the parent binary compounds were consistent with those reported by Ghiringhelli *et al.*, calculated with the all-electron FHI-aims code [120]. For simulations at different concentrations, we adopted the so called "Virtual Crystal Approximation" (VCA), based on virtual atoms interpolating between the real constituent atoms [121, 122]. However, as well known from the literature, the VCA approach neglects some effects, such as local distortions around atoms and, as such, should not be expected to reproduce fine details of disordered alloys properties [123]. Accordingly, in some cases (*i.e.* for $Mg_xCa_{1-x}Se$ alloys), in order to mimic disordered structures with an improved accuracy, we calculated total energies using supercell structures, rather than using the VCA method on primitive unit cells. Specifically, the considered supercell is the cubic unit cell composed by four AB formula units with planes of cations alternating along the **c** direction (see Figure S.4). The k-mesh was modified accordingly, to maintain the same density of points employed in the simulations of primitive cells.

2.2.6 Validation of the linear regression

We employed different verification parameters to check the model's efficiency, namely: RMSE, Pearson correlation coefficient, R^2 values, and classification accuracy. RMSE represents the root mean square error of the test dataset. The Pearson correlation coefficient, defined in equation 2.7, represents a measurement of input and output property dependence [124]. If two properties are highly dependent one on the other, one gets values closer to 1 or -1, whereas values closer to zero show a much lower dependence. R^2, *i.e.* the coefficient of determination defined in equation 2.8, describes how properly the regression line interpolates the data. Finally, the classification accuracy shows the ability of the linear model to qualitatively distinguish different classes of the dataset, in our case RS and ZB as stable phases.

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \tag{2.7}$$

$$R^2 = 1 - \frac{SS_{residual}}{SS_{total}} \tag{2.8}$$

In equation 2.8: $SS_{total} = \sum_i (y_i - \bar{y})^2$, $SS_{residual} = \sum_i (y_i - f_i)^2$ where f_i are the predicted values; x_i and y_i are values of input and output properties; $\bar{x}$ and $\bar{y}$ are mean of the input and output values.

2.3 Results

In this section, we present the results of our combinatorial approach and explore the contributions of various generators and formula dimensions. We also examine the role of atomic properties and their impact on energy and test the effectiveness of the trained model on new binary systems. Specifically, we investigate how different generators and formula dimensions affect the model's accuracy and compare our approach's performance to other methods in the literature. Additionally, we analyze the influence of atomic properties on the stability and energy of the materials, shedding light on the system's underlying physics.

2.3.1 Results from Formula Search

In this section, we will analyse the final formulas as obtained from different generators as explained in the section-2.2.2. The results are shown in Tables 2.1,2.2,2.4,2.5; in the first row we report the results obtained by Ghiringhelli *et al.*[4] for comparison.

First, by comparing the $avg(RMSE)$ values, we note that all 1D formulas obtained from our different generators better perform with respect to the 1D ones reported in [4], where the authors used the automated feature selection method LASSO [108]. Remarkably, some atomic primary features appearing in 1D formulas of Ref. [4] also appear in our obtained list of 1D formulas using $GEN1$ and $GEN2$; nevertheless, those are characterized by a higher $avg(RMSE)$ than other formulas we obtained via our combinatorial approaches. Additionally, formulas from $GEN3$ show the lowest $avg(RMSE)$ among all the others. We also note, from Table 2.1, that $GEN1$ and $GEN3$ provide lower $avg(RMSE)$ compared to $GEN2$ and $GEN4$ respectively; however, $GEN2$ and $GEN4$ have a higher success rate in terms of classification prediction. classification prediction is the process of using a trained machine learning model to assign a class label to a new data point based on its features. It is a form of supervised learning where the model has been trained on labeled data to learn the relationship between the features and the class labels. For instance, in Table-2.3(Appendix) we report the formula optimized using the success rate as target label. $GEN4$ results indeed the best performing generator with respect to $GEN3$, when targeting ΔE instead. It's noteworthy that the best formula from $GEN4$ shows similar terms to the corresponding case for the ΔE-based optimization. This testifies the fact that the choice of the performances metric to rank the material features

Formula	**avg(RMSE)**	**RMSE**	R^2	**Success Rate**	Generator type
$0.127 \times \frac{0.800 \times EA(B) - 1.000 \times IP(B)}{1.110 \times r_p(A)^2} - 0.352$	0.1457	0.1419	0.89	89%	1D descriptor[3]
$-0.611 \times \frac{0.730 \times r_p(B)^3 - 1.080 \times exp[r_s(B)]}{1.000 \times r_p(A)^2} - 0.342$	0.1224	0.1193	0.89	99%	$GEN1$
$-0.103 \times \frac{1.110 \times IP(A) - 0.030 \times IP(B)}{1.000 \times r_p(A)^2} - 0.247$	0.1343	0.1309	0.91	98%	$GEN2$
$0.719 \times \frac{1.100 \times r_p(B) + 0.200 \times \sqrt{\lvert r_d(A) \rvert}}{0.300 \times r_p(A)^3 + 0.700 \times r_p(B)^3} - 0.382$	0.1158	0.1126	0.93	100%	$GEN3$
$1.442 \times \frac{1.100 \times r_p(B) + 0.200 \times r_p(A)}{0.600 \times r_p(B)^3 + 1.100 \times r_p(A)^3} - 0.418$	0.1194	0.1148	0.93	100%	$GEN4$

Table 2.3 1D formulas after the optimization step, along with related statistics. Notation as in table-2.1. RMSEs are in eV. Here we use "Success rate" as target function

can be different according to the target problem to be studied; different models' performances metric are, in fact, not always correlated.

In order to gather hints on the relative contribution of the individual primary atomic properties to the stabilization of either the rocksalt or the zincblende structure, we extracted the best ten formulas with the lowest $avg(RMSE)$ from each generator (so called “original" formulas) and then apply the formula optimization, as detailed in the previous section-2.2.2. This procedure attributes relative weights to each $f(AP)$, allowing to measure the importance of the individual atomic properties in driving the energy stabilization. In principle, the $avg(RMSE)$ value depends on random test-train splits that we perform to our dataset. Therefore, to reduce the effect of randomization, as a target model performances metric, we rank our optimized formula based on the RMSE of the whole dataset, rather than based on $avg(RMSE)$.

By comparing Table 2.1 and Table 2.2, it is evident that the optimization procedure can further change the formulas ranking, providing a different final “best formula” with respect to the non-optimized formulas. In particular, we notice an improvement in RMSE around 5-10% after the formula optimization.

2.3.2 Effect of atomic radii on ΔE

Interestingly, our results reveal the size of the A cation to play a leading role in the phase stabilization; in fact, the $r_p(A)$ radius appears in the best performing formulas more frequently than the other basic atomic properties. Therefore, we further analysed the dependence of ΔE on $r_p(A)$. In Fig. 2.3, we show ΔE as a function of $r_p(A)$, including fitting curves proportional to $r_p(A)^{-2}$ and $r_p(A)^{-3}$. What can be observed is a clear dependence of ΔE on $r_p(A)$: larger (smaller) $r_p(A)$ favors RS

Formula	**avg(RMSE)**	**RMSE**	R^2	Success rate	Generator type
$0.117 \times \frac{EA(B)-IP(B)}{r_p(A)^2} - 0.342$	0.1455	0.1423	0.89	89%	1D descriptor [4]
$-0.751 \times \frac{r_p(B)^3 - exp[r_s(B)]}{r_p(A)^2} - 0.317$	0.1296	0.1193	0.92	90%	$GEN1$
$0.285 \times \frac{\sqrt{\lvert IP(B) \rvert}+\sqrt{\lvert EA(A) \rvert}}{r_p(A)^2} - 0.387$	0.1367	0.1309	0.91	91%	$GEN2$
$0.774 \times \frac{r_p(B)+\sqrt{\lvert r_d(A) \rvert}}{r_p(A)^3+r_p(B)^3} - 0.303$	0.0995	0.0963	0.95	94%	$GEN3$
$1.155 \times \frac{r_s(B)+r_s(A)}{r_p(B)^3+r_p(A)^3} - 0.368$	0.1103	0.1058	0.94	96%	$GEN4$

Table 2.1 1D formulas, along with related statistics: $avg(RMSE)$ denotes the root mean squared error for average over 1000 random train-test splits of dataset. Instead, the RMSE is the root mean squared error for the entire dataset as training and test. Similarly, the R^2 values are calculated considering the entire dataset and they show the quality of fit between predicted and actual values. The success rate (in percent) shows how many RS or ZB phases out of 82 have been correctly identified by the descriptor. The “Generator type" column indicates the different generators used to produce the corresponding descriptor. RMSEs are in eV.

Formula	**avg(RMSE)**	**RMSE**	R^2	**Success Rate**	Generator type
$0.127 \times \frac{0.800 \times EA(B) - 1.000 \times IP(B)}{1.110 \times r_p(A)^2} - 0.352$	0.1457	0.1419	0.89	89%	1D descriptor [4]
$-1.870 \frac{0.801 \times \sqrt{r_p(B)} - 0.606 \times exp[r_p(A)]}{1.010 \times r_p(A)^3} - 0.968$	0.1191	0.1143	0.93	91%	$GEN1$
$0.477 \times \frac{0.876 \times \sqrt{\lvert HOMO(B) \rvert} + 0.468 \times \sqrt{\lvert LUMO(B) \rvert}}{1.110 \times r_p(A)^2} - 0.372$	0.1340	0.1296	0.91	91%	$GEN2$
$1.609 \times \frac{0.642 \times r_p(B) + 0.502 \times \sqrt{\lvert r_d(A) \rvert}}{1.170 \times r_p(A)^3 + 1.170 \times r_p(B)^3} - 0.309$	0.0991	0.0961	0.95	94%	$GEN3$
$1.207 \times \frac{0.878 \times r_s(B) + 0.200 \times r_p(A)}{0.512 \times r_p(B)^3 + 0.610 \times r_p(A)^3} - 0.359$	0.1045	0.1016	0.94	99%	$GEN4$

Table 2.2 1D formulas after the optimization step, along with related statistics. Notation as in table-2.1. RMSEs are in eV. Here we use RMSE as target function

(ZB). Moreover, there is an overall good agreement with the fit, particularly using the $r_p(A)^{-3}$ function. The latter is, in fact, the most recurrent prototype function detected by the ML models. Such a strong dependence of the energy is not observed with respect to the other atomic properties; other comparative plots of ΔE as a function of other $f(p)$ are reported in Fig.2.4. This behavior is in line with the further observation that the rocksalt structures systematically show larger interatomic distances with respect to the zincblende counterparts (cfr. lattice parameters reported in Ref.4); therefore, larger cations prefer to adopt octahedral coordination (*i.e.* RS) with longer bond-lengths - and bigger polyhedral volume - compared to ZB with tetrahedral coordination.

From the obtained results, we remark that formulas based on "spatial" atomic properties achieve higher ranking, thus better performance, with respect to those including atomic energy terms, both in the original models and in the optimized ones. Accordingly, this behaviour further confirms the primary role played by the atomic size (in terms of steric and/or bonding-related effects), in determining the energetics of the AB compounds, *i.e.* in selecting the preferred crystal structure [4]. In particular, we note from the results that the well-performing GEN3 and GEN4 contain all the four radii ($r_s(A), r_s(B), r_p(A)$ and $r_p(B)$), as expected from a basic understanding of bonding in octet binary semiconductors. Note that GEN3 and GEN4 generally show better performances as they are built to explore a wider space of search (see Section 2.2, and Table S.4 in SI where the number of generated and evaluated formulas is also reported).

Dependencies of the different properties on ΔE

In addition to the dependence on the $r_p(A)^2$, Figure 2.4 reports the dependence of the total energy difference between RS and ZB phases as a function of the other atomic features (*i.e.* except for r$_p$, since that is reported in Fig. 2.3). It appears clearly that only r_s is strongly correlated with the energy difference, at variance with other atomic features where the correlation is small or absent.

2.3.3 Proof of concept using alloy systems

In the aim of further proving such trends and validate the implemented combinatorial ML method, we study the energetics in alloys of the type $[A_xA'_{1-x}]B$ and $A[B_xB'_{1-x}]$,

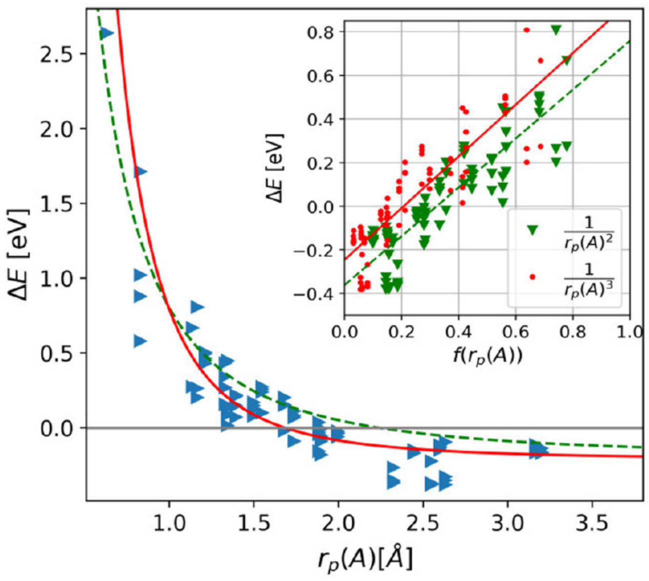


Fig. 2.3 Energy difference between rocksalt and zincblende, ΔE (in eV), as a function of $r_p(A)$ for different binary compounds (blue triangles). Data fit functions are also shown, using proportionality to $r_p(A)^{-2}$ and $r_p(A)^{-3}$ (cfr: $f(r_p(A))$) via green dashed line and red straight line, respectively in the main graph and insets. Insert shows the linear fit of ΔE for $f(r_p(A)) \in [0:1]$, as a function of $1/r_p(A)^2$ (green triangles) and $1/r_p(A)^3$ (red dots). Resulting slope (m) and intercept (q) are respectively: 1.120 (eV Å^2), -0.360 (eV) for $1/r_p(A)^2$, and 1.184 (eV Å^3), -0.240 (eV) for $1/r_p(A)^3$.

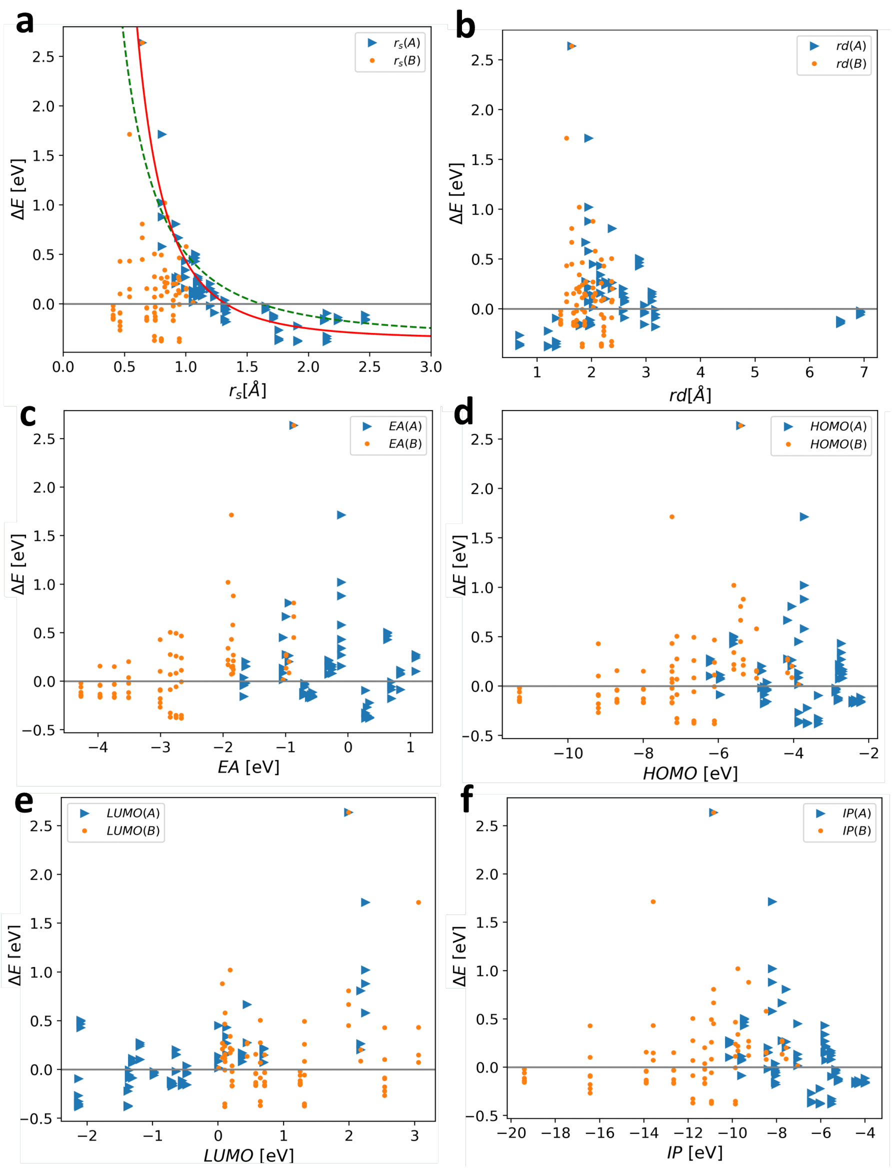


Fig. 2.4 Dependence of ΔE on atomic features: a) r_s, b) r_d, c) EA, d) $HOMO$, e) $LUMO$ and f) IP. Orange dots (blue triangles) indicate values relative to the B (A) atoms. In panel-**a**, we perform a fit using a function $f(x)$ proportional to x^{-2} (dotted green line) and to x^{-3} (straight red line).

where x is the relative concentration of the mixing ions, monotonically tuning thus the average size of one ion with respect the other. All the alloy input properties were linearly interpolated between corresponding values for end binaries (*i.e.* AB and $A'B$ in the $[A_xA'_{1-x}]B$ case), according to the Vegard's law [125]. For the A-ion mixing case, we considered SrSe, CaSe, MgSe, BeSe as parent AB compounds, already included in the original dataset. We then predicted the energy differences between RS and ZB phases for varying concentrations using the original and optimized 1D formulas constructed via $GEN3$ and $GEN4$ generators (Table 2.1 and Table 2.2, respectively). To confirm the obtained predictions, we thus calculated the energy difference via DFT simulations, for a few intermediate concentrations. The results, shown in Fig. 2.5, demonstrate an overall agreement between first-principles calculated and machine-learning predicted energetics. In particular, we notice a change of sign in ΔE, reflecting the change in the stability of the RS with respect to the ZB phase, when moving from the larger Strontium to the smaller Beryllium at the A-site, in line with the previously discussed relation between atomic radii of the A-ions and phase stabilization. At variance, no such change of phase is observed when mixing ions at the B-site, keeping fixed the A-type one. This is confirmed, by looking at the energetics in $B[Sb_{1-x}P_x]$ and $Sr[Se_{1-x}S_x]$ alloys, shown in Fig.2.6(a) and Fig. 2.6(b), respectively. Despite the changing size of the average B-site, the two systems preserve the crystal structure adopted by the the parent compounds, *i.e.* rocksalt for the Sr-based compounds and zincblende for the B-based compounds. Such a behavior is still in line with preferred atomic structure fixed by the ion at the A-site, consistently with Strontium being larger than Boron. Qualitative agreement between ML-predicted and DFT-calculated energetics is observed again.

2.3.4 Higher dimensional features

After discussing the results related to 1D models, we now comment about the higher dimensional formulas. Our best 2D and 3D formulas from different generators are reported in Tables 2.4 and 2.5, respectively.

To visualize the performance of the obtained formulas, we reproduce in Fig. 2.7 the scatter plots of DFT-calculated energies as a function of model-predicted energy differences for the best formulas obtained by $GEN3$ - in terms of $avg(RMSE)$ - for 1D, 1D after formula optimization, 2D, and 3D models. From these, one can infer the quality of the prediction for the different approaches: the narrower the area between

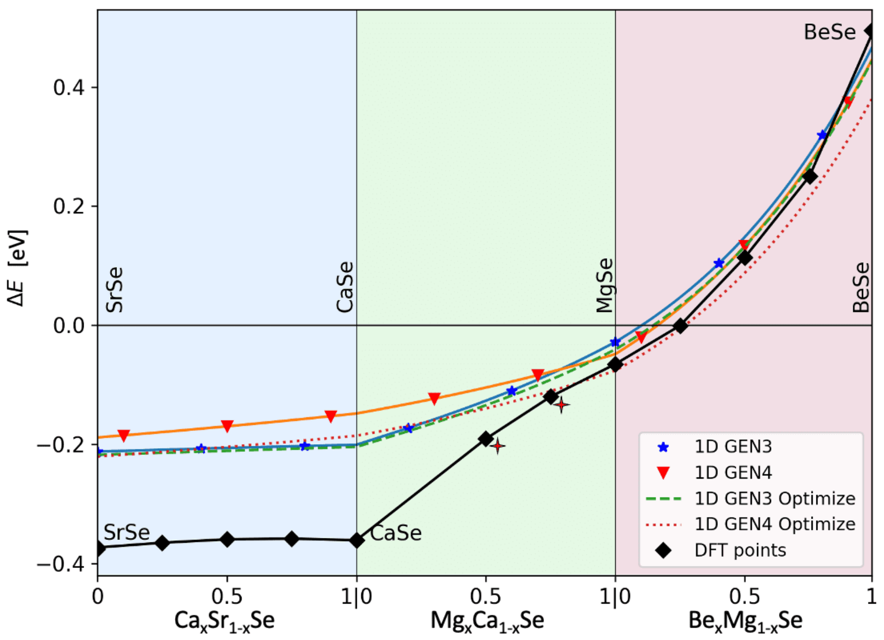


Fig. 2.5 Total energy difference ΔE as a function of concentration(x) for $[Ca_xSr_{1-x}]Se$, $[Mg_xCa_{1-x}]Se$ and $[Be_xMg_{1-x}]Se$ alloys, highlighted in blue, green, and pink regions respectively. Energy differences are predicted using original and optimized 1D descriptors constructed using $GEN3$ and $GEN4$ and verified using DFT (black line with diamond points) within VCA. For an improved accuracy, the two asterisk-highlighted intermediate points in the $[Mg_xCa_{1-x}]Se$ region are calculated using the supercell approach rather than VCA.

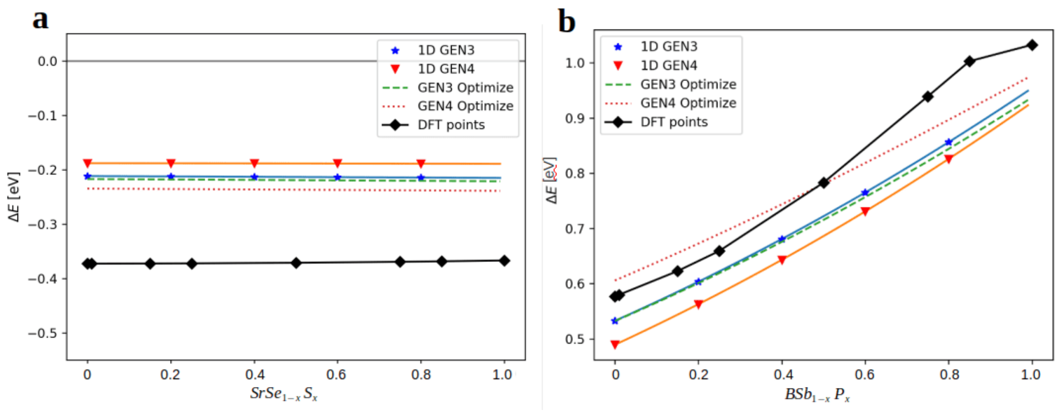


Fig. 2.6 Total energy difference ΔE as a function of concentration (x) for $Sr[S_xSe_{1-x}]$, see panel **a**), and $B[P_xSb_{1-x}]$ alloys, see panel **b**), predicted from original and optimized 1D descriptors constructed using $GEN3$ and $GEN4$. Model predictions are verified using energy differences calculated via DFT[2][3] (black-line with diamond points).

red lines. the smaller the error or, equivalently, the more reliable the prediction. Notably, this is the case when building higher dimension formulas.

In addition, a careful comparison between our results and those reported in the reference paper, Ref.[4], is reported in Table-2 of the Appendix. In particular, in Fig. 2.12 we compared the scatter plot of the 1D formula from $GEN3$ and Ref.[4], with bar graphs of errors for individual compounds. To check the improvement with respect to 1D formulas, we considered the $avg(RMSE)$ value, as also chosen in Ref.[4]. One can observe the improvement in $avg(RMSE)$ if we examine 1D and 2D formulas in Tables 2.1 and 2.4. We notice around 10-20% improvement from the original 1D to 2D, but less than 10% of optimized 1D to original 2D formulas. Furthermore, we also notice that original and optimized 1D formulas from $GEN3$ and $GEN4$ better perform with respect the corresponding 2D ones reported in Ref[4].

We remark that the process of formula optimization is less computationally expensive than the construction of higher-dimensional formulas. In addition, from the formula optimization one can gain a better physical insights about the contribution of individual primary atomic properties. These comments overall suggest that lower-dimensional formulas constitute a better choice in terms of physical interpretation and computational efficiency.

2.3.5 Heat map for 1D, 2D and 3D formulas

Before going on with the discussion, let us just remind that 2D and 3D formulas are constructed starting from 1D formulas. Indeed, it is interesting to take a look at the individual correlation between 1D formulas appearing inside 2D and 3D formulas.

Fig-2.9(a), 2.9(b), 2.10 show the Pearson correlation of different formulas using heat maps for 1D, 2D and 3D formulas, respectively. Here, we chose the "best-performing" formulas, *i.e.* those reported in Tables-2.1, 2.4, 2.5 (including the 1D formula presented in the reference paper [4]). In addition, the correlation of full 1D, 2D and 3D formulas is reported in Fig-2.11.

Let us discuss the individual heatmaps. First, we created a heat map using individual 1D formulas using different generators (cfr Fig-2.9(a)). One can see that all the 1D formulas, including the one in the reference paper[4], show a correlation higher than 97%; this indicates that the formulas found by different generators have a similar dependence as shown in the reference[4], despite having different

Formula	**avg(RMSE)**	**RMSE**	R^2	**Success Rate**	Generator type
$0.113 \times \frac{EA(B)-IP(B)}{r_p(A)^2} - 1.558 \times \frac{\lvert r_s(A)-r_p(B)\rvert}{exp[r_s(A)]} - 0.133$	0.1041	0.0988	0.95	96%	2D descriptor [4]
$-0.342 \times \frac{r_p(B)^3-exp[r_p(A)]}{r_p(A)^3} - 1.042 \times \frac{r_p(A)^2-\sqrt{\lvert r_d(A)\rvert}}{exp[r_p(A)]} - 0.062$	0.0989	0.0944	0.95	89%	$GEN1$
$-0.081 \times \frac{IP(B)+\sqrt{\lvert IP(A)\rvert}}{r_p(A)^3} - 0.001 \times \frac{r_s(A)^3-\sqrt{r_d(A)}}{exp(HOMOKS(A))} - 0.062$	0.1163	0.1100	0.93	86%	$GEN2$
$-1.175 \times \frac{r_p(A)-\sqrt{\lvert r_d(A)\rvert}}{r_s(B)^3+r_p(A)^3} + 0.513 \times \frac{r_s(B)+\sqrt{\lvert r_p(B)\rvert}}{r_p(B)^3+r_s(A)^3} - 0.250$	0.0911	0.0878	0.96	87%	$GEN3$
$0.618 \times \frac{r_d(A)/r_p(B)}{r_p(A)^3 \times \sqrt{r_d(A)}} + 1.097 \times \frac{r_p(A) \times \sqrt{\lvert r_p(B)\rvert}}{r_p(B)^3+r_p(A)^3} - 0.384$	0.0995	0.0955	0.95	92%	$GEN4$

Table 2.4 2D formulas, along with related statistics. Notation as in table-2.1. RMSEs are in eV.

Formula	**avg(RMSE)**	**RMSE**	R^2	**Success Rate**	Generator type
$0.108 \times \frac{EA(B)-IP(B)}{r_p(A)^2} - 1.806 \times \frac{\lvert r_s(A)-r_p(B)\rvert}{exp[r_s(A)]} - 3.782 \times \frac{\lvert r_p(B)-r_s(B)\rvert}{exp[r_d(A)]} - 0.023$	0.0818	0.0756	0.97	93%	3D descriptor [4]
$0556 \times \frac{r_p(B)^3-exp[r_p(A)]}{r_p(A)^3} + 0.364 \times \frac{r_p(A)^2-\sqrt{\lvert r_d(B)\rvert}}{exp[r_p(A)]}, -0.124 \times \frac{r_p(B)^2-\sqrt{\lvert r_d(A)\rvert}}{r_p(A)^3} - 1.87$	0.1003	0.0933	0.95	90%	$GEN1$
$-0.056 \times \frac{(LUMOKS(A)+HOMOKS(B))}{r_p(A)^3} + 0.266 \times \frac{\sqrt{\lvert EA(B)\rvert}+exp(EA(B))}{r_s(A)^3} - 0.016 \times \frac{HOMOKS(A)-exp(LUMOKS(B))}{(r_p(A)^3)} - 0.310$	0.1300	0.1205	0.92	91%	$GEN2$
$-0.885 \times \frac{r_p(B)-exp[r_p(A)]}{r_p(A)^2+r_p(A)^3} - 0.417 \times \frac{r_s(A)-exp[r_s(B)]}{r_s(A)^3+r_p(B)^3} - 0.579 \times \frac{r_p(A)-\sqrt{\lvert r_d(A)\rvert}}{r_p(B)^2+r_s(A)^3} - 0.616$	0.0875	0.0834	0.96	98%	$GEN3$
$0.635 \times \frac{\sqrt{IP(B)}/\sqrt{IP(A)]}}{r_p(A)^3+r_p(B)^3} + 0.730 \times \frac{r_p(B) \times \sqrt{\lvert r_d(A)\rvert}}{r_p(A)^3+r_p(B)^3} + 0.038 \times \frac{IP(A)^2-EA(A)^2}{exp(r_p(A)) \times exp(r_d(B))} - 0.358$	0.0989	0.0919	0.96	93%	$GEN4$

Table 2.5 3D formulas, along with related statistics. Notation as in table-2.1. RMSEs are in eV.

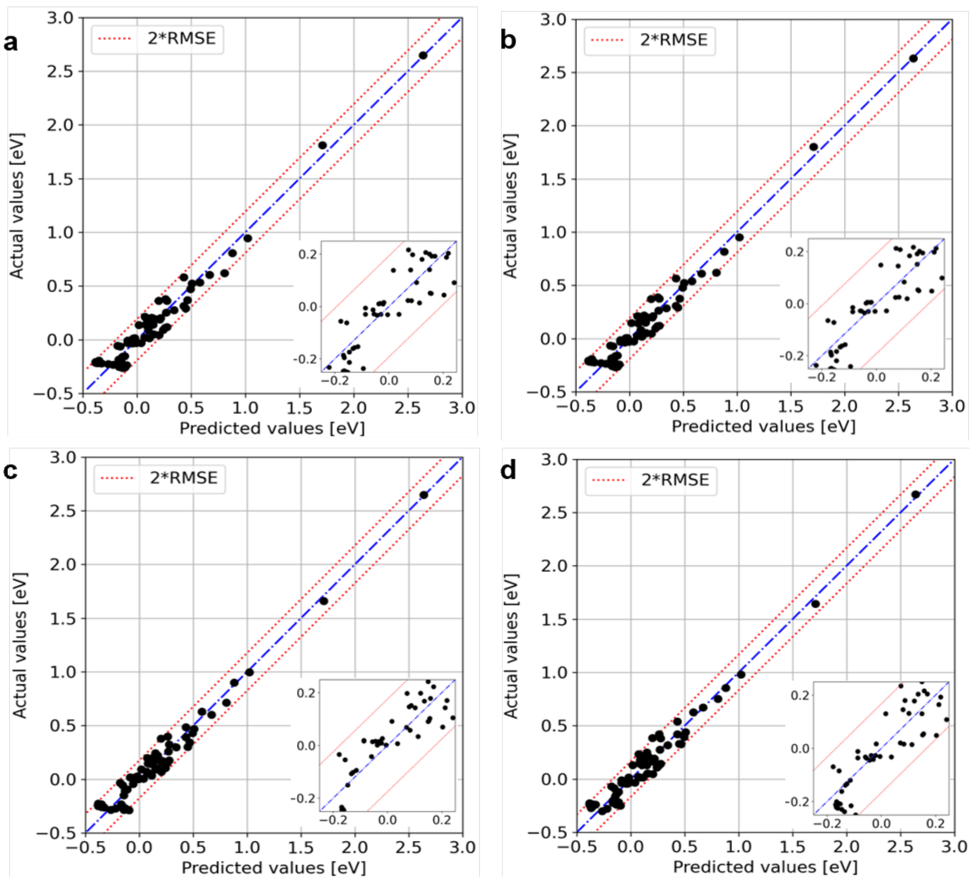


Fig. 2.7 Comparison of actual (*i.e.* DFT) vs predicted total energy difference ΔE for **a)** 1D, **c)** 2D and **d)** 3D formulas constructed using *GEN*3. Panel **b)** shows the best 1D descriptors after formula optimization. Lower-right insets show a zoom in the relevant region where many compounds are concentrated. Red dotted lines correspond to $2\times avg(RMSE)$ value. The respective descriptors can be inferred from tables-2.1, 2.2, 2.4, 2.5

$$\underbrace{-0.081 \times \overbrace{\boxed{\frac{IP(B)+\sqrt{|IP(A)|}}{r_p(A)^3}}}^{G21} - 0.001 \times \overbrace{\boxed{\frac{r_s(A)^3-\sqrt{r_d(A)}}{\exp(HOMOKS(A))}}}^{G22} - 0.062}_{2DG2}$$

Fig. 2.8 Nomenclature used in the heatmaps, taking as example a 2D formula constructed using two 1D formulas and Gen-2. Here, *G* represents the generator number followed by two numbers (1, 2 or 3): GIJ refers to the Jth 1D-formula used by Generator I when constructing higher dimensional formula. The complete formula can be then denoted as 2DG2 (*i.e.* 2D formula using generator 2).

RMSEs. On the other hand, the unique 1D formulas used in 2D and 3D show smaller correlations, compared to 1D, for different generators (cfr Section:2.2.2). Here, we would like to mention that we prevented our algorithm to construct higher dimensional features using 1D formulas that showed a correlation higher than 90%.

The nomenclature used in the heatmaps 2.9(a), 2.9(b), 2.10, 2.11 is shown in Fig-2.8, taking as example a 2D formula. A similar nomenclature can be extended to the case of 3D formulas. In fig-2.11, we report the correlation among different complete formulas, where we can see at-least 96% correlation between different formulas. This shows that most of the hints are already catched by the 1D features, which are quickly evaluated. The red boxes inside Figs. 2.9(b) and 2.10 indicate the “internal" correlation, *i.e.* the correlation between 1D formulas used in the construction of the 2D or 3D formulas. We note that in some cases the correlation within the red box is of the order of 85 %, signalling that using two 1D formulas in the 2D or 3D formulas is somehow “redundant".

2.3.6 Comparison with the reference 1D Formula

If we compare the error for individual binaries achieved by *i*) the method mentioned in section-2.2 and *ii*) reference-[4], one can visualize the performance of the different formulas. Fig-2.12 reports the error comparison of 1D formula constructed using Gen-3 (2.12 **b,d**) and 1D formula presented in the publication[4](2.12 **a,c**)). In addition to the predicted values, we show values corresponding to 2*RMSE using red lines, indicating the errors’ distribution. As such, a wider distance between two

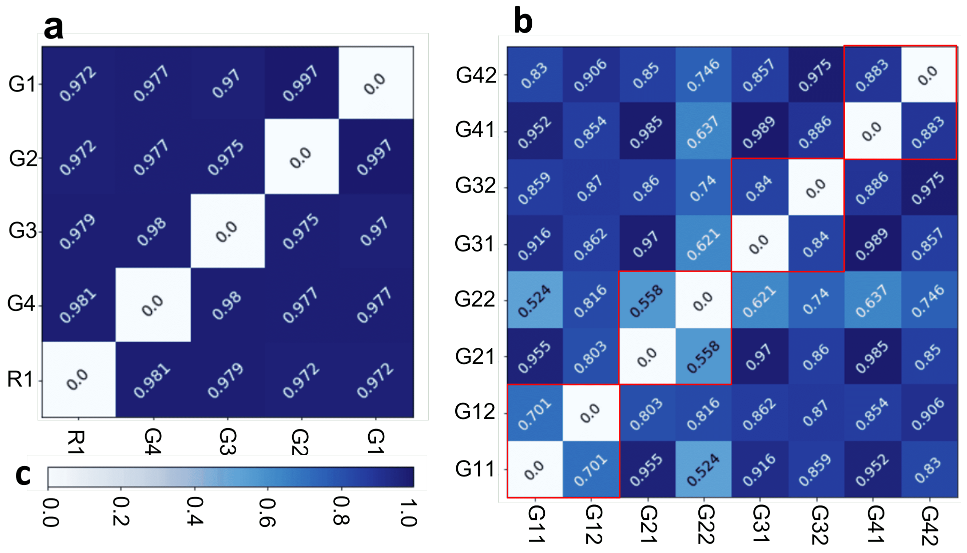


Fig. 2.9 Heatmap of Pearson correlation coefficient between different 1D (**a**) and elements of 2D (**b**) formulas, using different generators. (**c**) shows the colour bar for the heatmap. The nomenclature reported on the axes labels follows Fig-2.8. The red square inside panel (**b**) shows the correlation between 1D formulas inside each 2D formula. R1 represents the 1D formula obtained from the reference paper-[4].

red lines indicates a higher error and vice versa. We can see the improvement in our predicted 1D formula compared to the reference publication. From Fig-2.12 (**c,d**) we report the performances for different compounds of our predicted formula and the 1D formula by Ghiringhelli *et al.* Here, not just the overall RMSE is reduced by 40% (cfr panels a,b), but also the absolute error for individual binaries seem to be far more smoothly distributed.

2.3.7 Formula optimization using automated optimization methods

One may wonder the following: we are using the grid search method for optimization, but why one cannot use other automated optimization methods to find the relative contribution of individual atomic features in the material feature? To answer this question, we further benchmarked our grid-search optimization for the best 1D formula constructed using GEN3, defined in equation 2.9, by means of other automated optimization methods: Nelder-Mead[113], Conjugate Gradient(CG)[114], Broyden–Fletcher– Goldfarb– Shanno(BFGS)[115] and truncated Newton(TNC)[116]:

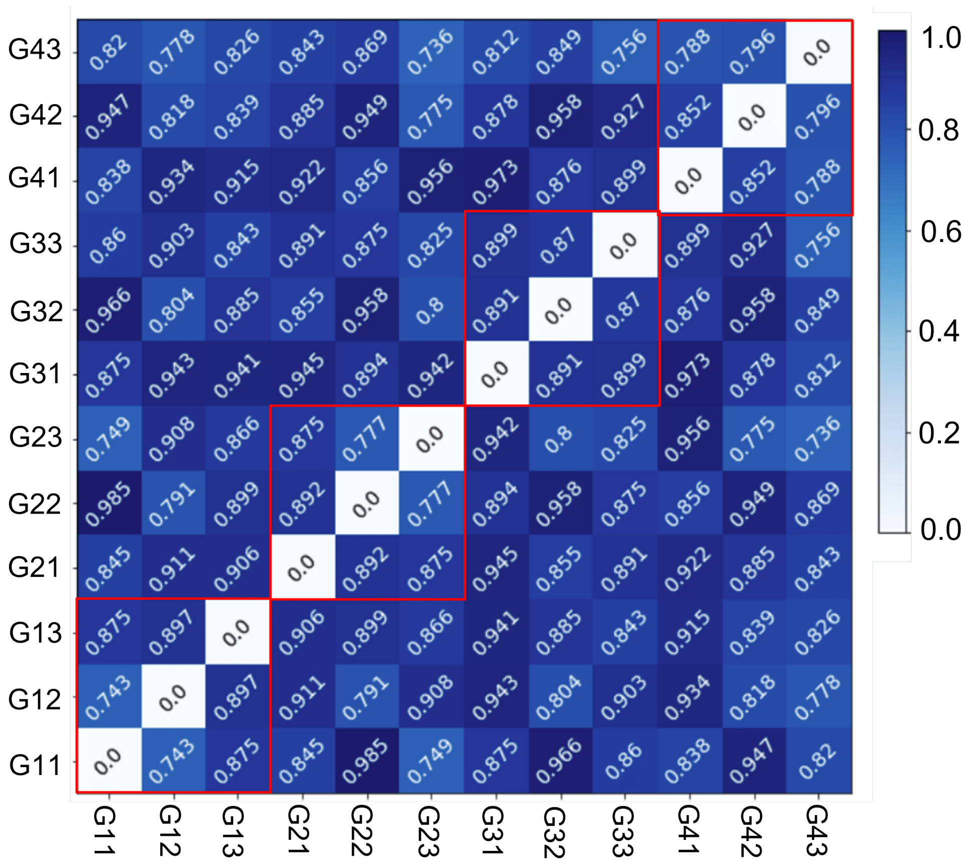


Fig. 2.10 Heatmap of Pearson correlation coefficient between different elements in 3D formulas using different generators . The nomenclature reported on the axes labels follows Fig-2.8. The red squares show the correlation between 1D formulas inside each 3D formula.

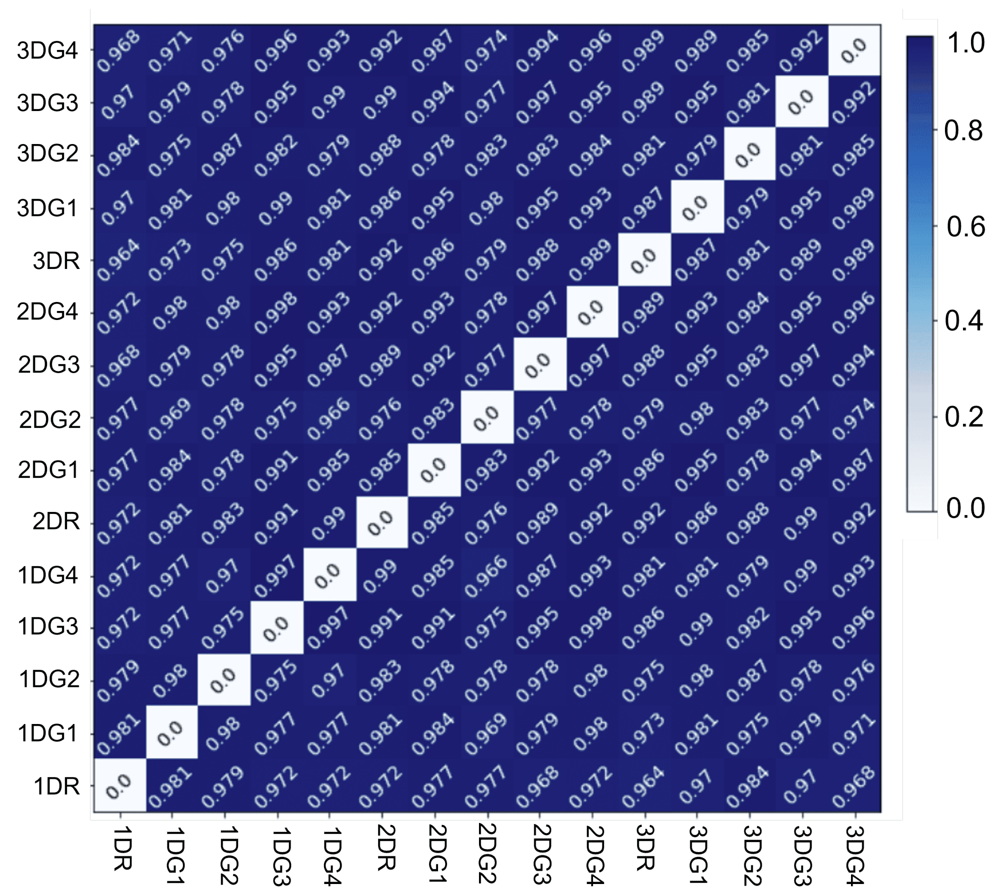


Fig. 2.11 Heatmap of Pearson correlation coefficient between different 1D, 2D and 3D complete formulas. The nomenclature for the axis label follows Fig-2.8. 1DR, 2DR and 3DR denote the 1D, 2D and 3D formulas as published in the reference-paper [4], respectively.

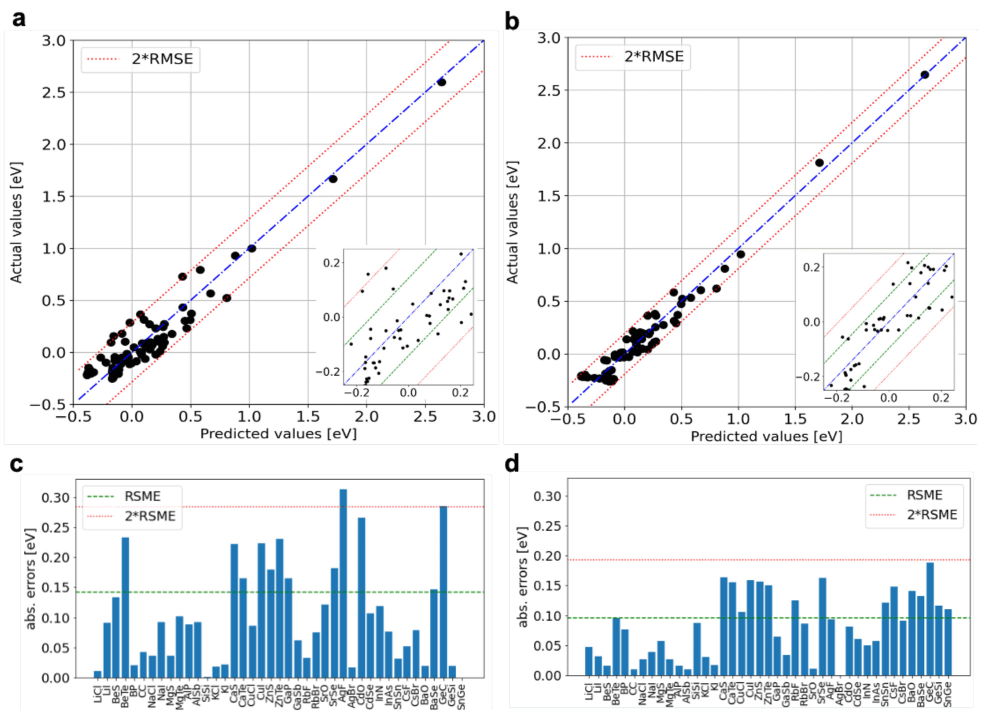


Fig. 2.12 Panel **a** reports the predicted against actual (*i.e* DFT) ΔE values for 1D formula presented in Ref. 4; panel **b** reports those obtained by *GEN*3 as a comparison. Absolute error for the same formula for different compounds in the bar graph (panels **c** and **d**). The related formulas used to calculate the values can be inferred from the main text.

$$\frac{r_p(B) + \sqrt{|r_d(A)|}}{r_p(A)^3 + r_p(B)^3} \tag{2.9}$$

if we introduce different coefficients (a,b,c,d) multiplying each atomic feature, then the expression can be written as:

$$\frac{a \times r_p(B) + b \times \sqrt{|r_d(A)|}}{c \times r_p(A)^3 + d \times r_p(B)^3} \tag{2.10}$$

Through the automated optimization methods mentioned above, one can obtain the set of coefficients that give the lowest RMSE. Each step of the optimization is followed by LR. Thus, the optimization modifies the slope (m) and intercept moving towards the coefficients combination with the lowest possible RMSE. From 80 to 100 iterations are needed for each method to reach the minimum RMSE, as reported in panel **a** of figure 2.13. Panels from **b** to **e** report the trend of the ratios $a/b, c/d, m \times a/c$ and $m \times b/d$. All these quantities appear to converge to constant values in the final steps of the optimization. We ran all the methods mentioned above for the top ten performed 1D descriptors to verify the precision of the grid search technique. We found that, in all forty cases, the grid search acquired variables show the lowest RMSE.

2.3.8 Comparison with deep neural network

This section will discuss some drawbacks of using an advanced method like NN. As shown in Fig-2.14, we are comparing fundamental ideas about the NN and formula search. The introduction-2.1 explains why the neural network works like a total or partial BlackBox. Indeed, the properties or features the NN uses are hard to control and verify. In addition, it becomes very troublesome if we want to get some physical insights out of NN. Therefore, our aim here is to use ML algorithms that can provide output properties and scientific insights. The method shown in this chapter is an excellent example of transparent ML techniques. Here, as we can see in Fig-2.14(bottom path), ML not just provides a prediction, but also the path to interpret the physical ingredients underlying the specific prediction.

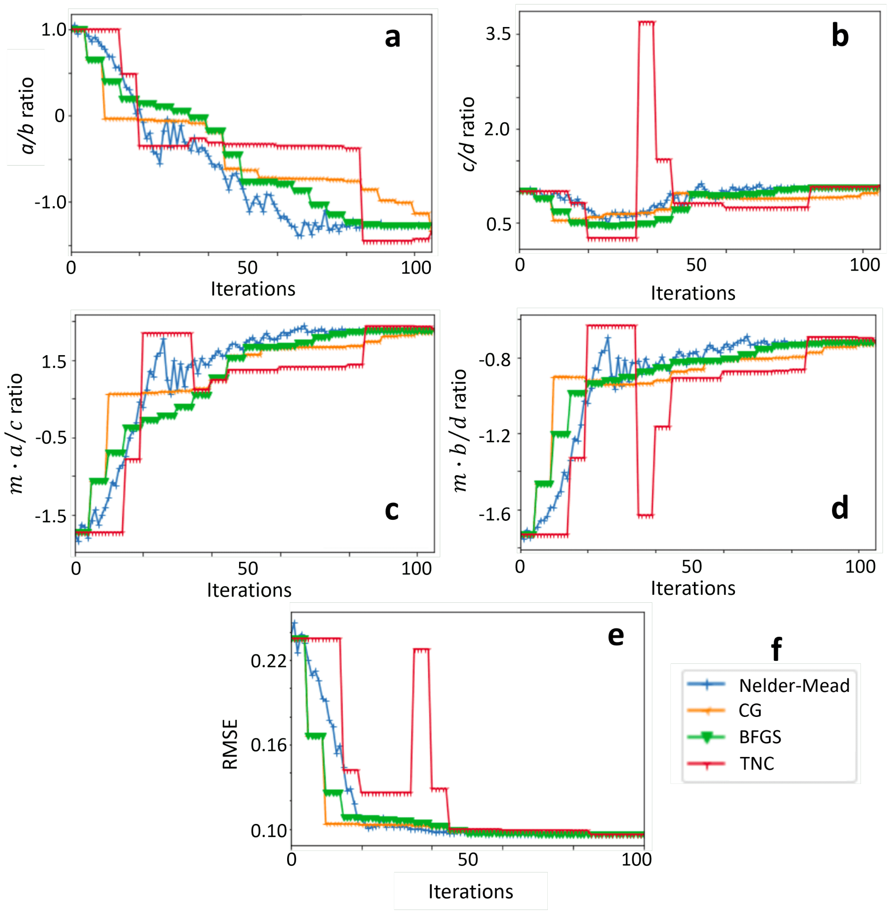
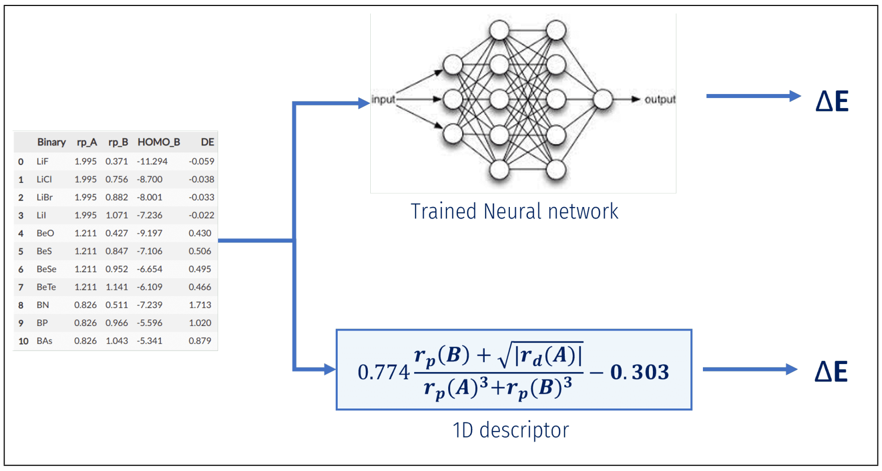


Fig. 2.13 Evolution of different parameters at each minimization step (i.e. iteration) using different automated optimizing algorithms: Nelder-Mead (Blue lines), CG (orange lines), BFGS (green lines) and TNC (red lines) as shown in **f**. Here, we show the evolution of RMSE (eV), $a/b, c/d, m \times a/c$ and $m \times b/d$ in panels **a, b, c, d** and **e** respectively. Here, to show the evolution of different parameters, we use a GEN3 formula $\left((r_p(B) - \sqrt{|r_d(A)|})/(r_p(B)^3 + r_p(A)^3)\right)$. The ratios for grid search are -1.28, 1.00, 0.88, -0.67 for a/b, c/d, $m \times a/c$ and $m \times b/d$ respectively.



	Binary	rp_A	rp_B	HOMO_B	DE
0	LiF	1.995	0.371	-11.294	-0.059
1	LiCl	1.995	0.756	-8.700	-0.038
2	LiBr	1.995	0.882	-8.001	-0.033
3	LiI	1.995	1.071	-7.236	-0.022
4	BeO	1.211	0.427	-9.197	0.430
5	BeS	1.211	0.847	-7.106	0.506
6	BeSe	1.211	0.952	-6.654	0.495
7	BeTe	1.211	1.141	-6.109	0.466
8	BN	0.826	0.511	-7.239	1.713
9	BP	0.826	0.966	-5.596	1.020
10	BAs	0.826	1.043	-5.341	0.879

$$0.774\frac{r_p(B)+\sqrt{|r_d(A)|}}{r_p(A)^3+r_p(B)^3}-0.303$$

Fig. 2.14 Conceptual visualization of approaches based on Neural Network (top) or Formula search method (bottom).

2.4 Summary

The knowledge of a material stable crystal structure constitutes the starting point for any ab–initio modelling, since materials properties crucially depend on the periodic atomic arrangement in the crystal. Within this general framework, our aim here has been to exploit ML methods to correlate the energetic stability of different crystal structures (zincblende vs rocksalt) for popular binary semiconducting compounds with primary properties of their atomic constituents, the latter representing simple and easily-accessible ingredients. Based on atomic properties, we therefore built the material features using a combinatorial approach, we trained the machine learning model using the created features over a density–functional–theory dataset and we obtained simple mathematical expressions to quantitatively predict the energetic stability of one crystal structure over the other (i.e., a formula). In addition, we have also introduced an extra step following the linear regression to explore the relative contributions of individual basic atomic properties.

To investigate the performance of the combinatorial approach, we compared our results with a reference paper [4], where the authors predicted the stability of the crystal structure using an automated feature selection method. We found that our 1D formulas constructed using the combinatorial approach achieved a higher accuracy with respect to the reference ones. Furthermore, we also learned more about the underlying mechanism from the formula optimization, where we found that the stability of RS and ZB heavily depends on the r_p radius of A-sites. This kind of understanding is, in general, much more difficult to achieve in heavily-automated artificial–intelligence methods, such as neural networks, where it is not possible to interpret directly the model results. In this respect, our approach based on linear regression allows the construction of physical models supported by machine-driven suggestions of relevant ingredients; as such, it should be regarded as a methodology offering a huge range of applications in addressing microscopic mechanisms underlying different phenomena, calling for extensive investigations in the nearby future.

Chapter 3

Optimizing the Design and Performance of Magnonic Devices through Modelling and Simulation Techniques Enhanced by Machine Learning Algorithms

3.1 Introduction

When looking at the history of computation, we can see that the first fully transistor-based computer was built in 1955 and replaced vacuum tubes. Since then, the transistor size has decreased drastically, therefore providing room to place more and more transistors in the circuit. As such, the number of transistors in the CPUs has been on a constant rise, improving the performances of the CPUs to meet the ever-rising demand for computation power. In 2006, CPUs were manufactured with around 42 million transistors; after four years, in 2010, they were manufactured with 295 million transistors. This number increased to more than 1 billion transistors in 2013; in 2018 a 16nm CPU used 1 billion transistors. Moore's Law, stating the number of transistors on a chip to double every two years, has therefore proven to be rather accurate so far. However, two significant problems started rising along with the smaller size of the traditional silicon-based transistor, which will be discussed

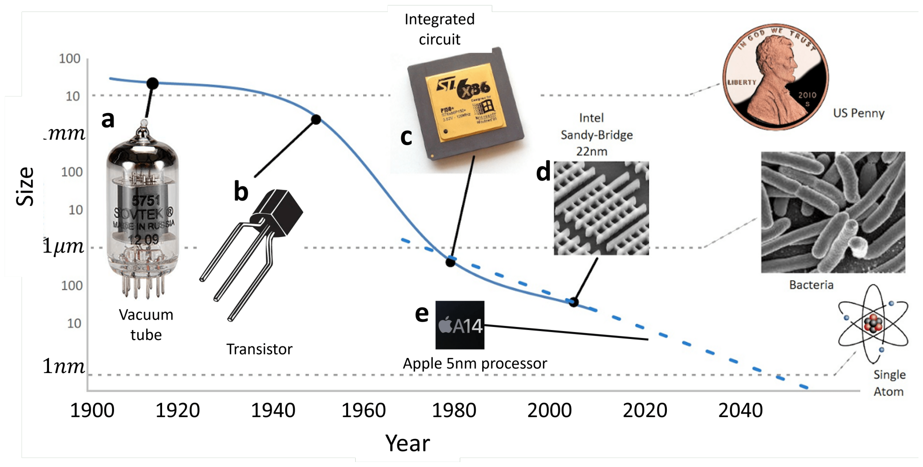


Fig. 3.1 Evolution of logic devices from Vacuum tubes (a) to integrated circuits (c) to high efficiency processors (e). The figure is reproduced using the ref-[5].

below. For a deeper understanding, we will refer to the human biological brain, in comparison with logical devices.

Let's go back to the two significant problems occurring with a traditional transistor-based device: heat dissipation and energy consumption. Let's consider the average life of SOC as ten years and a PC with 1 billion transistors as our standard SOC. The energy consumption for this processor would be around 95-125W, which accumulates to 11kW when considering 100 billion transistors, heat dissipation summing up to $11kW \times 3.41 = 37.5kBTU/h$. Let's now compare this to the human brain, which contains around 100 billion neurons working as processing units. At variance with the PC case, the energy consumed by the brain would be close to 20W[126], and heat dissipation close to $0.072kBTU/h$. From the above comparison, we can see that even though we are close to biological processing in terms of number of processing units, we still need to make significant progresses in terms of energy- and heat-efficient devices. On the positive side, advancements in technology have resulted in smaller transistor sizes, which consume less energy to switch. Currently, the trend in transistor size reduction has nearly reached its limits, as evidenced by the 4nm gap layer between p-type and n-type semiconductors, as depicted in Figure 3.1(e). However, further downscaling will result in increased Joule heating in interconnects, leading to higher energy consumption by CPUs. Moreover, even minor imperfections in the transistors can potentially disrupt their functioning.

At present, we cannot replace the traditional silicon doped transistor, due to its reliability[127]. Therefore, in recent years, computer specialists have tried to distribute the workload on the CPUs by exploiting the GPUs and TPUs for different processing tasks[128]. Nevertheless, the core components of these units are traditional transistors with specific types of structuring, which come with similar problems to those mentioned above. Therefore, it is of interest to propose processing units that can assist the traditional transistor-based SOCs, to perform tasks - like neuromorphic computing - to help CPUs to distribute loads. It is in this broad context that the studies presented in this chapter could find application. Graphics processing unit

Magnonics is an emerging field of complex magnetism, in turn one of the most fascinating areas of solid-state physics. Magnonics combines the study of waves and magnetism. In closer detail, a spin wave relates to spin-excitations and can be defined as a propagating disturbance in an ordered magnetic material; the wave propagates via rotation of magnetic moments. The field of magnonics offers a new type of low-power information processing in which magnons (*i.e.* the quanta of spin waves) - instead of electrons - carry and process data. As such, no electrical charge motion is involved, in principle dissipating no heat. In recent years, scientists have been exploiting different characteristics of magnons, such as magnetic anisotropy control using voltages [129], spin Hall effect [130], spin orbit torques [131], the propagation of magnetic droplets [132] and the creation of a nanoscale wake-up receiver using spin waves [133]. In addition to that, other magnetic-based technologies are also getting popular; for example, MRAM can store data with lower switching time/energy than traditional memories [134] and spin valves can be used for neuromorphic computing[135].

Here, the aim is to propose a magnonic device that could perform a particular task of NN (like filtering images), reducing the computation load on the CPU. This is motivated by the extensive use of magnonic devices, that has come to light in recent years [39]. Tan et al.[136] suggested the multi-sensory neural network with cross-modal integration, inspired by biological objects such as eyes, ears, etc. However, the development of each device requires complex investigations, and usually one device design is suitable for one function only[136].

Demultiplexer. The spin waves have wavelengths ranging from micrometers down to atomic scales[137] and show a variety of pronounced nonlinear phenomena[138];

as such, they look promising for Boolean computing[139], radio frequency applications [140] and neuromorphic computing[141] at the nanoscale. Recently, in the field of photonics (i.e. operating with light waves to process data) the approach of inverse-design devices[142] was proposed. This approach was adopted by Wang et al. [6] to create and demonstrate a demultiplexer, which could separate two different spin-wave frequencies using a material-based structure. We considered this device as our starting point, which we used to construct more complex devices using state of the art machine learning and optimization methods. To this end, we perforned micromagnetic simulations by means of the GPU-accelerated simulation package MuMax3[143], including both exchange and dipolar interactions, to calculate the space- and time-dependent magnetization dynamics in the investigated structures. After the simulation phase carried out in this thesis, in the longer term, the experimental group in Aalto University (Finland), led by Prof. Sebastiaan van Dijken, will consider experimentally fabricating the device, to cross-check the theoretical preditcions and test the relevance of the proposed device from the technological point of view.

The design of the demultiplexer is shown in Fig.3.2. Here, the magnonic frequency demultiplexer consists of one input waveguide and two output waveguides with a width of 300 nm connected by a $1 \times 1\mu m^2$ design region. The design region has been divided into 10×10 elements, each with a size of $100 \times 100nm^2$. The yellow region indicates the magnetic material (Yttrium Iron Garnet, YIG), and the white holes denote regions where the material has been removed. Here, spin waves react differently depending on the arrangement of the defects ; therefore, our aim is to find the combinations of holes (defects), for which the frequency splitting is the highest. In this case, we note incidentally that a traditional trial and error method will not work, due to many possible combinations. In fact, the possible combinations are 2^{100}, which convert to 1.26×10^{30}. If we consider 10 seconds to perform one simulation, it will take around 4×10^{23} years to finish the sample design ! Therefore, we must use innovative, novel machine-learning techniques to find the optimum structure. To this aim, the authors in [6] used the Binary search method, an approach which is easy to implement and to understand. Therefore, first, we tried to simulate the stripe to check the frequencies and wavelength for the spin waves; after that, we have implemented the Binary search algorithm, as suggested in our reference article[6]. Then, we have implemented a more efficient method to find the optimum structure. At last, we used our in-house implemented method to construct a more

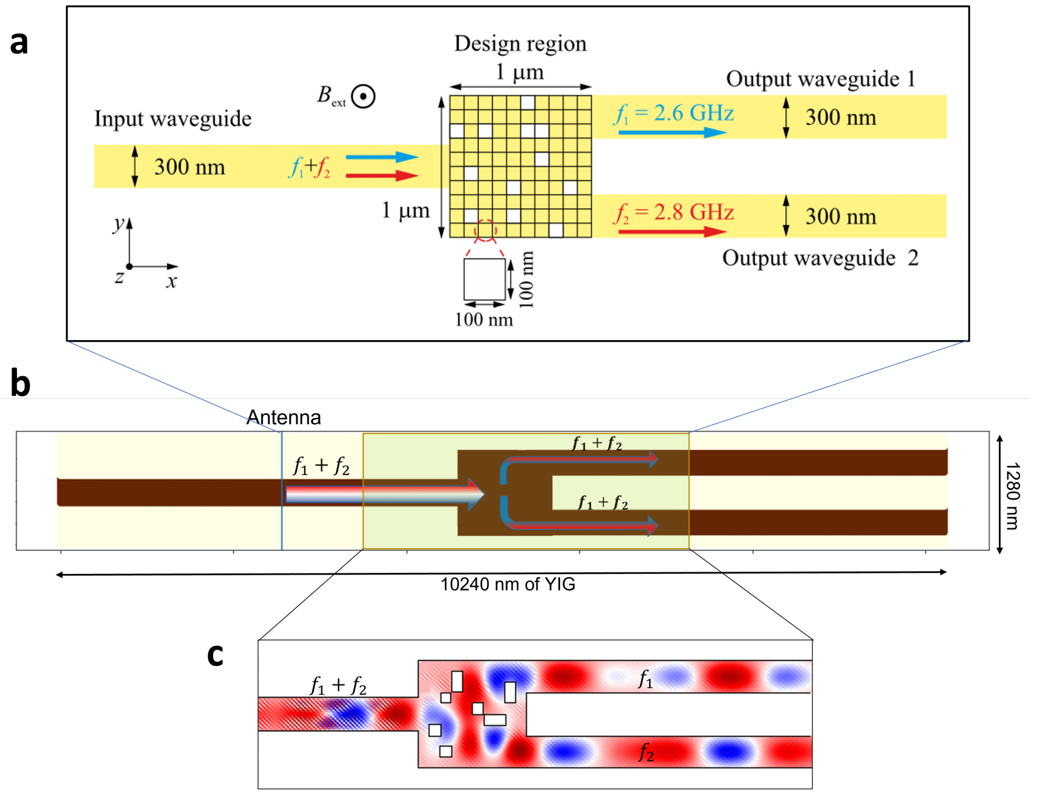


Fig. 3.2 a) Structure for the design of the magnonic frequency demultiplexer suggested by Wang et al.[6]. b) Complete structure of the demultiplexer device. C) Visualization of spin waves inside the demultiplexer.

complex device. We would like to mention that the research activity presented in this chapter is performed in collaboration with the group led by Prof. Sebastiaan Van Dijken at Aalto University.

3.2 Methodology and Technical details

3.2.1 MuMax3 code

MuMax3 is an open-source software package for micromagnetic simulations, used to model the behavior of magnetic materials at the micro and nano scale. It uses the Landau-Lifshitz-Torque(as referred in the equation-3.2.1) [143] equation to simulate the dynamics of the magnetization and can be used to study a wide range of phenomena, such as magnetic domain formation, spin wave excitations, and magnetization reversal. It is written in the programming language Go and can run on a variety of platforms, including Windows, Mac, and Linux[143]. It is capable of handling large simulation cells, and has support for multi-GPU acceleration which helps to speed up the simulations. The Landau-Lifshitz-Torque[143] can be written as follows;

$$\vec{\tau}_{\mathrm{LL}} = \gamma_{\mathrm{LL}} \frac{1}{1+\alpha^2} \left(\vec{\mathbf{m}} \times \vec{\mathbf{B}}_{\mathrm{eff}} + \alpha \left(\vec{\mathbf{m}} \times \left(\vec{\mathbf{m}} \times \vec{\mathbf{B}}_{\mathrm{eff}} \right) \right) \right)$$

with γ_{LL} the gyromagnetic ratio (rad/Ts), α the dimensionless damping parameter and $\vec{\mathbf{B}}_{\mathrm{eff}}$ the effective field (T). The default value for γ_{LL} can be overridden by the user. $\vec{\mathbf{B}}_{\mathrm{eff}}$ has the following contributions:

- externally applied field $\vec{\mathbf{B}}_{\mathrm{ext}}$
- magnetostatic field $\vec{\mathbf{B}}_{\mathrm{demag}}$
- Heisenberg exchange field $\vec{\mathbf{B}}_{\mathrm{exch}}$
- Dzyaloshinskii-Moriya exchange field $\vec{B}_{\mathrm{dm}}$
- magneto-crystalline anisotropy field $\vec{\mathbf{B}}_{\mathrm{anis}}$
- thermal field $\vec{\mathbf{B}}_{\mathrm{therm}}$.

MuMax3 also provides a wide range of post-processing tools to analyze the results of the simulation, such as visualization of the magnetic structure and the calculation of various physical quantities (*i.e.* magnetic energy, magnetization, and magnetic susceptibility). It also includes a scripting interface that allows users to automate the simulation process and perform parameter sweeps.

It is used in a variety of research fields such as spintronics, magnetic data storage, and microwave engineering, and has been applied to the study of magnetic materials such as thin films, multilayers, and nanoparticles[143].

MuMax3 is actively developed and maintained by the group of Prof. M. V. Costache at the Hasselt University, Belgium and is distributed under the GNU General Public License. The software can be downloaded and used for free, and the source code is available on Github[143].

3.2.2 Micromagnetic simulations of demultiplexer

All the simulations were done using the MuMax3 micro-magnetic package, as mentioned in section-(Section-2.2). In this case, our first task was to replicate the computational experiment that Wang *et al.* performed. Therefore, for simple verification, we considered the single stripe with $1\mu m$ in length and $300nm$ in width, to verify the wavelengths mentioned in the reference article[6].

As a second task, we simulated the entire device where the simulation box size is the $10240 \times 1000nm^2$ with a cell size of $20 \times 20 \times 20nm^3$. We set the saturation magnetization at $14 \times 10^4 A/m$, the exchange stiffness at $3.5 \times pJ/m$, and the damping constant at 2×10^{-4} . An external magnetic field of 200 milli-Tesla (mT) is being imposed and the direction of this field is along the y-axis. This simulation was run for $100ns$. Here, we considered the YIG material for the spin wave propagation and vacuum for the holes. In addition, we have also created a $0.5\mu m$ wide waveguide on the left and right side of the device with higher damping parameters to prevent the reflection of the spin waves. The entire device is shown in Fig.3.2(b), highlighting the middle region and crop-out section.

3.3 Results and Discussion of different ML methods

3.3.1 Binary Search Method

Methodology

The binary - or half-interval - search is an efficient algorithm for finding a specific item from a sorted list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until the possible locations have been narrowed down to just one. In this case, first, we represented our $1\mu m \times 1\mu m$ by means of a two-dimensional array; $M_0(m,n)(m \in [1;10], n \in [1;10]$. This array was optimized to maximize the following objective function:

$$O = (I_{f_1,o_1} - I_{f_1,o_2})(I_{f_2,o_1} - I_{f_2,o_2}) \tag{3.1}$$

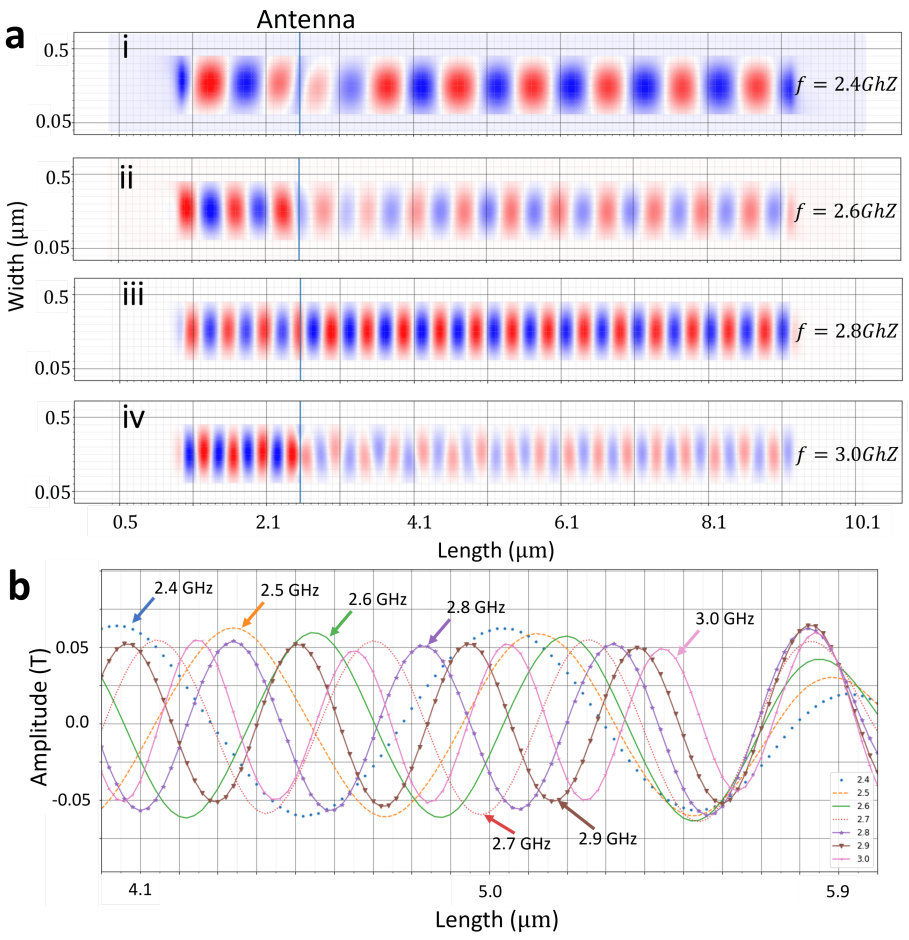


Fig. 3.3 Simulations of different stripes of YIG to calculate the wavelengths associated to different frequencies. The size of the stripes is $10\mu m \times 0.4\mu m$. The vertical blue line in each of the stripes indicates the place of the antenna that we have used to excite the spin waves. The different frequencies are shown in the legend. Red and blue regions show the spin wave amplitude (high and low) in the Z direction. Here, panel a) shows the spin waves for different frequencies and panel b). shows the spin wave amplitude in the central section of the stripes for different frequencies.

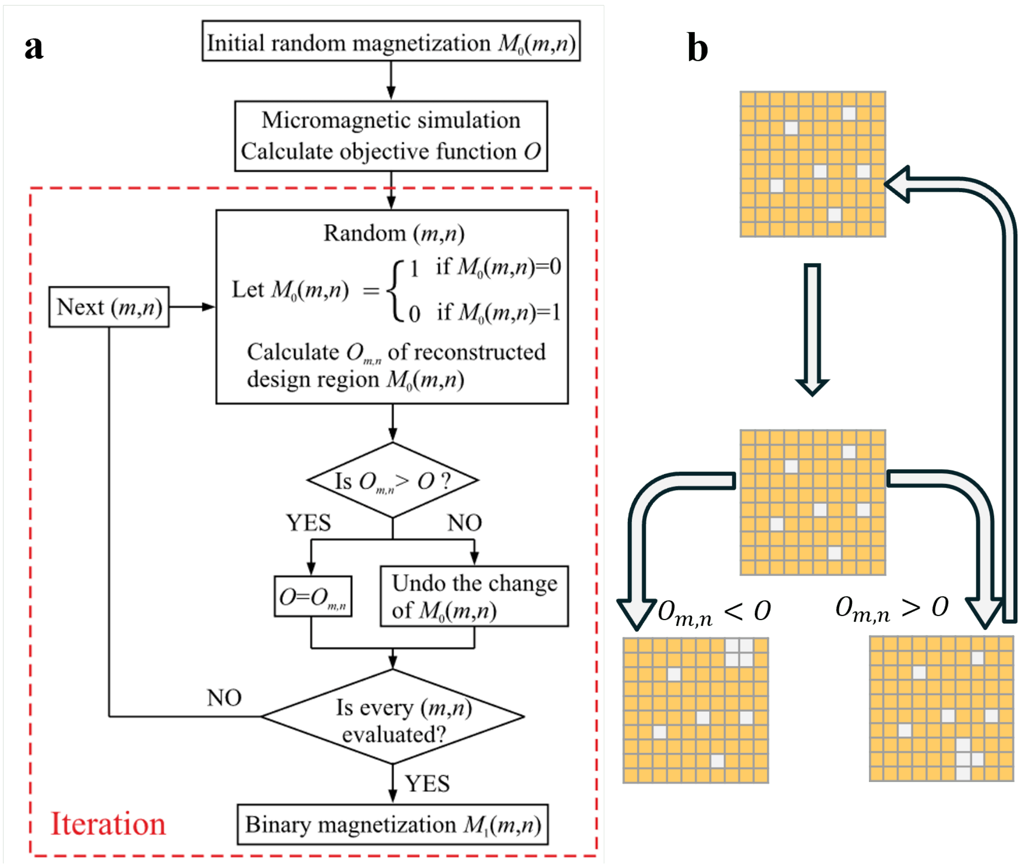

a
Initial random magnetization $M_0(m,n)$
Micromagnetic simulation
Calculate objective function O
Random (m,n)
Let $M_0(m,n) = \begin{cases} 1 & \text{if } M_0(m,n)=0 \\ 0 & \text{if } M_0(m,n)=1 \end{cases}$
Calculate $O_{m,n}$ of reconstructed design region $M_0(m,n)$
Next (m,n)
Is $O_{m,n} > O$?
YES
NO
$O=O_{m,n}$
Undo the change of $M_0(m,n)$
NO
Is every (m,n) evaluated?
YES
Iteration
Binary magnetization $M_1(m,n)$
b
$O_{m,n} < 0$
$O_{m,n} > 0$

Fig. 3.4 (a) Algorithm of the Binary search method published in the paper-[6]; (b) schematic explanation, using different structures. The detailed explanation is provided in Section-3.3.1

where I_{f_i,O_j} is the spin-wave intensity of frequency f_i at the $j-th$ output waveguide $(i, j = 1,2)$. We estimated each value individually, and only positive values were accepted. This objective function aimed at maximizing the transmission of the spin wave of frequency $f_1(f_2)$ to the output waveguide 1 (2) and simultaneously minimizing the crosstalk between them, i.e., to realize the function of a demultiplexer.

The Binary search algorithm is an optimization method that sequentially evaluates every element of the design region. First, we select a random structure with holes and material inside the design region. After that, we set the initial saturation magnetization and other parameters needed for the MuMax3 simulations. We obtained the ground state of the device by relaxation of magnetization. Subsequently, the spin-wave propagation in the structure is simulated after relaxation, to calculate the spin-wave intensity in the output waveguides. In the structure, we put the vertical antenna in the middle of the input waveguide, as shown in Fig-3.4 to excite the spin waves. In the device, the wave will propagate through the input waveguide, then to the design region with material and holes and, finally, to the output waveguides. In the end, we check the intensities of two frequencies at two output waveguides to calculate the objective function-3.1.

We then repeat this process with a new structure with random combinations of indices (m,n). We apply these new indices into the device's design region part, where the material properties will be set according to either YIG or vacuum. Again, we calculate the objective function $O_{m,n}$ for the new combination. If the value of the new function are $O_{m,n} > O$, then we accept the new structure and the value of O is replaced by $O_{m,n}$ for the next iteration. Otherwise, we keep O as reference and proceed from there on. We repeat this process until no further improvement is achieved, as Wang et al. suggested in their article [6].

Results

The advantage of the binary search method is that it is easy to implement and converges quickly to find the minima for various linear, nonlinear, and nonreciprocal magnonic devices, as shown by Wang et al. [6] for different simulation cases. In our simulations, we found the same structure as the authors of the reference paper [6] when we started from the given initial structure; however, the number of steps we required was close to 800, to be compared to 400 that was required by the authors[6].

From this study, we learned some valuable insights:

- The final structure is highly dependent on the initial structure in the BS.
- The binary search algorithm is only saving information about the last structure; as such, there is a high probability to be stuck in local minima.
- We are not exploiting any knowledge from the rejected structures in BS, which could anyway provide insights.

The above-mentioned observations also possibly happened due to a different implementation of the BS algorithm or of the objective function, since the authors didn't provide any detailed description about the code or extraction method. In any case, our aim was to find a more efficient method than the BS algorithm and, in order to do so, we decided to list a few rules that can be used to select the most appropriate method. The rules are listed as follows:

1. The method should be easy to understand and implement.
2. The method should be robust.
3. The method should not be stuck in local minima.
4. The method should be computationally more efficient.
5. The method should learn from all the simulated structures.

3.3.2 Encoder-Decoder Sequence Model (EDSM)

Methodology

Our literature study points to the type of Neural network, known as the Encoder-Decoder network or Sequence to Sequence network, a model consisting of two Recurrent Neural Networks (RNN) called the encoder and decoder. The encoder reads an input sequence and outputs a single vector known as the state, and the decoder reads the state (vector) as input to produce an output sequence. An RNN is an artificial neural network (ANN) generally applied for sequential or time series data. Researchers use these neural networks for ordinal or temporal problems, such as language translation, natural language processing (NLP), and speech recognition. They are related to popular mobile applications such as Siri [144] in Apple, voice search in Alexa [145], and language translation in Google Translate [146]. We show the schematic difference between the CNN and RNN (Encoder-decoder network) in Fig-3.5. Here, one can see that CNN (right) uses a feed-forward neural network (transferring all the information in one direction) compared to RNN (left), which uses a backward feed network in the hidden layer (transferring information back to the hidden layer). One can find a detailed comparison in the article [147].

EDSMs refer to two recurrent neural networks: one network to encode the input sequence, called the encoder, and a second to decode the encoded input sequence into the decoder target sequence. Here, we discuss the list-3.3.1 as mentioned above to evaluate the performance of the EDSM method compared to BS as shown in Fig-3.6. From the list, we can see that we are satisfying points 2, 3, and 5. In fact,

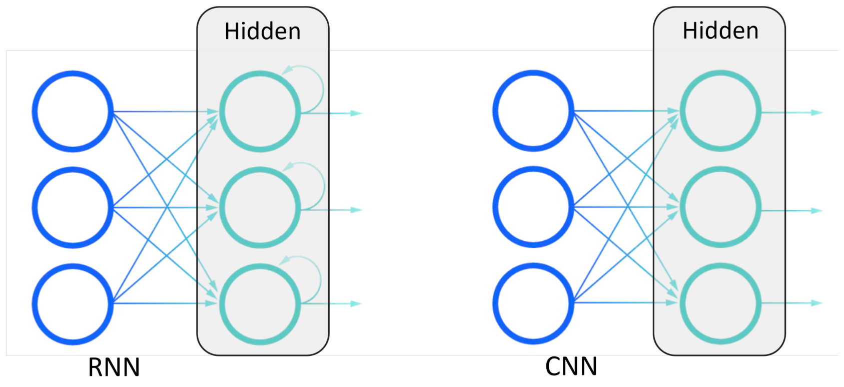


Fig. 3.5 Schematic difference between the CNN (right) and RNN (left) types of Networks. The figure shows the main difference between the two types of networks: RNN feeds the results back into the network, whereas CNN uses feed-forward architecture types.

since RNNs are "Robust" by definition and are trained using all the structures as shown in schematics Fig-3.6(b), we have less probability of being stuck into local minima. However, EDSM is not as easy to understand and implement as BS, and we need to check the method's efficiency. To compare with the BS, first, we need to generate the dataset, then we decide on the architecture, and later, we check the performance of the EDSM.

Dataset Generation

In order to train the EDSM, we need the dataset to train and test the algorithm. Therefore, instead of generating a new dataset for the EDSM, we use the data calculated during the Binary search method. In that case, we calculated around one thousand different configurations, which we use for the training and test cases. Before using the data to train the model, we should check the distribution of the objective function. Therefore, first, we separate our objective function according to the two output waveguides. So, for output waveguide 1, we calculate $(I_{f_1,O_1} - I_{f_1,O_2})$ and for output waveguide 2, we use $(I_{f_2,O_1} - I_{f_2,O_2})$. Moreover, we normalized both values and plotted the distribution in Fig-3.7.

In order to train the EDSM properly, the distribution should be normal; in fact, if the distribution is not normal, the model will automatically incline towards the more frequent regions. From Fig-3.7, we can see that the number of the samples having intensities between 0.0 and 0.1 is double with respect to the rest of the distribution.

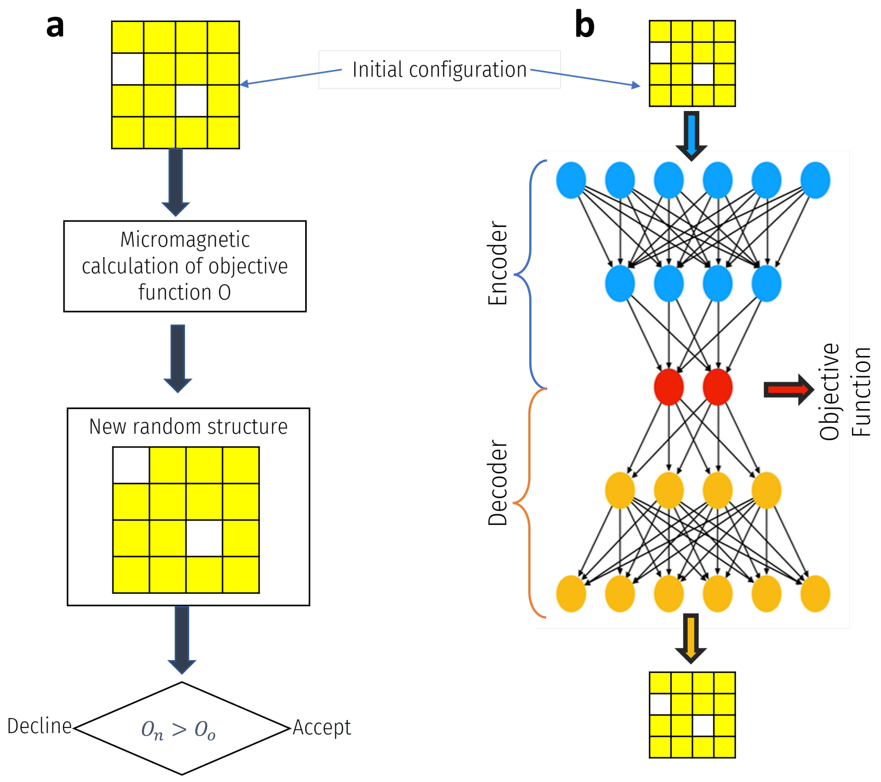


Fig. 3.6 Schematic difference between the BS (left) and EDSM (right) types of Networks. We can see that the BS will start from the initial structure and optimize from the first one. On the other hand, EDSM needs to be trained before generating results. Other differences are given in the text. The panel (b) is constructed using the ref-[7].

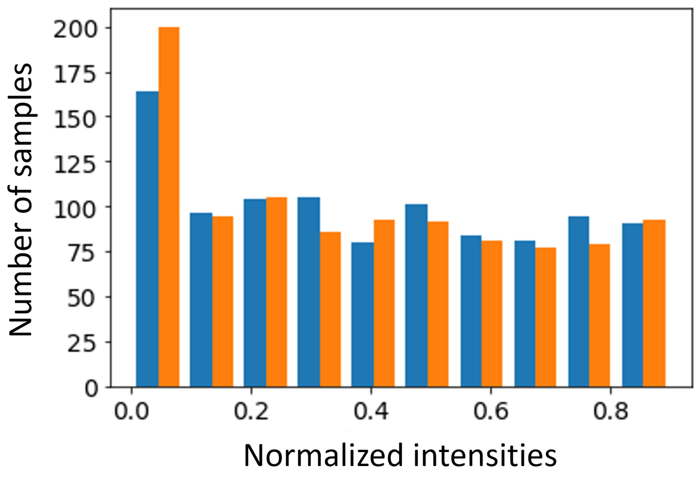


Fig. 3.7 Histogram of relative intensity flowing through each arm. The blue bars indicate the 2.6GHz flowing in output guide 1 and orange bars indicate the 2.8GHz flows into the output guide 2 for any particular configurations. The Y-axis indicates the number of sample (configurations) resulting in the intensity.

To account for this situation, we used a random number of 100 samples out of 200, with an intensity of frequencies between 0.0 and 0.1. In this way, we are left with 900 configurations, which have intensities of frequencies between 0.0 to 0.92, as shown in Fig.3.7.

EDSM Results.

In our case, we use the "Many to Many sequence model" architecture for the EDSMs, as shown in Fig-3.8. We provide the detailed code in the Appendix-1. This type of Network "waits" for the encoding step to finish producing the internal state (Fig-3.8 blue colored region). Then, it starts decoding when the encoder is finished (Fig-3.8 orange colored region). So, the two parts work individually, translating from sequence to state for the encoder and state to sequence in the decoder.

In the EDSM, first, we will try to predict the objective function from the sequence of {0,1} in the encoder part, and after that, we will transmit the objective function to the decoder part to recreate the sequence which was given as the input as shown in Fig-3.9. We will stop the training as the decoder generates a similar sequence as the input. After that, we will disconnect the encoder part and only use the decoder part to predict the effective sequence.

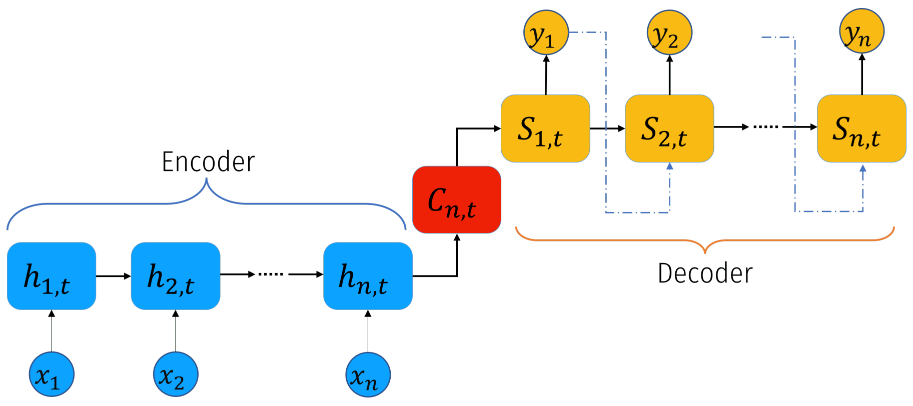


Fig. 3.8 Graphical representation of Many to Many sequence model. Here, (x_1, x_2, x_n), (y_1, y_2, y_n) represent the input and output vector respectively. $(h_{1,t}, h_{2,t}, h_{n,t})$, $(s_{1,t} s_{2,t}, s_{n,t})$ represent the hidden state vector for the time t for the encoder and decoder respectively. $C_{n,t}$ represents the objective function we try to predict at time t.

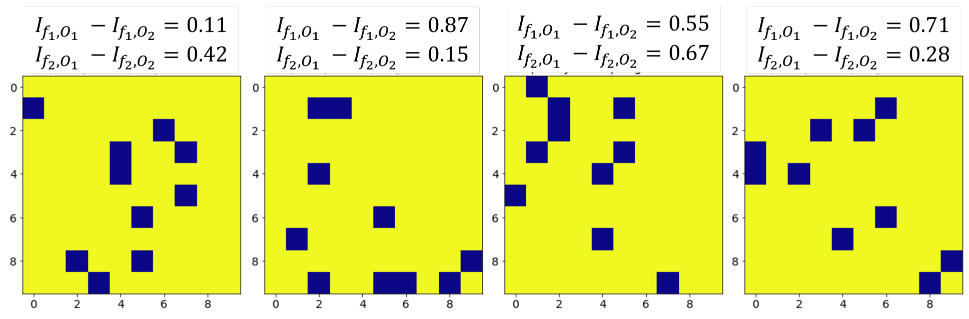


Fig. 3.9 Examples of the configuration that we have used to train the Encoder part. On every configuration, the top line suggests the value of the intensity of f_1 in the output guide 1, similarly for the output guide 2. The yellow color shows the YIG material, whereas the black color indicates the vacuum.

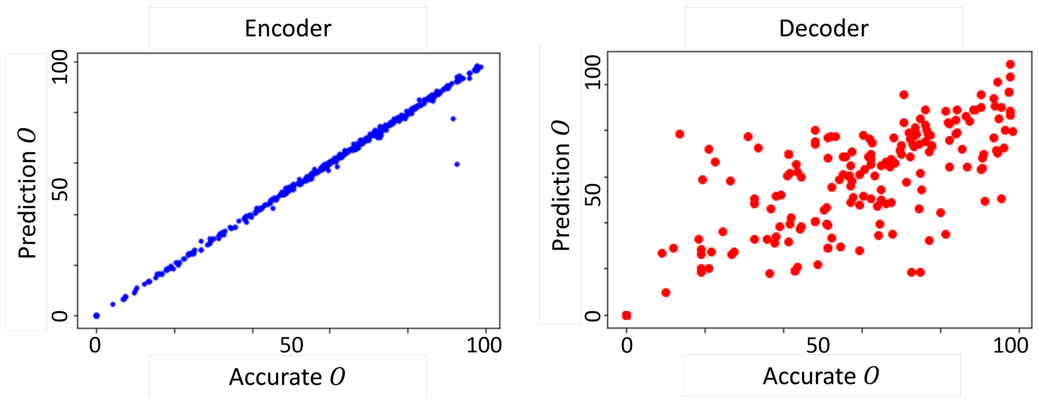


Fig. 3.10 Left (right) panel: results from the encoder (decoder) network. Here, we give the sequence of the configuration to the encoder to predict the intensity in two output guides. These intensities will transmit to the decoder network to predict the input configuration. Here "O" represent value of objective function3.1 on X and Y axis. In addition to that, accurate and prediction O in the encoder suggest the Mumax3 calculation and machine learning output for the provided sequences respectively. However, on the decoder side, accurate and prediction O suggest the Mumax3 simulations of actual sequence and predicted sequence from EDSM

In Fig-3.10, we plot the results from the encoder (left) and decoder (right) for the 900 configurations. In Fig-3.10, Encoder(left) part predicts the two output values of the objective function from the sequence of 0,1. At a later stage, the decoder(right) part predicts the sequence of 0,1 from the applied objective function. In the end, We simulate the sequence(predicted configuration) using Mumax3 and compare these values again with the predicted values shown in the decoder(right) From these plots, we can conclude that the encoder part works relatively better than the decoder. These two sequences iteratively optimize the structure, and training is done by making a probability map of accuracy for different configurations. We found that the number of steps required to train the model is higher than the previous BS. However, we also remark that, provided enough steps are taken to train the model, the accuracy is higher than the BS algorithm. Nevertheless, since our bottleneck is represented by the time to perform MuMax3 simulations, we limit ourselves to optimizing the configuration within 1000 steps. Therefore, our conclusion is that point-4 (about computational efficiency) in the list3.3.1 is not well fulfilled and we need to find another method.

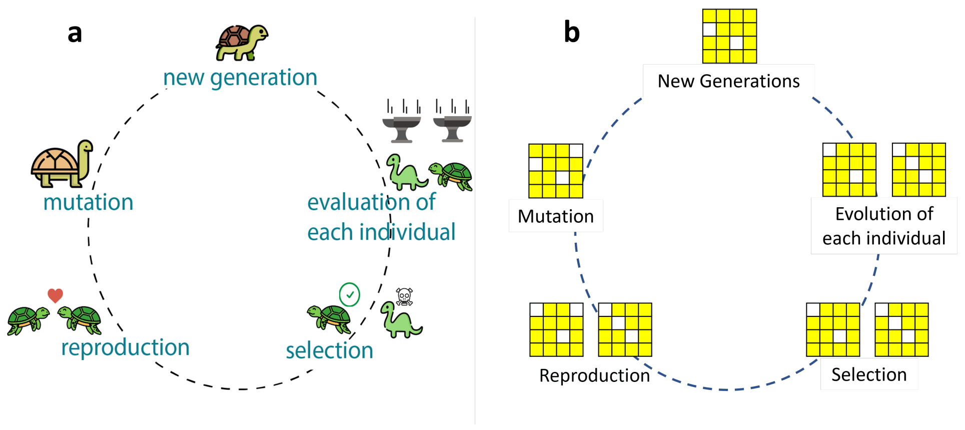


Fig. 3.11 Graphical representation of the concept of Genetic algorithm in nature (a) and its use to optimize the configuration (b). The panel a) was taken from the ref-[8]

3.3.3 Genetic Algorithm

From the above two methods (BS and EDSM), we can conclude that BS takes fewer steps to find the optimum configuration, but it is more likely stuck in local minima. On the other hand, EDSM is less likely to be stuck in local minima, but needs a considerable amount of data to be trained. Therefore, we need to find a method which shows the advantages of both methods. To this end, we used a Genetic Algorithm, inspired by the process of natural selection (Fig-3.11(a)) and belonging to the larger class of evolutionary algorithms (Fig-3.11(b)). This method creates a population consisting of several different configurations and calculates the frequency separation of each configuration. The configurations with higher separation of frequency will go ahead for crossover, stochastically generating new solutions from an existing population. Recursively, we optimize the configurations, and after several generations, we have an efficient structure as shown in Fig-3.11(b). We anticipate that configurations evolved in 80 steps, i.e. five times less than the BS algorithm. Furthermore, we found that the GA could optimize the structure more efficiently than the method used in the reference article[6].

Methodology

Initial Population The process begins with a set of individuals, called a Population. Each individual is a solution to the problem one needs to solve. These individuals contain a set of variables or parameters, known as Genes. In the mathematical model, these genes are combined to create an individual. In our case, we have 10 configurations in a population, each containing 100 genes (converting 10×10 matrix into one dimensional array). A simple example of conversion is shown in Fig-3.12

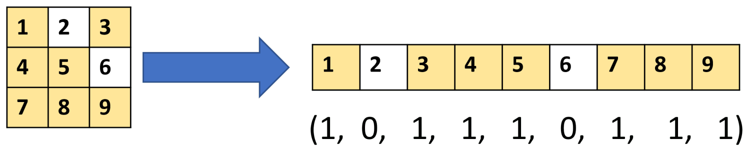


Fig. 3.12 Graphical representation of population generation.

Fitness Function In general, the fitness function determines how fit an individual is. It gives a fitness score to each individual. The idea of the selection phase is to select the fittest individuals from the population and let them transfer their genes to the next generation. Here, we select two pairs of individuals based on their fitness scores. Individuals with high fitness have more chance to be selected for reproduction. In our case, we consider the fitness function to be the same as the objective function-3.1 calculated above. We would like to mention that this is the most expensive step in Genetic Algorithm, since it includes the MuMax3 [143] calculations.

Crossover This is the most important step in the GA. Each individual selected in the previous step, becomes a parent. For each pair of parents to be mated, a crossover point is chosen at a random point from where genes will interchange, as shown in Fig-3.13. The interchanged individual is known as offspring.

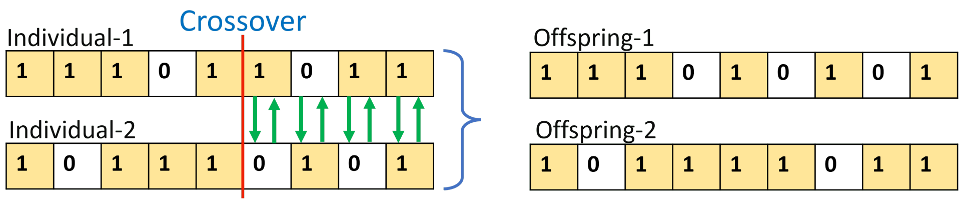


Fig. 3.13 An example of single point crossover and its new off springs.

Mutation In certain new offspring formed, some genes can be subjected to a mutation with a low random probability. This implies that some of the bits in the bit string can be flipped. Mutation occurs to maintain diversity within the population.

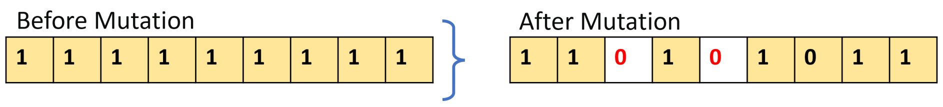


Fig. 3.14 An example of Mutation. The panel on the left shows the unmutated individual and the panel on the right shows the mutated version (see numbers in red).

Termination This function serves two purposes. 1). If the population reaches the predefined target fitness, the entire algorithm stops and provides the solution. 2). If one does not achieve the target function, then the function decides to keep the individual (configuration in our case), which shows high fitness (objective function value) and terminates the rest of the individuals. Here, we can keep one, two or three pairs which can move forward to the next generation to keep growing from there. However, there is no rule of thumb to decide the number of pairs to keep. In our case, we found that two pairs out of five show an optimal convergence. If we use less than two pairs, the randomness in the result will increase, and it will take longer to converge; on the other hand, if we use more than two pairs, we might again face the problem of being stuck in local minima, though convergence is faster.

Results from the Genetic Algorithm

Performance of methods for the demultiplexer with respect to the main general objectives. Here, we compare the results from three different methods. We utilized the above mentioned list-3.3.1 to check the method's efficiency. For simplicity, we recall the list 3.3 below: From that, we can see that the GA method satisfies point number 2, 3 and 5, which we recall to be related to robustness, local minima and learning from all the simulated structures, respectively. Indeed, the GA method can be used for any set of problems, therefore it is robust. In addition to that, we are adding random structures at all times, therefore the algorithm is less likely to be stuck in local minima. Lastly, the GA method is optimizing from all the structures that we are using. Furthermore, the implementation of the method is not as easy as BS, but also not as difficult as the EDSM. Therefore, we partially satisfy point number 1. In addition, the results shown in Fig. 3.15, are achieved in 9 generations (converted to 90 MuMax3 structures) compared to the 800 steps needed in the BS. Wang at el. reaches the maximum accuracy of 96% and 80% for two frequencies after 800 steps. By using GA, we achieve the accuracy of 86% and 83% for the same frequency in around 90 steps. Therefore, the GA satisfactorily reaches all our objectives, as shown in list-3.3.1.

1. The method should be easy to understand and implement.
2. The method should be robust.
3. The method should not be stuck in local minima.
4. The method should be computationally more efficient.
5. The method should learn from all the simulated structures.

Given these premises, we proceeded by using GA to simulate another device.

Attenuator. Among the possible choice of devices to be simulated, we have created an attenuator using the GA. Here, we would like to reduce the amplitude of the two output waveguides without changing the wavelength. Hence, we used the same design shown in Fig-3.2. In this case, we decided to reduce the amplitude by one-third without loosing the phase. Therefore, we should get one-third of the input amplitude into each output waveguide and one-third of the total amplitude should be absorbed inside the design region.

In the aim of doing this, we first initiated the structure and then created different configurations inside the design region. We used the steps described in section-3.3.3 to find the optimum structure. In order to do so, we first generated the initial population with ten configurations. Then we performed the MuMax3 simulations to check the fitness of each configuration. After that, we selected the top 4 configurations with the highest fitness score (high-objective function). Then, we performed single-point cross-over and mutation on the best four configurations. In addition to that, we also created six new random configurations which will be our population for the next generation. In the end, we got the optimal solution shown in Fig-3.15(c). Here, from the graph shown in the right insets of Fig-3.15(c), we can see that the Attenuator works as expected. In fact, the orange and blue lines in the plot show that both output spin-wave signals have the same phase and we also reduce the spin-wave amplitude to 33% in both output waveguides. Therefore, this additional example of the attenuator helps us to confirm the effectiveness and robustness of the GA method as a tool to be used for more complex tasks.

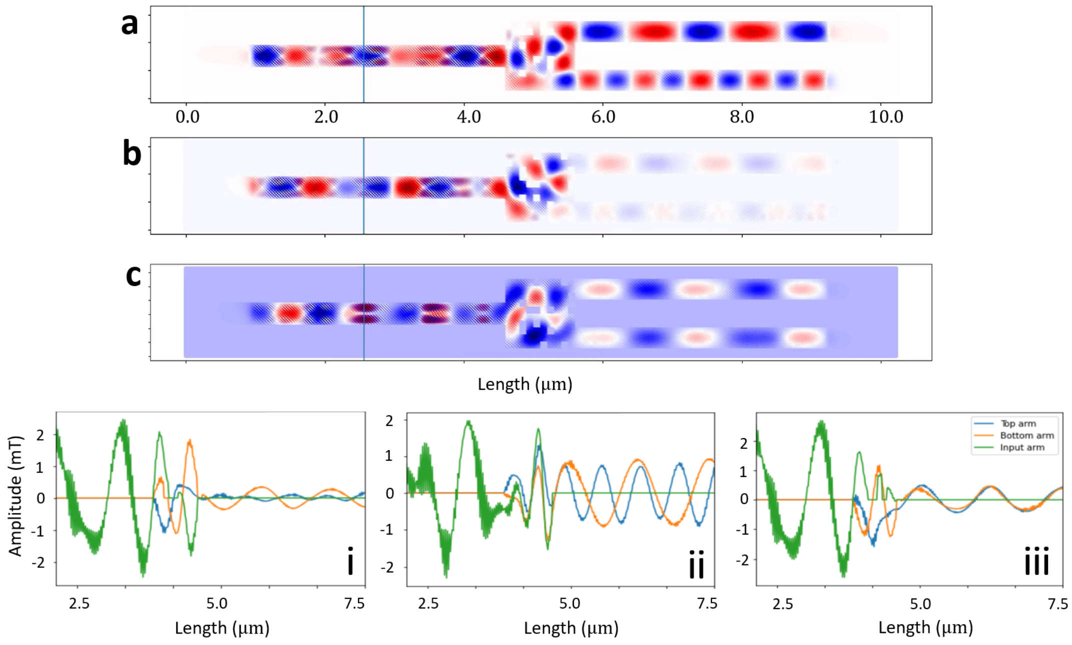


Fig. 3.15 Three types of devices. **a)** demultiplexer submitted in the reference article [6]. **b)** demultiplexer achieved by the GA. **c)** attenuator (reducing amplitude) created using the GA. The graphs on the right column show, for each device, the amplitude at different lengths. Here, we plot the green line for $2.6 + 2.8GHz$ at the input waveguide, whereas the orange and blue lines show the spin-wave signals in the two output waveguides. Moreover, Figure-**i,ii,iii** shows the amplitude of spinwave at the horizontal cross section of device for **a,b,c** respectively.

3.4 Additional GA-optimized magnonic devices for complex tasks

After checking on benchmark problems, we started simulating more complex devices, where - instead of separating two different frequencies - we would like to create a device which can perform more complex tasks. So, after successfully verifying the technique, we created some complex magnonic devices, which will be discussed in this section. We would like to mention that all the devices mentioned in this section are optimized by the GA. Remarkably, we were able to achieve the optimal configuration in less than 100 steps, *i.e.* a number significantly lower than the BS algorithm and EDSM and, as such, a clear sign of efficiency.

3.4.1 Demultiplexer for three frequencies

As mentioned above, we also would like to use multiple input and output guides to simulate complex devices. However, we must first verify that multiple input and output guides are applicable. In order to do that, we first used a fairly complex example shown in Fig-3.16, where the design region is separating three frequencies (2.4, 2.5, 2.6 GHz). We use one input guide and check the output at three output waveguides. We aim to guide each of the frequencies' spin waves into one of the output arms; in other words, each frequency (2.4, 2.5, 2.6 GHz) should have the highest amplitude in one of the three waveguides.

The size of the total simulation box is $15.3 \times 4.096 \mu m^2$. In addition, the size of input and output arms are $5.7 \times 0.42 \mu m^2$ with a square design region of $15.68 \mu m^2$. Each square defect inside the design region is $0.1764 \mu m^2$. The entire device is shown in Fig-3.16(a). After constructing the device mentioned above, the aim is to find the configuration of the optimal defects to separate the three frequencies. Therefore, we used a similar technique to that used for the demultiplexer with two output waveguides[6]. We optimize the device using the GA constructing 10×10 matrix inside the design region constituted by material and holes.

After optimization with GA, we found the optimal configuration to separate the frequencies. In Fig-3.16(a), we show one of the frequencies (2.6GHz) guiding to one of the output waveguides $(f_{0,1})$. In addition to that, we also observe frequencies (2.4 GHz, 2.5GHz) that are maximized in the output waveguides $(f_{0,2}, f_{0,3})$ respectively.

However, we also notice some problems, when converting to more complex devices, as follows;

- The absorption - as observed in Fig-3.16(b) - is here comparatively higher than in the demultiplexer.
- The separation between all three frequencies is less distinguishable, compared to two output waveguides; in other words, i.e. much of the amplitude of a particular frequency is propagating through the other waveguides.

First, we notice that the number of defects (holes) directly affected the spin waves' absorption rate; this is straightforward, due to the increase of interactions between the spin waves with a larger number of holes. From Fig-3.16(b), we can see that the amplitude in the input waveguide (blue line) is significantly higher than the output waveguides (orange and green lines). The results mentioned above effectively address our second concern, which relates to the trade-off between the absorption and separation of spin waves. In order to separate more than two frequencies, additional defects (holes) need to be created, which, in turn, results in higher absorption. Conversely, fewer defects lead to less separation. To strike a balance between the two, we have slightly modified our objective function- 3.1. Rather than calculating the intensity of the output waveguide, we have calculated the relative intensity among three waveguides. This approach has also been employed in another case discussed below. Moreover, we remark that, the small difference in frequency between the input signals represent a limiting factor for differentiation, since spin waves with similar frequencies interact with defects in a similar way. Using signals with larger frequency differences, such as 2.2, 2.5, and 2.8 GHz, could yield better results. This presents a promising opportunity to enhance the performance of the magnonic neural network.

3.4.2 Alphabet Identifier

In this section, we delve deeper into creating a more complex magnonic device by leveraging the knowledge gained thus far. Our goal is to build a magnonic device that can identify alphabet characters from pixels. We use a frequency of 2.6 GHz with nine inputs, each representing a 3x3 grid of pixels.

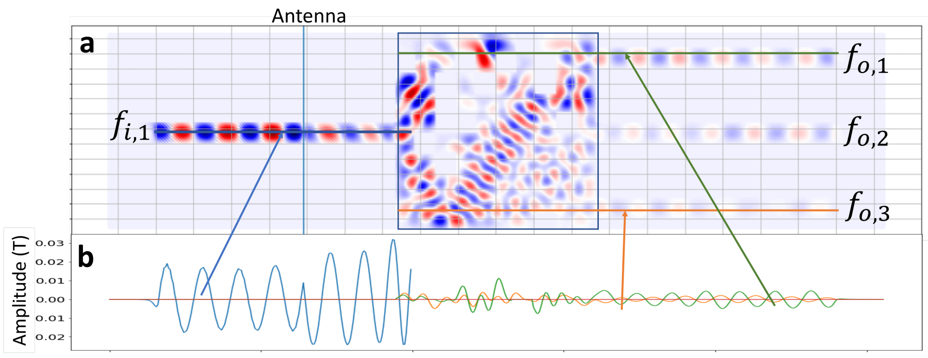


Fig. 3.16 (a) Demultiplexer with one input and three output. (b) Amplitude of the device at the input (blue line) and output waveguide (orange and green lines). The red and blue colors inside the device represent the total magnetization in Z direction (positive and negative Z shown in red and blue, respectively).

To avoid cross-talk between the output signals, we use three output waveguides. Cross-talk refers to the undesired effect caused by a spin wave transmitted on one waveguide affecting another waveguide. Furthermore, we must address the absorption problem identified in the previous section. As a result, we decided to distinguish between three classes of characters.

Using a 3x3 grid of pixels, we can create 12 alphabet characters, as depicted in Fig-3.17. To ensure compatibility with future experimental realizations, we use a simulation box size of 50 micrometers by 10 micrometers. We represent each pixel with a separate input waveguide, as shown in Fig-3.18(b). To set an alphabet character, we block the input corresponding to the character's pixels, as illustrated in Fig-3.18(c). The complete MuMax3 script is provided in the appendix (Appendix-.2) along with the accompanying figure (Fig-2)

As benchmark, we used three alphabet characters out of 12, as shown in Fig-3.17. We follow the process shown in Fig-3.18 to simulate the device. First, we chose three alphabet characters: L, N and X. We then converted our 3×3 matrix into the one-dimensional array, as shown in Fig-3.18(b). According to the generated array, we block the spin-wave of the input device by creating a hole inside the waveguide, as shown in Fig-3.18(c). Then, we created the defects inside the design region. Subsequently, we excited the spin waves by antennas, which propagate from the input waveguide to the design region and output waveguide. We calculated the intensity of the spin wave at each output waveguide after 100ns. We repeated this process for other classes of the alphabet (*i.e.*letters N and X). We again use the GA

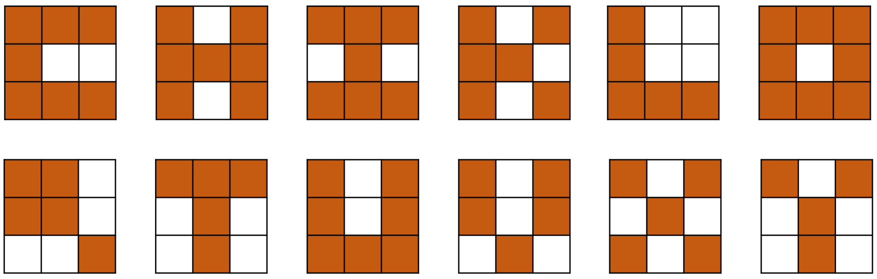


Fig. 3.17 Representation of 12 alphabet characters by a 3×3 matrix.

to optimize the design region and configuration, so that it can distinguish between the three alphabet classes.

In this case, we could indeed differentiate between three different alphabetical characters. The simulation for the letter "L" can be observed in Fig. 3.19. As depicted in Fig.3.19(b), the matrix in the center absorbs a considerable amplitude. Nonetheless, the demultiplexer remains capable of distinguishing between three letters. One issue that needs to be addressed is thus the balance between amplitude separation and absorption in the matrix, as can be seen in Fig-3.19(b). In fact, we also observe weak spin waves magnitude at output waveguides. In addition to the problems we had faced in the previous device, we also encounter cross-talks between the two output waveguides. To this end, one also needs to optimize the distance between output waveguides.

3.4.3 Digit Identifier

We used our previous experience to build a device that works as a physical neural network. In order to do that, we simulate a device which can identify the digits from the pixels. Digit identification is a fairly common problem in Deep Learning, so that it can be a good starting point for constructing the Physical neural network. Therefore, unlike the previous section, here we use seven pixels, as shown in Fig-3.20(b). Using these seven pixels, we can create all the numbers between zero to nine, as shown in Fig-3.20(a). In addition, here, we are using one input and nine output arms. Furthermore, each pixel is represented by a different frequency of $2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0$ GHz. In addition, upon learning from our previous problems, we would like to be free from the waveguide cross-talk issues. Therefore,

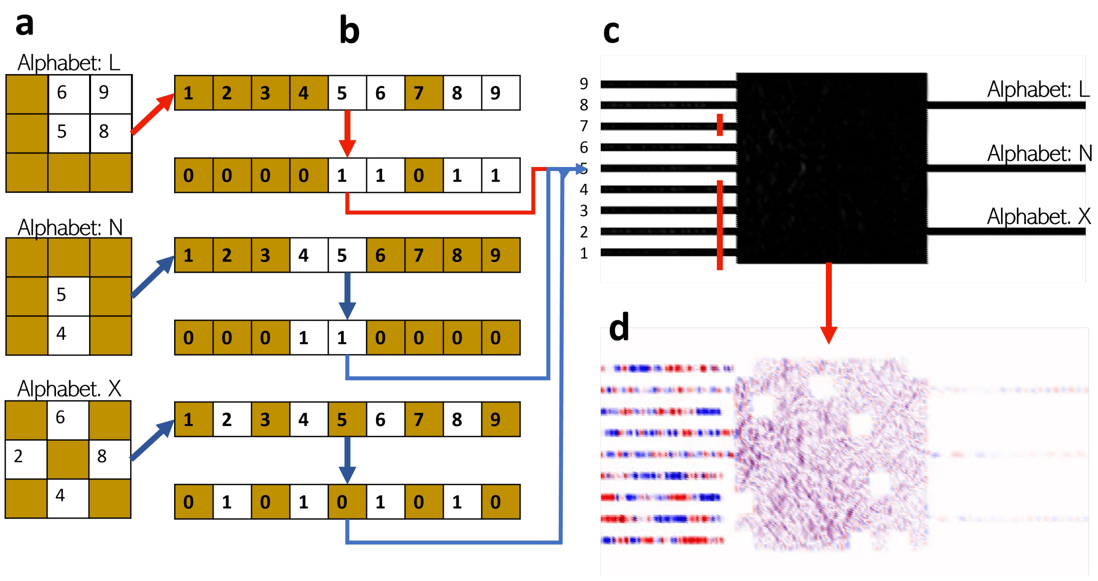


Fig. 3.18 Workflow of the alphabet identifier. Panel a) shows three letters: L, N, X. Panel b) shows the binary conversion of the 3×3 pixels. Panel c) shows the complete device with nine inputs, representing the binary array created in panel-b. Panel d) shows the simulation result for the letter L. Here, we took the example of the letter L to show the Workflow, represented by the red lines.

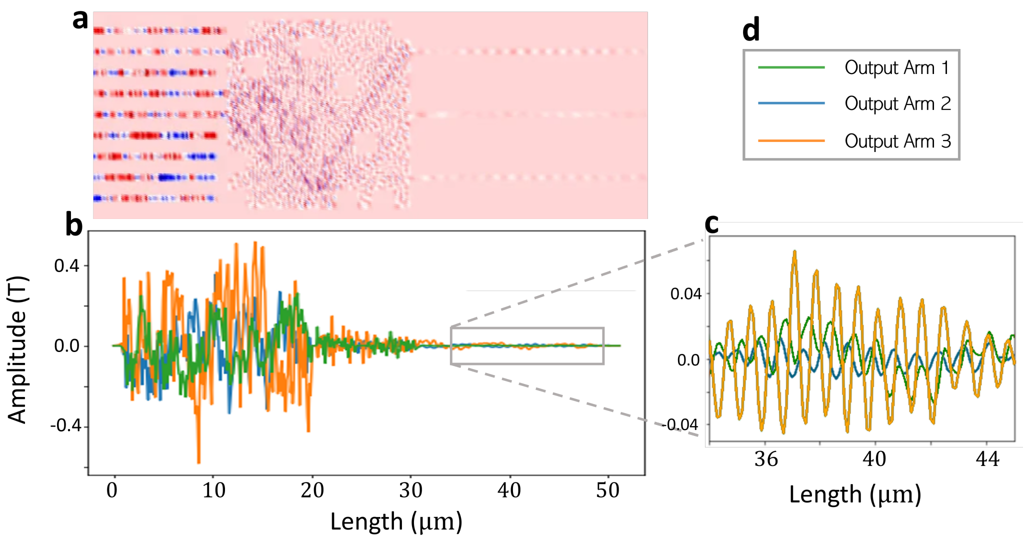


Fig. 3.19 Example of alphabet identifier. a) device simulation using input letter "L" at 50 ns. b) magnetic amplitudes, presented for three distinct cross sections located at the midpoint of each output waveguide along its length. c) zoom out of panel-(b) in the range from 34 to 44 μm. d) legend used for the different arms.

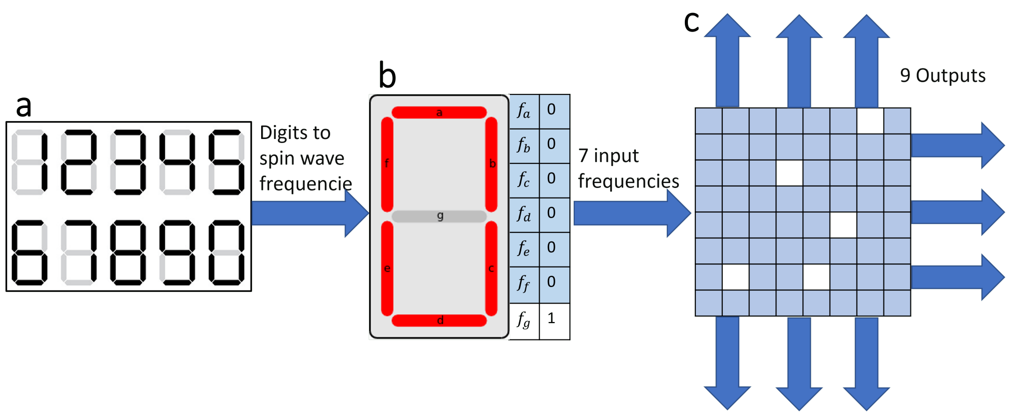


Fig. 3.20 Concept of physical Neural Network devices where Digits from pixels can be classified using optimized structure defects. Panel a) shows that all digits can be described by seven pixels. Panel b) shows the example of digit zero conversion to related frequencies. Panel c) shows the physical magnonic device to classify each digit.

instead of putting nine arms in one direction, we put three output arms in three different directions, as shown in Fig-3.20(c).

The size of the simulation box is $15\mu m \times 15\mu m$, with the arms' size being $5.68\mu m \times 0.4\mu m$. The middle section of the design region of the device is $4\mu m \times 4\mu m$, where we also created a 10×10 matrix; the latter is to be optimized according to the objective function. Here, we again use the GA to optimize the device and to calculate the objective function; we calculate the amplitude of each arm at $95ns$. We assign the arm with the highest intensity for that particular number and run the new simulation for other number. In the end, we check how many of the numbers are assigned to their unique output arm. The algorithm stops exploring the new configuration when all the arms have uniquely assigned numbers. Here we would like to mention that according to Fig-3.20, we did not consider an output guide for the number "8", because it has no input frequency. Therefore, we only need to care about nine arms instead of ten.

We limit ourselves to 1000 steps with 100ns simulation time for each number, since MuMax3 simulations need intensive computational resources. In our case, 1000 steps with a 1024×1024 simulation size took around three days. Therefore, if we limit GA to optimize the structure in 1000 steps limit, we found the optimum configuration with seven unique arms assigned to seven different numbers. However, we again run into the problem of spin-wave absorption, when we increase the number of defects. In addition, we also noticed that the input frequencies are not significantly

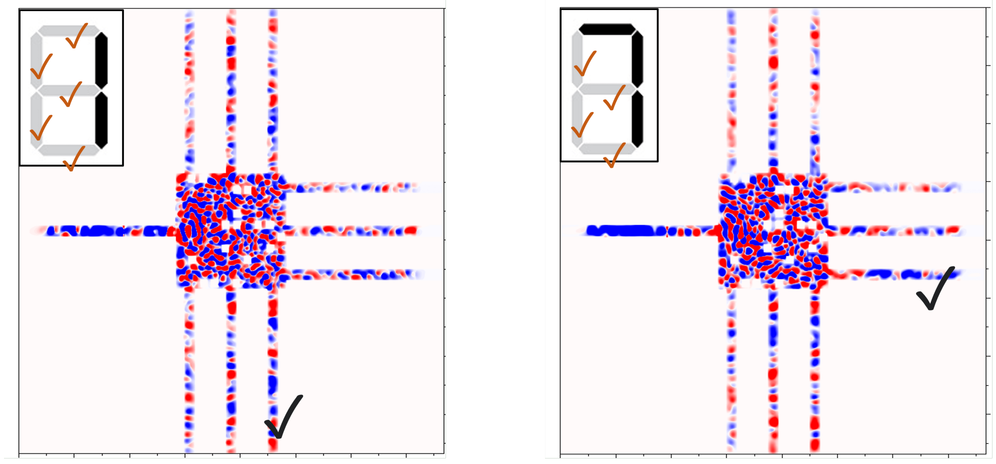


Fig. 3.21 The problem with the digit identifier is shown in the Figure above. The Left and right side panel shows the spin-wave propagation of frequencies representing number 1 and number 7, respectively. The badge in the output guides of both figures shows the waveguide with the highest amplitude, suggesting the input number. The two panels show that the waveguide with the highest amplitude is less distinguishable than in the demultiplexer case, due to the frequency similarity between the two numbers. The insets in the top left corner of each Figure with badge show that 4 frequencies are the same in both numbers.

different from some of the numbers, as shown in Fig-3.21. This figure shows the spin-wave structure of two numbers: one and seven. By pixelating the numbers, as depicted in the top left corner of each figure, we notice that four of the five frequencies are identical, causing slight differences in the intensities of the output signals. The frequency separation in the input signals is also limited. To obtain more favorable results, using a wider frequency separation may be considered..

3.5 Summary

Upon successfully separating spin waves between two frequencies, our main objective for this project was to build a device that can work like a physical NN or NN filter; as such, it could help to assist the traditional transistor based silicon SOC. In order to do that, first we have started with a simple example from the literature[6], where we separated two frequencies by creating defects in the design region. Compared to the literature, we used an alternative approach based on Genetic Algoirhtm which proved much more efficient than previously proposed approaches. As a next step, we wanted to identify the digits using a magnonic NN device, as shown in

Fig-[3.20]. Here, we assigned different frequencies to different pixels for each digit, as shown in 3.20(b). After this step, we created defects as 3.20(c) and found the optimized configuration, where each frequency related to each digit is guided into one of the specific output arms.

However, in our study, we found that the CPU time for micro-magnetic stimulations increases with complexity. For example, the present method takes around a week of simulation time of sample optimization. Therefore, we need to optimize the ongoing method to shorten the simulation time or to find a more efficient method. After successfully modeling the device, the group in Aalto will consider experimentally fabricating the device to get an experimental confirmation of our predictions and move to real-world applications.

In the future, we would like to create a dynamic device, where we can exploit another magnetic property, such as the anisotropy, to manipulate the spin waves. In this case, we can add an additional layer on top of the device, which can change or control the magnetization at different sites. On a longer time-scale, building more accurate spin wave-based devices might indeed have the potential to replace traditional transistor-based devices.

Chapter 4

Understanding the Modification of Magnetic Domains Using Machine Learning

4.1 Introduction

Magnetic thin films with perpendicular magnetic anisotropy (PMA) are of great technological relevance and are considered as promising materials for spintronic nanodevices. These films are widely used in areas like ultrahigh-density magnetic storage, fast memory applications, and nanosensors. When favorable atomic ordering is achieved, ultrathin stacking of Co with Pd or Pt exhibits PMA[42–45].

Having information on magnetic microstructure of PMA thin films and multilayers is critical in understanding magnetization reversal processes. This is because the net magnetic energy of a thin film can be represented as the sum of the anisotropy energy, exchange energy, magnetostatic energy, and Zeeman energy[46, 47]. The minimization of the anisotropy energy occurs when the magnetization points along the easy axis; the latter can be tuned by adjusting the thickness, growth parameters, and processing routes of the material. The exchange energy is minimized when the spins align in a parallel (or antiparallel in case of antiferromagnets) way, leading to the formation of a single domain. On the other hand, the magnetostatic energy opposes the single domain formation and is primarily related to the saturation magnetization and to the shape anisotropy. Finally, we recall that the Zeeman energy is the energy term that occurs when the magnetized system is placed in an external magnetic field.

Recently, machine learning techniques, particularly Convolutional Neural Networks (CNNs)[147], have been used to understand the modifications of magnetic domains in perpendicularly magnetized multilayers. These advanced machine learning methods have been applied in various interdisciplinary research domains, such as microstructure optimization, prediction of magnetic fields, phase transitions, magnetic grain size studies, modeling of magnetic domains, and estimation of different components of Hamiltonian [148–156].

In additon to that, we recall that Ion-beam irradiation is a popular technique used to tune magnetic properties in magnetic multilayers[50, 51]. The ion energy and fluence can be adjusted to control the depth and lateral extent of the modification effects, leading to changes in magnetic properties[52]. Studies have shown that ion-beam irradiation can create graded anisotropy media by positioning domain walls and can also induce depth-resolved structural modifications in magnetic multilayers[53, 54, 50].

Despite ongoing efforts to refine the control of these parameters, the multidimensional phase space in which parameters can be changed pose a significant challenge for researchers to effectively use this approach. That's why we're proposing to harness the power of Machine Learning to overcome the limitations. As already noted, in the 21st century, machine learning techniques are increasingly proposed as a useful tool to understand materials systems with interdependent and simultaneously variable properties. In this chapter, we will address the application of the convolutional neural network to understand the modifications of magnetic domains in perpendicularly magnetized multilayers, observed experimentally by using ion-beam irradiation[50–52]. Recently advanced machine learning techniques have acquired immense importance in interdisciplinary research related to magnetism, such as micro-structure optimization[157], prediction of harmonic magnetic field based on Helmholtz equations [158], phase transition [159], magnetic grain size study[160], modelling magnetic domains [161, 162], the relation between different magnetic chiral states[163], prediction of effective magnetic spin configurations[164, 165], 2D metal-organic frameworks with high magnetic anisotropy[166], and different components of Hamiltonian including Dzyaloshinskii–Moriya interaction (DMI)[167] using different deep learning and machine learning methods. From the point of view of atomistic magnetism, researchers [165, 164] have tried to estimate and analyze various components of spin-based Hamiltonians, such as exchange constants, anisotropy constants, and DMI using different convolutional neural networks (CNN)[168]. The advanced CNN methods showed effectiveness and accuracy in different research domains. However, these methods generally require a large data set[169] to properly train and test the model, which might represent an obstacle in using these methods to analyze regular experimental data (cfr .3).

Before proceeding further, we mention that the presented work was done in collaboration with Abhishek Talapatra (at Aalto University, Finland) Syam Prasad (at Indian Institute of Technology, India), Jeyaramane Arout Chelvane (at Defence Metallurgical Research Laboratory, India) and Jyoti Ranjan Mohanty (at Indian Institute of Technology, India).

4.2 Methodology

4.2.1 Micromagnetic Simulations

Micromagnetic simulations of magnetic domains in perpendicularly magnetized thin films/multilayer were performed using the MuMax3 software [143], as explained in Chapter-2. The system of interest was a Co/Pd multilayer, considered as a single magnetic layer with an effective anisotropy constant (Ku_1) and effective thickness (t_{eff}) of the magnetic layers, comparable to the multilayer one. The input parameters were close to the values reported in the literature [170]: exchange constant (A_{ex}) = 23 pJ/m, saturation magnetization (M_s) = 1 MA/m, and Ku_1 = 1 MJ/m^3. The simulation temperature (T) and damping constant (α) were 300 K and 0.9, respectively. Cubic meshes of volume $(4\ nm)^3$ were used for the discretization of the total area of simulations ($\sim 2\mu m \times 2\mu m$) with t_{eff} = 16 nm. The simulations started from arbitrary initial spin configurations and ran for 100 ns to obtain energy-minimized stable magnetization configuration, following the Landau-Lifshitz-Gilbert equation [171, 172]. Here our main focus was to analyze the domain images, which essentially indicate the spatial variation of the overall magnetization (M).

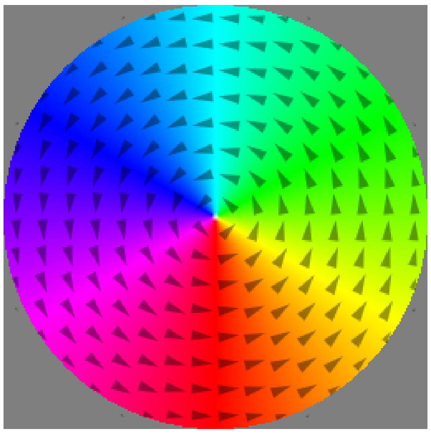


Fig. 4.1 Correspondance between the colorbar for magnetic moments projected on the XY plane and related arrows. We will use the same colorbar to describe the domain images in this chapter. In addition to that, we note that the black and white colors show the magnetisation in positive Z and negative Z direction, respectively.

4.2.2 Simulated Domain Configurations

In this section, we are going *i*) to discuss the possible changes in the magnetic domains pattern, as a function of Ku_1 and A_{ex} and *ii*) to address the qualitative comparison between simulated and experimental results. It is well known that domain walls (DW) can be represented by the net in-plane component of magnetization ($\sqrt{M_x^2+M_y^2}$), occurring at the boundary between the out-of-plane magnetized ($\pm M_z$) domains. The width of the DW is proportional to $\sqrt{\frac{A_{ex}}{Ku_1}}$. We recall that the spin reorientation transition (SRT) is a phenomenon that occurs in magnetic materials, characterized by a change in the direction of the magnetic moments of the constituent atoms as a result of variations in external parameters, such as temperature, magnetic field, or other factors. The SRT can be well understood in terms of magnetization anisotropy, which is also relevant in controlling the width of the DW. The simulated domain images are presented in Fig. 4.2 for two different A_{ex} values and for four different Ku_1 values. For all the domain images of Fig-4.2, the white and black colors indicate two mutually opposite out-of-plane components of magnetization, and the other colors indicate the in-plane components, the orientations of which are shown by the arrows. The domain image in Fig. 4.2(a) with Ku_1 = $1MJ/m^3$ (top) displays an extended maze-like domain structure with strong perpendicular anisotropy. With the reduction in Ku_1 to 0.8 MJ/m^3 , the domain size decreases, and the pattern appears as extended periodic stripes with no preferential orientations and higher

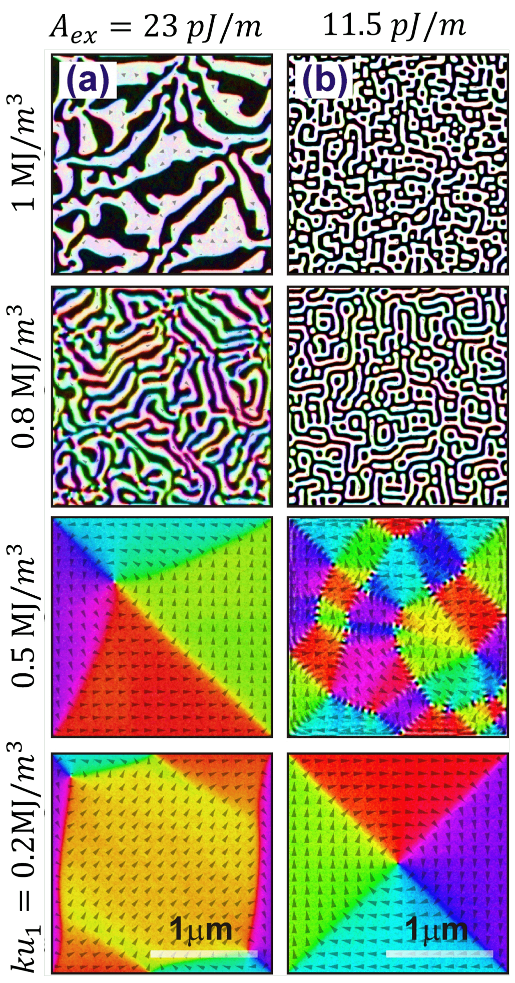


Fig. 4.2 Simulated domain images with different Ku_1 (repored on the side of the panel) with A_{ex} (reported on the top of the panel). The color bar is the same for all the images, consistently with Fig-4.1.

in-plane contrast. Upon further reduction in Ku_1, the extended stripe patterns shrink to circular stripes (not shown) to minimize the magneto-static energy. Interestingly, a strong in-plane contrast leading to asymmetric vortex structure with an out-of-plane magnetized core can be observed with $Ku_1 = 0.5\ MJ/m^3$. Further reduction in Ku_1 results in a feather-like structure with multiple vortices (bottom of Fig. 4.2). The domain features display significant changes with the reduction of A_{ex} to 50%, keeping the same Ku_1. Unlike extended maze-like patterns, worm-like domains without preferential orientation can be observed with $A_{ex} = 11.5\ pJ/m$, as shown in Fig. 4.2(b). Stronger out of plane contrast with reduced domain sizes can be observed with respect to the corresponding domain images of Fig. 4.2(a) with the same Ku_1. Here, the threshold for transition from out-of-plane to in-plane magnetization can be marked at $Ku_1 = 0.5\ MJ/m^3$, where the out of plane components shrink to a thread-like region, bounded by in-plane magnetization in a different direction. Further reduction of Ku_1 leads to a symmetric vortex configuration. Thus, by changing the combination of variable Ku_1 and A_{ex}, one can achieve a tunability in magnetic domain structure. This appears to be qualitatively similar to the experimentally observed domain images in terms of extended maze-like pattern in the pristine condition, fragmented thinner domains with reduced out of plane magnetization after irradiation at lower fluences, and finally, feather-like structure with dominating in-plane magnetization in response to the irradiation at higher ion fluences, as shown in Appendix-.3.

4.2.3 Overview of the Convolutional Neural Network

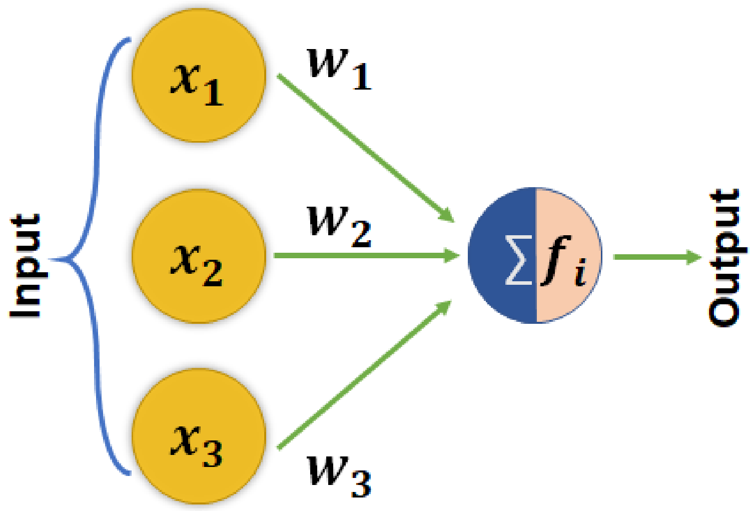


Fig. 4.3 Schematic of an artificial neural network with three input and one output neurons. Here, w_i, x, and f_i are weights, input values, and threshold function respectively.

First, we are going to explain the different components of a Neural Network (NN) and the generally used terminologies. A NN primarily consists of node layers that include input and output layers, and hidden layers (*i.e.* dense layers between the input and output layers). Each neuron is connected with other neurons through weights. If the output value of a neuron reaches the threshold values, that particular neuron is activated to send information to the next layer. In general, the weights between different connections can be adjusted by various methods following the back-propagation, or forward-propagation algorithms to minimize the error in the output node layer. The simplest model is to consider a NN of 3 input layers and 1 output layer, as shown in the schematic of Fig. 4.3. In this case, we can write the equations for the input layer, and the simple threshold or activation function ($f_i(x)$) as,

$$y = \sum w_i x_i + b = w_1 x_1 + w_2 x_2 + w_3 x_3 + b \tag{4.1}$$

$$f_i(x) = \begin{cases} 1, & \text{if } (\sum w_i x_i + b) \geq 0. \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

where, w_i, x_i and b are the weights, input values and bias respectively. A bias vector is an additional set of weights in a neural network with no input, and thus it corresponds to the output of an artificial neural network with zero input. Once the weights w_i are determined, we can also investigate the contribution of different input properties. In general, larger weights for the input values with comparable magnitude represent a higher contribution to the prediction. Then, the summed function will pass through the activation function which determines the output. If the output values are higher than the threshold values, the neuron will be activated. This leads to the propagation of the data to the next layer. In this way, finally, we calculate and minimize the cost function or error function as $f_e(x) = \sqrt{\sum_0^n (y-\bar{y})^2}$, by adjusting the weights.

4.2.4 Addition of Convolutional layers to the Neural Network

While dealing with images, we don't have exact numerical values of the parameters, as mentioned above. Therefore, convolutional layers are used to extract those features from the images[173]. Thus, in addition to the dense layers, the CNN model contains convolutional layers and pooling layers which are used to extract important features from the images. A convolutional nervous system (CNN) is a network of hundred or thousands of neurons that covers the entire image (2D) or visual field (3D) and processes data in the similar way as a biological eye. Each neuron works in its attuned small field, and every neuron is connected to other neurons to interconnect in a way that covers the entire image area. Just as each neuron responds to a particular stimulus only in the restricted region of the visual field, each neuron in a CNN processes data only in its receptive field.

In addition to the fully connected NN, the CNN also consists of the Convolution layer and pooling layer. The Convolution layer, core building block of a CNN, performs a dot product between two matrices: one matrix contains the set of learnable parameters - also known as a kernel -, and the other matrix is the restricted portion of the receptive field (part of the image), as explained by the authors[174] and shown in Fig-4.5. The kernel is spatially smaller than an image, but develops more "in-depth" (usually 3×3 or 4×4 matrix). This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels. In addition to that, we divide the RGB image into three color layers image, where each kernel will work on each of the color layer as part of the image (cfr Fig-4.5). Generally, neural network layers use matrix multiplication to describe the interaction between the input and output unit, as explained in the previous section. This means that every output unit interacts with every input unit. However, convolution neural networks use scattered or sparse interaction. As such, the resolution of the image can vary from hundreds to millions of pixels, whereas the kernel dimension can vary from tens to hundreds of pixels, resulting in fewer parameters and therefore reducing the memory requirements.

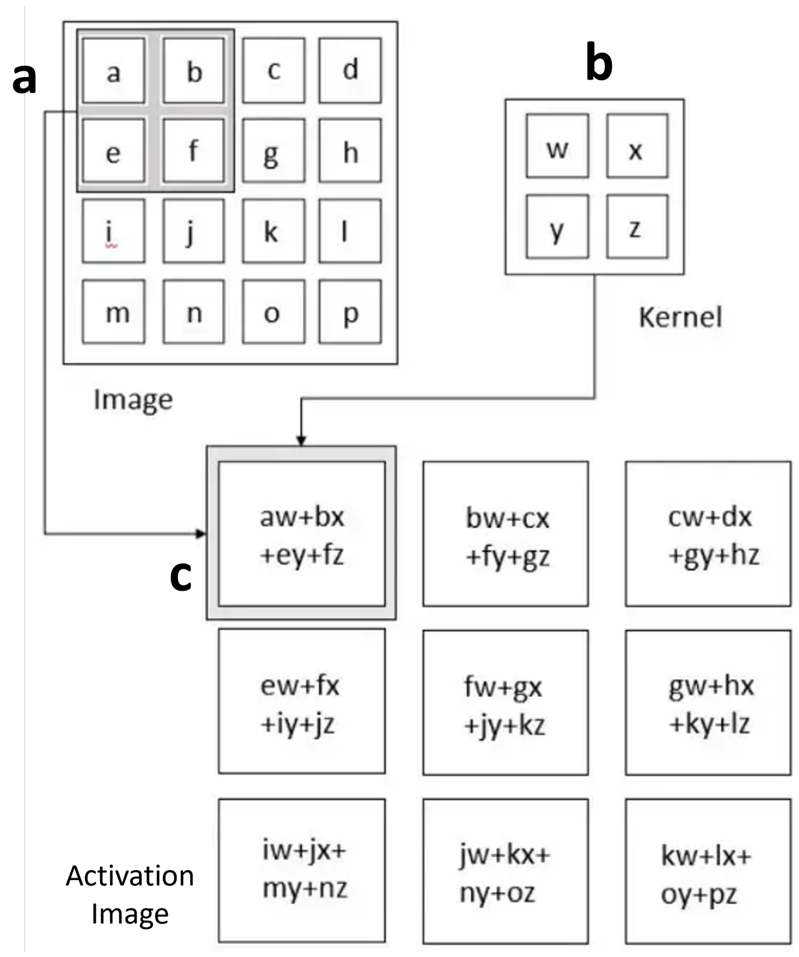


Fig. 4.4 Convolutional layer expression between (a) the image and (b) the example kernel. The result of the activation map is shown in panel (c).

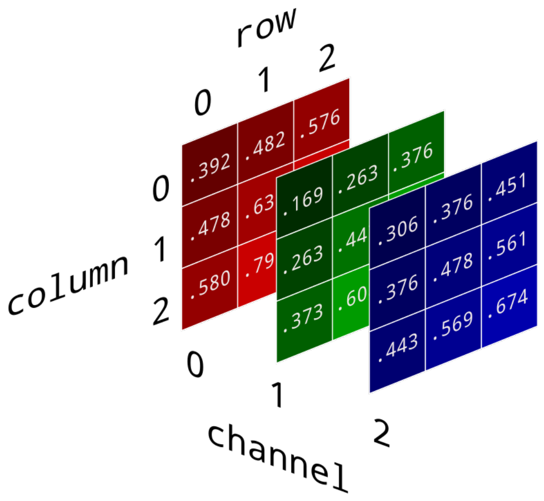


Fig. 4.5 An example of image separation into the three primary colors: Red, Green, Blue

Pooling layer

The pooling layer replaces the network's output at specific locations by deriving a summary statistic of the nearby outcomes. Pooling operations help reduce the representation's spatial size, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the image individually.

There are several pooling functions, such as the average of the rectangular neighborhood, the L2 norm of the rectangular neighbor area, and a weighted average based on the distance from the central pixel. However, researchers generally use the max pooling and average pooling, *i.e.* reporting the maximum output and average output from the neighborhood, respectively, as shown in Fig-4.6.

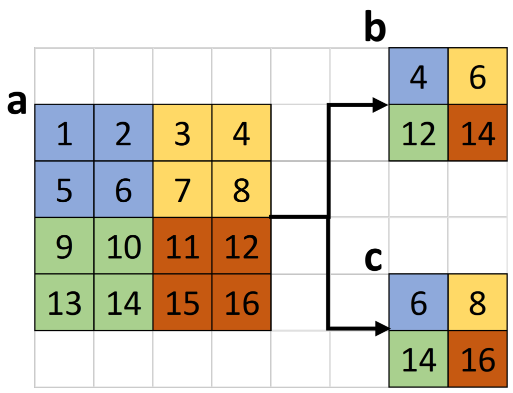


Fig. 4.6 The two different pooling operations: a) the original image; b) the average pooling; C) the max pooling.

4.2.5 Building and Training of CNN

In this work, the convolutional neural network-based image regression technique is used. We have used different popular deep learning (DL) models as mentioned below:

- Custom multi-layer perceptron model using TensorFlow [175].
- Residual neural network architectures ResNets with and without pre-trained model [55].
- VGG16 with improved (3x3) convolution filters [56]
- DenseNet which uses dense connections between layers through dense blocks [176].
- One of the classic DL algorithms: AlexNet [177]
- EfficientNet uniformly scales all dimensions of depth, width, and resolution using compound coefficients [178, 179].

Some other technical details related to the machine learning models are given in this paragraph. The custom architecture of CNN contains a multi-layer perceptron[180] with a batch normalization[181] layer followed by a dense neural network. For optimization, during the training of CNN, the default function Adam[182] is used, which is a first-order gradient-based optimization of stochastic objective functions. For the pre-trained ResNets models, we have used the Fastai[183] and PyTorch[184] libraries to train and predict the different magnetic properties from the simulated domains images. We have also compared the performances using different models, such as ResNet-18 and ResNet-34, comprising 18 and 34 layers, respectively. Overfitting is prevented by employing the default early-stopping[185] algorithm. The learning rate is set to 0.001 with a strategy to reduce the learning rate when the error stops decreasing with several steps. Furthermore, early-stopping is implemented in a way that it decides to stop training based on the accuracy improvement. The fit-one-cycle method was used for the dynamic learning rate[186], as implemented in PyTorch. We have also checked our model using pre-trained weights. In addition to the custom model, we have also used other models to verify advantages and disadvantages. For example, is "Deeper NN always better"? If that is the case, the performance of ResNet-34 should be more satisfactory than that of ResNet-18. Furthermore, can we run into the famous "vanishing gradient" problem in the VGG16 model?. The vanishing gradient problem refers to the (rare) situation in which the gradient will be relatively small, effectively preventing the weight from changing its value. As the network becomes denser by an increasing number of layers, we need to use certain activation functions for the neural networks, resulting in gradients of the loss function approaching zero, making the network hard to train. The latter situation is commonly known as the vanishing gradient problem [187].

Dataset Preparation

In order to create a large data set for training the neural network, we have used 960 simulated images, obtained by variations of four micromagnetic input parameters in realistic ranges *viz.,* A_{ex} (range: 10-230 pJ/m), α (range: 0.825-0.925), Ku_1 (range: within the order of $1k$-$100M\ J/m^3$), and T (range: 300-1000 K). Fig-4.7 shows some examples for different combinations of A_{ex} and Ku_1. Physically, the variation of A_{ex} and Ku_1 result in changes of the domain wall energy (proportional to $\sqrt{A_{ex} \times Ku_1}$). The variation in temperature introduces thermal motion, that results in fluctuations of magnetization near the boundary between two oppositely magnetized domains. α controls the relaxation of magnetization. One can see that a wide variety of domain patterns is captured. The considered range of parameters make the data sets versatile by incorporating various types of magnetic domains of different characteristic length scales and spatial features.

The entire data set was divided into three different groups, as mentioned below.

- The first group was used for training the model, using 63% (605) of the total images.
- The second group was used for validating and for the rearrangement of weights, using 7% (67) of the total images.
- The third group was used to test the model, using 30% (288) of the total images.

It is worth mentioning that the images used for validating were also used during the training process. However, the images used for testing the model were never exposed to the machine during training.

brief introduction of the CNN model

As shown by the simulated images reported in Fig-4.7, it is evident that a proper analysis of magnetic domain patterns can provide a route towards understanding the changes in the micromagnetic energetics. Thus, it would be desirable to establish a reliable path for a proper estimate of important parameters, which control the energetics and which we visualize in the form of magnetic domains. It is important to mention that recognizing the modifications in the micromagnetic parameters from the domain images becomes extremely difficult with human eyes, when the change in parameters does not lead to a significant change in the images, or domain patterns obtained by simultaneous changes of two or more parameters. In this regard, we propose different convolutional neural network architectures to identify the micromagnetic parameters from the domain images; this also helps immensely to complement the experimental results. As mentioned earlier, the training of CNN requires a large and homogeneous data set, which is not easy to obtain through the modifications of experimental process conditions. Therefore, in this project, we are using 960 domain images obtained through micromagnetic simulations considering different combinations of Ku_1, A_{ex}, T, and α . The tuning of domain size and net magnetization can arise from the modification of the above parameters, but

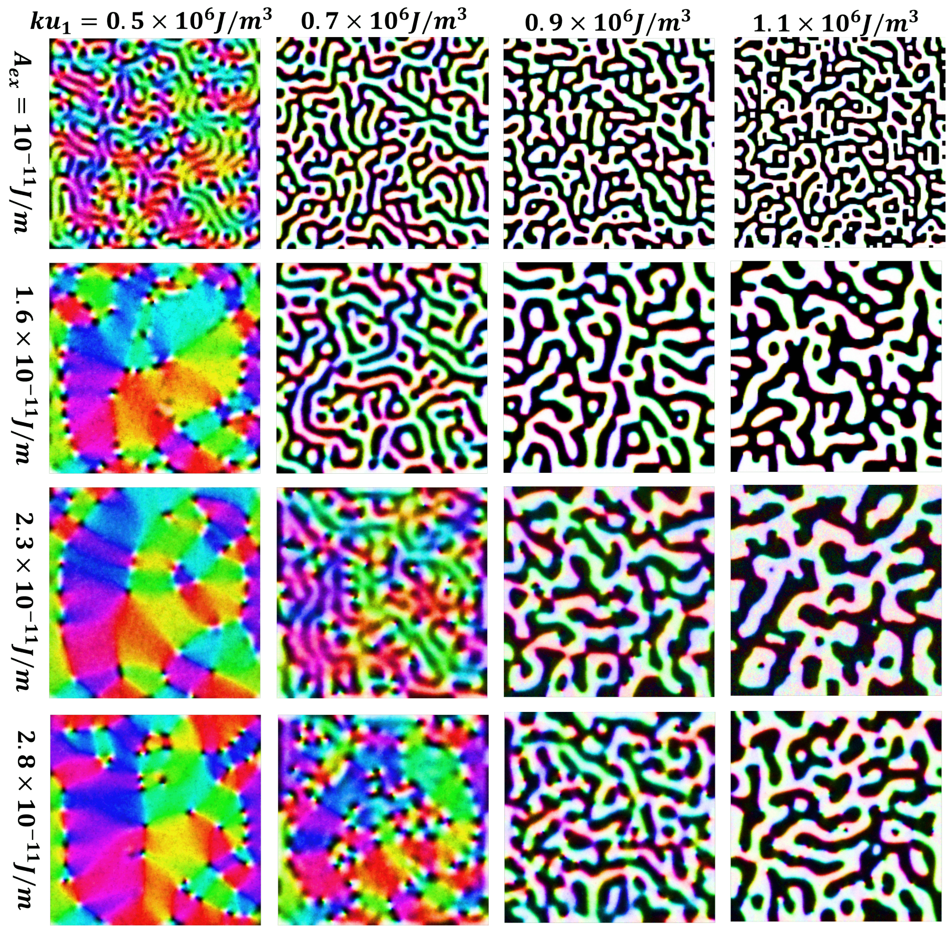


Fig. 4.7 Simulated domain images with different Ku_1 (reported in the top of the panel) with A_{ex} (reported on the left of the panel). The scale bar is the same for all the images, as shown in Fig-4.1.

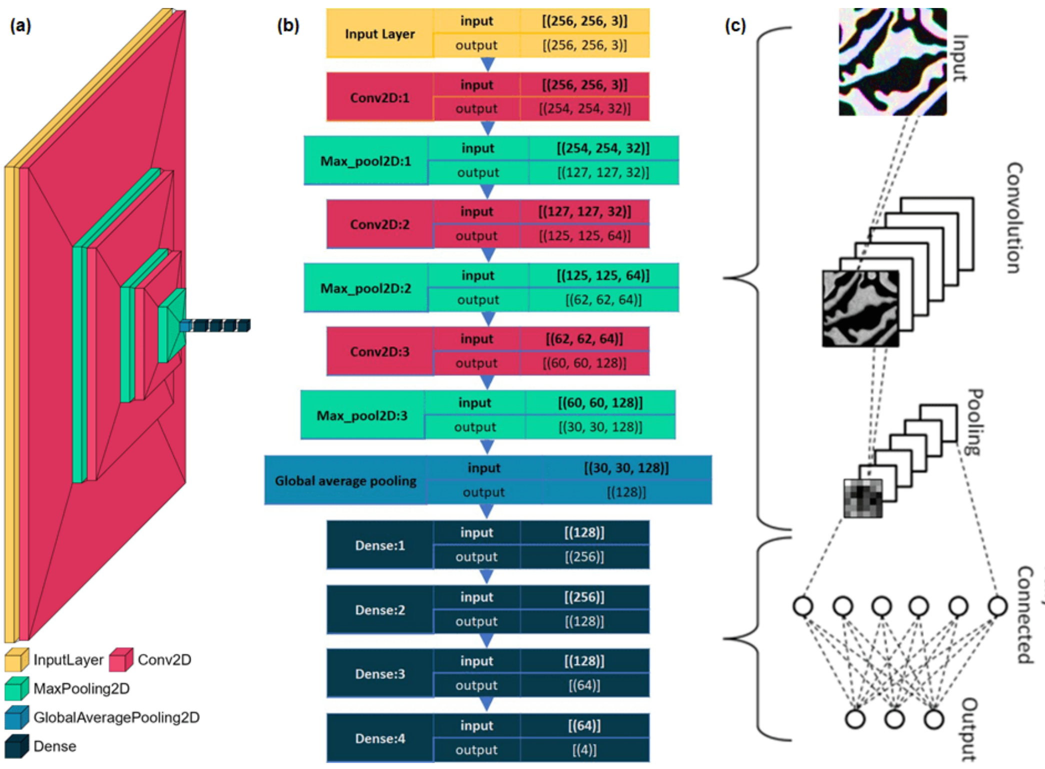

(a)
InputLayer
Conv2D
MaxPooling2D
GlobalAveragePooling2D
Dense
(b)
Input Layer input [(256, 256, 3)] output [(256, 256, 3)]
Conv2D:1 input [(256, 256, 3)] output [(254, 254, 32)]
Max_pool2D:1 input [(254, 254, 32)] output [(127, 127, 32)]
Conv2D:2 input [(127, 127, 32)] output [(125, 125, 64)]
Max_pool2D:2 input [(125, 125, 64)] output [(62, 62, 64)]
Conv2D:3 input [(62, 62, 64)] output [(60, 60, 128)]
Max_pool2D:3 input [(60, 60, 128)] output [(30, 30, 128)]
Global average pooling input [(30, 30, 128)] output [(128)]
Dense:1 input [(128)] output [(256)]
Dense:2 input [(256)] output [(128)]
Dense:3 input [(128)] output [(64)]
Dense:4 input [(64)] output [(4)]
(c)
Input
Convolution
Pooling
Fully Connected
Output

Fig. 4.8 Details of custom neural network with different descriptions: a) schematic for the reduction in dimension of the input image in each layer of CNN, b) flow chart of a custom CNN model, c) qualitative working principles of CNN.

pinpointing the exact parameters with higher accuracy from the entire parameters space is not straightforward. In order to reduce the cost of simulation time and to increase accuracy, we reduced the simulation area to around $1\mu m \times 1\mu m$. However, the CNN model is not limited by the size of the images, as we have confirmed by verifying our model with different sizes of images.

When dealing with images, we don't have the exact numerical values of the parameters mentioned above. Therefore, convolutional layers are used to extract those features from the images[173]. Thus, in addition to dense layers, the CNN model contains convolutional layers and pooling layers, which are used to extract important features from the images. The schematic for the distribution of different layers in a typical custom CNN is shown in Fig. 4.8(a), where three convolution layers and four dense layers were used. In order to extract the features, the convolution layer transforms the input image using different convoluted filters. A filter is a small matrix, with a dimension smaller than that of the image to be convoluted. A single convolution layer contains a series of filters, as shown in Fig. 4.8(c). Following the convolution layer (Conv2D), we have also used the pooling layer (MaxPooling2D) to reduce the dimensions of the feature maps (Fig. 4.8(a)); this is helpful to reduce the number of parameters to learn and the amount of computations performed in the network. A reduction in the lateral dimension of the input image can be observed after getting filtered through each layer. While the convolution layer increases the depth effect with a little reduction in the lateral dimension, the pooling layer only contributes to the reduction in the lateral dimension (cfr. Fig-.3 for the ResNet34 model). After getting filtered through a series of convolution and pooling layers, the images pass through the global average pooling layer and various dense layers. The dimension reduction chart is also shown for the ResNet-34 in the appendix-4. The sequence of different filters in our custom model and the corresponding changes in dimension at each layer is clearly illustrated in the flow chart of Fig. 4.8(b). The machine essentially understands and compares the 'higher dimensional images´, obtained after filtering through all the layers of the CNN model. The features, extracted using the filters, were further used for the prediction or classification. The entire process of working principles of CNN can be understood through the scheme of Fig. 4.8(c). However, in practice, substantially more convolutional and pooling layers are used in a CNN to extract different features from the images. In order to understand the effect of filters inside different convolution and pooling layers, we show an example of image regression in Fig. 4.9, illustrating the action of filtering three important features when the simulated domain image of Fig. 4.2(a) pass through different filters in different convolution and pooling layers. In Fig. 4.9(a), the first filter (top row) is extracting the distribution of the out-of-plane magnetization; the second filter (middle row) in Fig. 4.9(b) marks the domain walls (*i.e.* boundaries between two oppositely magnetized areas); the third filter (bottom) in Fig. 4.9(c) brings information about the domain curvature, which is physically related to domain nucleation and branching. However, the information extracted in the last layer (4^{th} Conv2D layer and higher) from a specific filter in Fig. 4.9 becomes complex to understand with human eyes. Several other filters were also used to produce this higher dimensional information which was further fed to the neural network to identify particular features inside the image where we use different functionalities mentioned above to predict the magnetic properties. It is important to notice that the lateral dimension of the images decreases from 256×256 to 15×15 after passing through all the filters, as observed from Fig. 4.9.

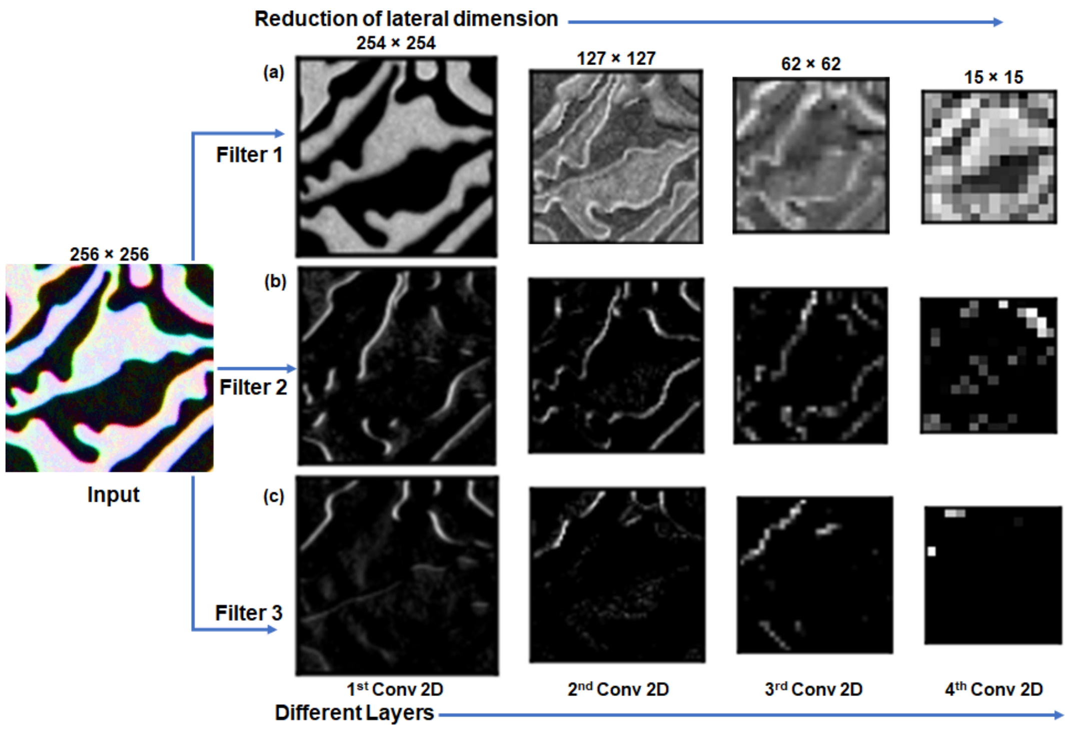


Fig. 4.9 Evolution of images with the application of different filters on the input image and feature extraction from different layers: a) distribution of out-of-plane magnetization, b) domain boundaries, c) curvature or branching.

Let us now focus on the training process of the CNN model. For the training, we use particular images, grouped with batch sizes of 32 images (the batch size decides the number of images trained at a time). We trained the CNN model on the entire training data set, also known as epochs. After training the model, we checked the prediction of the model on validated images and calculated the Mean squared error (MSE) for each epoch. Based on the errors, the model tries to adjust the weights to reduce the MSE. We have used different models for training and their comparative performances are shown in Fig. 4.10, where the MSE has been plotted at different epochs. A large MSE difference between the trained and validation data sets indicates over-fitting in the model. As observed from Fig. 4.10, the difference between the validation data set and the trained data set is small, implying that the models are learning different features from the images. Furthermore, the three different models are converging with comparable values of MSE (close to 5%) for higher epochs.

4.3 Results and Discussion

Comparison with Pre-trained CNN Architectures

We have constructed our custom CNN architecture and compared it with different available CNN architectures. Many numerical methods - such as Pearson correlation coefficient, Mean Squared Error (MSE), R^2 score, etc. - are used to check the accuracy of predictions. In our case, we have checked the accuracy of each model by the goodness parameter, *i.e.* the R^2 score, as defined below in equation 2.8. It is a measure of fit that indicates the variation of a dependent variable, expressed by the independent variable(s) in a regression model. Furthermore, we also report different R^2 scores for different methods in Table 4.1. Here, it is worth mentioning that reported R^2 scores are calculated using the test images, which were never exposed to the model during the training process.

$$R^2 = 1 - \frac{SS_{residual}}{SS_{total}} \tag{4.3}$$

In equation 2.8, $SS_{total} = \sum_i (y_i - \bar{y})^2$, $SS_{residual} = \sum_i (y_i - f_i)^2$ where f_i are the predicted values; x_i and y_i are values of input and output properties; $\bar{x}$ and $\bar{y}$ are the mean of the input and output values. In general, the values of R^2 score should lie between 0

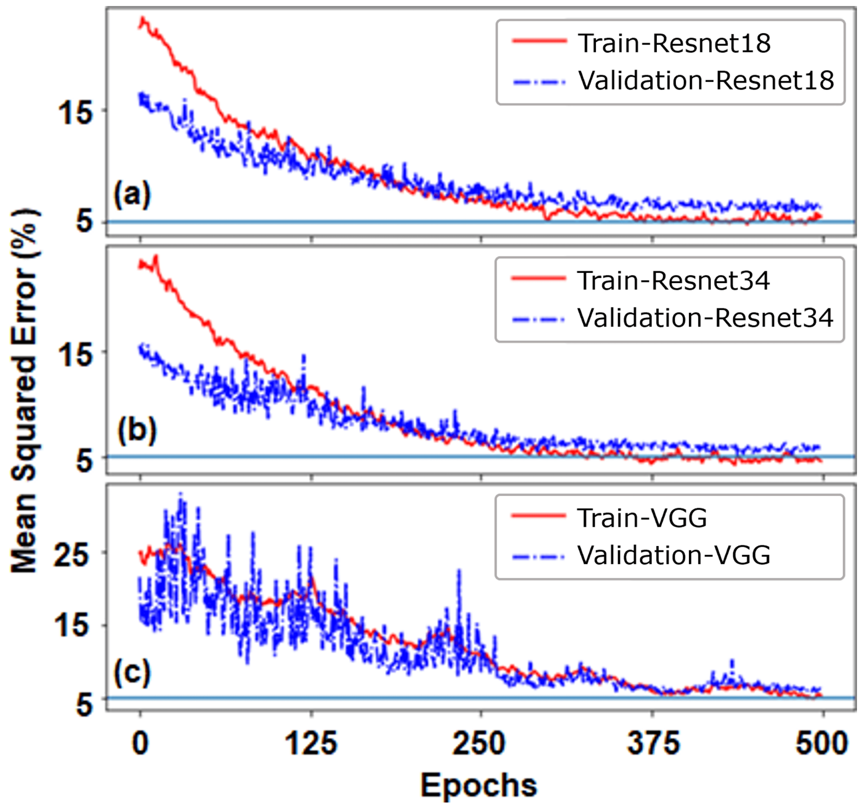


Fig. 4.10 Evolution of loss function for different CNN models as a function of epochs. Here loss for training and validation images are shown by solid and dashed lines, respectively. The blue solid line represents a reference level at 5%.

Table 4.1 Comparison of R^2 scores, and time of training for 1000 epochs (500 epochs with each of frozen and unfrozen weights) for different CNN models. The time for training is based on an Intel i5 quad core processor.

Name of the Model	R^2 score (%)	Training time ($sec \times 10^3$)
Custom Model[Fig. 4.8]	81.8	5
ResNet18	90.2	130
ResNet34	93.4	145
VGG16	93.9	72
EfficientNet	92.9	65
AlexNet	89.1	35
DenseNet	91.1	75

to 1; 0 indicates no dependency of input parameters on output properties (minimum accuracy), while 1 shows a 100% dependence (maximum accuracy). Negative values of R^2 score are also possible, indicating the worst performance by selecting the mean value from the data set.

When referring to Table 4.1, we observe that the pre-trained models, which have a significantly large number of layers compared to the custom model, work comparatively better. We run each pre-trained model in two-parts scheme: 1) using the frozen (unchanged) weights, and 2) using the unfrozen weights, as suggested in the model manual[55]. This is a general practice to decrease the load on the computer and preserve the prefixed weights in pre-trained models; this is due to the fact that these CNN architectures have around half a million parameters to adjust. We can see a significant improvement, when we increase the layers from the custom model to ResNet18; after that, the improvement is not significant when increasing the number of layers in ResNet34. Larger number of layers generally capture higher-dimensional features from the images; however, above some threshold number of layers in the model, the possibility of over-fitting increases. In addition to that, we also didn't encounter the Vanishing gradient problem in the case of VGG16. We note that the VGG16 has turned out to be the best model for properties prediction.

4.4 Test on experimental MFM image: discussion and summary

As mentioned in this chapter, we have generated almost 960 images using micromagnetic simulations by changing the combinations of four input parameters [namely: Ku_1, exchange constant (A_{ex}), Temperature (T), and damping constant (α)]. Training, testing, and validating with the simulated domain images provided a maximum accuracy of 93.9% (cfr Table 4.1 of the present chapter). Let us now apply the previously trained model on an experimental MFM image. Fig-4.11(a) shows an experimental image (yellow scale), which we convert into the grey-scale Fig-4.11(b). Due to a different colour profile with respect to the simulated images - as shown in Fig-4.1- , we cannot use the yellow scale image to predict the parameters. Lastly, the simulated image - obtained with MuMax3 using the parameters predicted from CNN - is shown in Fig-4.11(c). We note that the colour information in the experimental MFM images differs from that of the simulated images. Moreover, the Fourier transform filter was applied to the original MFM images to eliminate the tip-induced artifacts. Despite the dissimilarities, we tried to compare the simulated and experimental images on the same footing by converting them into grayscale. In order to do that, we again trained the convolutional neural network (CNN) separately with the simulated domain images using the grayscale format. Now, the MFM image in Fig-4.11(a) of the present chapter was tested with the CNN and provided the following values of the magnetic parameters: $Ku_1 = 2.39\ MJ/m^3$, $A_{ex} = 196\ pJ/m$, $T = 316\ K$, and $\alpha = 0.9$. The same set of values was used for the domain simulations. The results are presented in Fig-4.11, displaying the experimental (Fig-4.11(a), (b) and simulated domain images in the grayscale format (Fig-4.11(c)). The simulated domain image is qualitatively comparable with the experimental domain images in nanoscale, periodic, interconnected maze-like domains with comparable dimensions, the curling and branching of domains highlighting the probability for nucleation-dominated reversal and the presence of in-plane domain walls. As such, the proposed approach proved to be entirely satisfactory and promising.

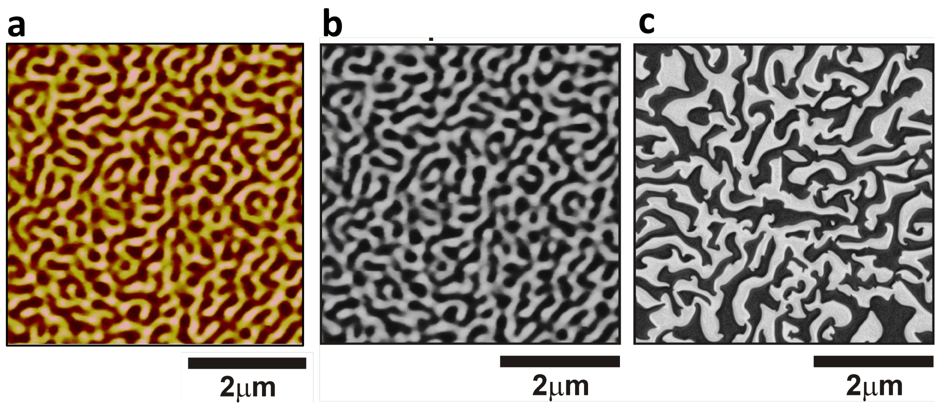


Fig. 4.11 Experimental and simulated domain images, obtained with the input parameters derived from the image regression algorithm, applied on the experimental domain image. **a**) shows the experimental image of the sample. **b**) shows the greyscale conversion of the image-a. **c**) shows the simulated image from the MuMax3 where different parameters were predicted using the trained CNN model on image-(b).

Chapter 5

Conclusion and Outlook

In conclusion, the three scientific studies presented above showcase the potential of machine learning in advancing our understanding of complex phenomena in materials science and technology.

The first study (Chapter-2) addresses the general problem of a compound crystallographic phase in the ground state and explores the use of machine learning to predict the energetic stability of different crystal structures in binary semiconductors. By building material features based on atomic properties and by training a machine learning model using density-functional-theory data, our study developed a formula that can quantitatively predict the energetic stability of one crystal structure over the other with high accuracy. Based on linear regression, the approach allows for a deeper physics understanding supported by machine-driven suggestions of relevant ingredients, offering a methodology with a huge range of applications in addressing microscopic mechanisms underlying different phenomena. Future research in this field could explore the application of our machine learning software to predict complex properties of ferroelectric oxide-based perovskites (for example Curie temperature, ferroelectric polarization or piezoelectric coefficients) as a function of descriptors based on atomic properties of the constituen elements on the A and B perovskite sites. We also note that our publicly-available software can be easily modified to include not only descriptors based on the constituents' atomic properties (as in the current version of the code), but also ingredients related to structural properties of the compound (*i.e.* lattice constants, bond lengths, bond angles) that might be important in the description of complex properties, such as ferroelectric or magnetic properties. We finally remark that integrating experimental data into machine learning models could significantly help in refining predictions and in expanding the range of applications.

Chapter-3 of this thesis presents a potential advancement in the field of neuromorphic computing, by introducing a spin wave-based device that functions as a physical neural network or neural network filter. The proposed device separates spin waves between two frequencies and goes beyond the solution presented in the literature, by using Genetic algorithms to optimize the defect-based configuration at a much faster pace. The results show the versatility of Genetic algorithms, which could be employed to simulate other devices (such as attenuators, three-frequencies separation, and digit identifiers), opening up new possibilities in neuromorphic devices. This study not only contributes to the computational design of spin wave-based devices, but also promises further developments from the experimental point of view. Our collaborators at Aalto University (Finland) are planning to fabricate the device to obtain real data and validate our computational predictions, paving the way for further research in this field. Future research can also explore other “handles", such as magnetic fields or magnetic anisotropy, to manipulate spin waves and create more dynamic neuromorphic devices, on the way to the exploration and design of innovative magnonic devices.

The third study (Chapter-4) investigates the modifications of magnetic domains in perpendicularly magnetized Co/Pd multi-layers, intending to improve the understanding of magnetic materials and their properties. The study focuses on developing an automated and reliable way to predict magnetic properties directly from a microscopic investigation using advanced machine learning algorithms in the form of a Convolutional Neural Network (CNN). By generating a large dataset of images to train the CNN through micromagnetic simulations, the study successfully predicts micromagnetic parameters with high accuracy, achieving maximum efficiency of 93.9%. The results of this study have several implications for the field of magnetic materials and their applications. Using machine learning algorithms, along with the automated prediction of magnetic properties, can lead to significant time and cost savings in identifying promising materials for specific applications, such as magnetic storage devices or sensors. In the future, the CNN model developed in this study can also be extended to more complex systems, including the prediction of novel magnetic materials with chiral domain walls or skyrmion–like spin textures, possibly manipulated using electrical currents. Overall, this study presents an innovative contribution to the field of ML in magnetic materials and its results hold promise for future technological advancements.

In summary, these studies demonstrate the strong potential of machine learning in advancing our understanding of complex phenomena in materials science and technology at different length scales and for different purposes. The insights gained from these studies have the potential to lead to the design of novel materials and devices with improved performances and functionalities. It is clear that the potential applications of versatile ML methods in materials science are vast and, upon further research and developments, we can undoubtedly expect to see significant progresses in the years to come.

References

[1] Koichi Momma and Fujio Izumi. *VESTA*: a three-dimensional visualization system for electronic and structural analysis. *J. Appl. Crystallogr.*, 41(3):653–658, Jun 2008.

[2] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864–B871, Nov 1964.

[3] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133–A1138, Nov 1965.

[4] Luca M. Ghiringhelli, Jan Vybiral, Sergey V. Levchenko, Claudia Draxl, and Matthias Scheffler. Big Data of Materials Science: Critical Role of the Descriptor. *Phys. Rev. Lett.*, 114(10):105503, March 2015.

[5] Quantum Transport in Graphene Atomic-sized Contacts. https://rua.ua.es/dspace/bitstream/10045/109594/1/Quantum_transport_in_graphene_nanocontacts_Del_Castillo_Hernandez_Yelko.pdf. Author: Yelko del Castillo Hern´andez.

[6] Hongwei Tan, Yifan Zhou, Quanzheng Tao, Johanna Rosen, and Sebastiaan van Dijken. Bioinspired multisensory neural network with crossmodal integration and recognition. *Nat Commun*, 12(1):1120, December 2021.

[7] Understanding Autoencoders - An Unsupervised Learning approach. https://ai-pool.com/a/s/understanding-autoencoders---an-unsupervised-learning-approach. Accessed: 2022-09-30.

[8] What is a Genetic Algorithm? https://www.generativedesign.org/02-deeper-dive/02-04_genetic-algorithms/02-04-01_what-is-a-genetic-algorithm. Accessed: 2022-09-30.

[9] Pablo Ruiz. Understanding and visualizing resnets. https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8, October 2018.

[10] Is it required to have predefined Image size to use transfer learning in Tensorflow? https://stackoverflow.com/questions/56267652/is-it-required-to-have-predefined-image-size-to-use-transfer-learning-in-te Author: Dennis Chicaiza.

[11] Hitarth Choubisa, Md Azimul Haque, Tong Zhu, Lewei Zeng, Maral Vafaie, Derya Baran, and Edward H Sargent. Closed-loop error correction learning accelerates experimental discovery of thermoelectric materials, 2023.

[12] Ahmed Shoyeb Raihan and Imtiaz Ahmed. Guiding the sequential experiments in autonomous experimentation platforms through ei-based bayesian optimization and bayesian model averaging, 2023.

[13] Nikolaos N. Vlassis and WaiChing Sun. Denoising diffusion algorithm for inverse design of microstructures with fine-tuned nonlinear material properties, 2023.

[14] Julian Lißner and Felix Fritzen. Hybrid machine-learned homogenization: Bayesian data mining and convolutional neural networks, 2023.

[15] Jonas Linkerhägner, Niklas Freymuth, Paul Maria Scheikl, Franziska Mathis-Ullrich, and Gerhard Neumann. Grounding graph network simulators using physical sensor observations, 2023.

[16] Fred X. Han, Keith G. Mills, Fabian Chudak, Parsa Riahi, Mohammad Salameh, Jialin Zhang, Wei Lu, Shangling Jui, and Di Niu. A general-purpose transferable predictor for neural architecture search, 2023.

[17] Yigitcan Comlek, Thang Duc Pham, Randall Snurr, and Wei Chen. Rapid design of top-performing metal-organic frameworks with qualitative representations of building blocks, 2023.

[18] G. H. Teichert, S. Das, M. Faghih Shojaei, J. Holber, T. Mueller, L. Hung, V. Gavini, and K. Garikipati. Bridging scales with machine learning: From first principles statistical mechanics to continuum phase field computations to study order disorder transitions in lixcoo2, 2023.

[19] Kishalay Das, Bidisha Samanta, Pawan Goyal, Seung-Cheol Lee, Satadeep Bhattacharjee, and Niloy Ganguly. Crysgnn : Distilling pre-trained knowledge to enhance property prediction for crystalline materials, 2023.

[20] Prathik R Kaundinya, Kamal Choudhary, and Surya R. Kalidindi. Prediction of the electron density of states for crystalline compounds with atomistic line graph neural networks (alignn), 2022.

[21] Lenz Fiedler, Karan Shah, Michael Bussmann, and Attila Cangi. A deep dive into machine learning density functional theory for materials science and chemistry, 2021.

[22] Sadman Sadeed Omee, Steph-Yves Louis, Nihang Fu, Lai Wei, Sourin Dey, Rongzhi Dong, Qinyang Li, and Jianjun Hu. Scalable deeper graph neural networks for high-performance materials property prediction, 2021.

[23] Brian C. Barnes, Daniel C. Elton, Zois Boukouvalas, DeCarlos E. Taylor, William D. Mattson, Mark D. Fuge, and Peter W. Chung. Machine learning of energetic material properties, 2018.

[24] Lars Hulstaert. Interpreting machine learning models. https://towardsdatascience.com/interpretability-in-machine-learning-70c30694a05f, Feb 2018.

[25] H J Kulik, T Hammerschmidt, J Schmidt, S Botti, M A L Marques, M Boley, M Scheffler, M Todorović, P Rinke, C Oses, A Smolyanyuk, S Curtarolo, A Tkatchenko, A P Bartók, S Manzhos, M Ihara, T Carrington, J Behler, O Isayev, M Veit, A Grisafi, J Nigam, M Ceriotti, K T Schütt, J Westermayr, M Gastegger, R J Maurer, B Kalita, K Burke, R Nagai, R Akashi, O Sugino, J Hermann, F Noé, S Pilati, C Draxl, M Kuban, S Rigamonti, M Scheidgen, M Esters, D Hicks, C Toher, P V Balachandran, I Tamblyn, S Whitelam, C Bellinger, and L M Ghiringhelli. Roadmap on machine learning in electronic structure. *Electronic Structure*, 4(2):023004, aug 2022.

[26] Jason Brownlee. How much training data is required for machine learning? https://machinelearningmastery.com/much-training-data-required-machine-learning/, July 2017.

[27] Ritesh Ranjan. Overfitting and underfitting in machine learning. https://towardsdatascience.com/overfitting-and-underfitting-in-machine-learning-89738c58f610, April 2020.

[28] Jenny Benois-Pineau and Akka Zemmari. *Multi-faceted Deep Learning: Models and Data*. Springer Nature, 2021.

[29] Samprit Chatterjee and Jeffrey S. Simonoff. *Handbook of regression analysis*. Wiley, Hoboken, New Jersey, 2013.

[30] Jure Leskovec, Anand Rajaraman, and Jeffrey D Ullman. *Mining of Massive Datasets*. Stanford University, Stanford, 2010.

[31] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

[32] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Adv. Neural Inf. Process. Syst.*, 29, 2016.

[33] Vijini Mallawaarachchi. Introduction to genetic algorithms — including example code. https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3#:

~:text=A%20genetic%20algorithm%20is%20a,offspring%20of%20the%20next%20generation., July 2017.

[34] John P. Perdew and Yue Wang. Accurate and simple analytic representation of the electron-gas correlation energy. *Phys. Rev. B*, 45:13244–13249, Jun 1992.

[35] Roberto Peverati and Donald G. Truhlar. Exchange–correlation functional with good accuracy for both structural and energetic properties while depending only on the density and its gradient. *Journal of Chemical Theory and Computation*, 8(7):2310–2319, 2012. PMID: 26588964.

[36] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002.

[37] Anthony J Myles, Robert N Feudale, Yang Liu, Nathaniel A Woody, and Steven D Brown. An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 18(6):275–285, 2004.

[38] Erik van der Giessen, Peter A Schultz, Nicolas Bertin, Vasily V Bulatov, Wei Cai, Gábor Csányi, Stephen M Foiles, M G D Geers, Carlos González, Markus Hütter, Woo Kyun Kim, Dennis M Kochmann, Javier LLorca, Ann E Mattsson, Jörg Rottler, Alexander Shluger, Ryan B Sills, Ingo Steinbach, Alejandro Strachan, and Ellad B Tadmor. Roadmap on multiscale materials modeling, Mar 2020.

[39] Anjan Barman, Gianluca Gubbiotti, S. Ladak, A. O. Adeyeye, M. Krawczyk, J. Gräfe, C. Adelmann, S. Cotofana, A. Naeemi, V. I. Vasyuchka, B. Hillebrands, S. A. Nikitov, H. Yu, D. Grundler, A. V. Sadovnikov, A. A. Grachev, S. E. Sheshukova, J.-Y. Duquesne, M. Marangolo, G. Csaba, W. Porod, V. E. Demidov, S. Urazhdin, S. O. Demokritov, E. Albisetti, D. Petti, R. Bertacco, H. Schultheiss, V. V. Kruglyak, V. D. Poimanov, S. Sahoo, J. Sinha, H. Yang, M. Münzenberg, T. Moriyama, S. Mizukami, P. Landeros, R. A. Gallardo, G. Carlotti, J.-V. Kim, R. L. Stamps, R. E. Camley, B. Rana, Y. Otani, W. Yu, T. Yu, G. E. W. Bauer, C. Back, G. S. Uhrig, O. V. Dobrovolskiy, B. Budinska, H. Qin, S. van Dijken, A. V. Chumak, A. Khitun, D. E. Nikonov, I. A. Young, B. W. Zingsem, and M. Winklhofer. The 2021 Magnonics Roadmap. *J. Phys.: Condens. Matter*, 33(41):413001, August 2021. Publisher: IOP Publishing.

[40] Sean Molesky, Zin Lin, Alexander Y. Piggott, Weiliang Jin, Jelena Vucković, and Alejandro W. Rodriguez. Inverse design in nanophotonics, Oct 2018.

[41] Andreas Vogelsang and Markus Borg. Requirements engineering for machine learning: Perspectives from data scientists, 2019.

[42] Bharati Tudu and Ashutosh Tiwari. Recent developments in perpendicular magnetic anisotropy thin films for data storage applications, Dec 2017.

[43] Reza Ranjbar, Kazuya Z. Suzuki, Yuta Sasaki, Lakhan Bainsla, and Shigemi Mizukami. Current-induced spin–orbit torque magnetization switching in a mnga/pt film with a perpendicular magnetic anisotropy, Nov 2016.

[44] R. Sbiaa, H. Meng, and S. N. Piramanayagam. Materials with perpendicular magnetic anisotropy for magnetic random access memory, Oct 2011.

[45] Atsuo Ono, Kazuya Z. Suzuki, Reza Ranjbar, Atsushi Sugihara, and Shigemi Mizukami. Ultrathin films of polycrystalline mnga alloy with perpendicular magnetic anisotropy, Jan 2017.

[46] Yuki Hibino, Tomohiro Koyama, Aya Obinata, Kazumoto Miwa, Shimpei Ono, and Daichi Chiba. Electric field modulation of magnetic anisotropy in perpendicularly magnetized pt/co structure with a pd top layer, Oct 2015.

[47] Larysa Tryputen, Feng Guo, Frank Liu, T. N. Anh Nguyen, Majid S. Mohseni, Sunjae Chung, Yeyu Fang, Johan Åkerman, R. D. McMichael, and Caroline A. Ross. Magnetic structure and anisotropy of, Jan 2015.

[48] Gabriel D. Chaves-O'Flynn, Georg Wolf, Jonathan Z. Sun, and Andrew D. Kent. Thermal stability of magnetic states in circular thin-film nanomagnets with large perpendicular magnetic anisotropy. *Phys. Rev. Appl.*, 4:024010, Aug 2015.

[49] P.M. Paulus, F. Luis, M. Kröll, G. Schmid, and L.J. de Jongh. Low-temperature study of the magnetization reversal and magnetic anisotropy of fe, ni, and co nanowires, Mar 2001.

[50] A. Talapatra, J. Arout Chelvane, and J. Mohanty. Engineering perpendicular magnetic anisotropy in tb-fe-co thin films using ion-beam irradiation, Apr 2021.

[51] Ajit Kumar Sahoo, Abhishek Talapatra, Jeyaramane Arout Chelvane, and Jyoti Mohanty. Modification of magnetic properties in tb–fe/gd–fe/tb–fe trilayer using ion-beam irradiation, Feb 2022.

[52] A. Talapatra, J. Arout Chelvane, B. Satpati, S. Kumar, and J. Mohanty. Tunable magnetic domains and depth resolved microstructure in gd-fe thin films, Feb 2019.

[53] A. Talapatra and J. Mohanty. Role of patterning induced defect on the switching field in magnetic nanostructure, Aug 2016.

[54] Cui-Lan Ren, Yang Yang, Yong-Gang Li, Ping Huai, Zhi-Yuan Zhu, and Ju Li. Sample spinning to mitigate polarization artifact and interstitial-vacancy imbalance in ion-beam irradiation, Dec 2020.

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. Technical Report arXiv:1512.03385, arXiv, December 2015. arXiv:1512.03385 [cs] version: 1 type: article.

[56] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-scale Image Recognition. Technical Report arXiv:1409.1556, arXiv, April 2015. arXiv:1409.1556 [cs] version: 6 type: article.

[57] Gordon E Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8), 1965.

[58] Rampi Ramprasad, Rohit Batra, Ghanshyam Pilania, Arun Mannodi-Kanakkithodi, and Chiho Kim. Machine learning in materials informatics: recent applications and prospects. *npj Comput. Mater.*, 3(1):54, December 2017.

[59] Masahiro Fukuda, Jingning Zhang, Yung-Ting Lee, and Taisuke Ozaki. A structure map for AB $_2$ type 2D materials using high-throughput DFT calculations. *Mater. Adv.*, 2(13):4392–4413, 2021.

[60] Daniel Schwalbe-Koda, Soonhyoung Kwon, Cecilia Paris, Estefania Bello-Jurado, Zach Jensen, Elsa Olivetti, Tom Willhammar, Avelino Corma, Yuriy Román-Leshkov, Manuel Moliner, and Rafael Gómez-Bombarelli. A priori control of zeolite phase competition and intergrowth with high-throughput simulations. *Science*, 374(6565):308–315, 2021.

[61] Eric R. Homer. High-throughput simulations for insight into grain boundary structure-property relationships and other complex microstructural phenomena. *Comput. Mater. Sci.*, 161:244–254, 2019.

[62] Stefano Curtarolo, Gus L. W. Hart, Marco Buongiorno Nardelli, Natalio Mingo, Stefano Sanvito, and Ohad Levy. The high-throughput highway to computational materials design. *Nat. Mate.*, 12(3):191–201, March 2013.

[63] M. L. Green, C. L. Choi, J. R. Hattrick-Simpers, A. M. Joshi, I. Takeuchi, S. C. Barron, E. Campo, T. Chiang, S. Empedocles, J. M. Gregoire, A. G. Kusne, J. Martin, A. Mehta, K. Persson, Z. Trautt, J. Van Duren, and A. Zakutayev. Fulfilling the promise of the materials genome initiative with high-throughput experimental methodologies. *Appl. Phys. Rev.*, 4(1):011105, March 2017.

[64] Aron Walsh. The quest for new functionality. *Nat. Chem*, 7(4):274–275, April 2015.

[65] Jiahong Shen, Vinay I. Hegde, Jiangang He, Yi Xia, and Chris Wolverton. High-Throughput Computational Discovery of Ternary Mixed-Anion Oxypnictides. *Chem. Mater.*, 33(24):9486–9500, December 2021.

[66] Sean D. Griesemer, Logan Ward, and Chris Wolverton. High-throughput crystal structure solution using prototypes. *Phys. Rev. Mater.*, 5(10):105003, October 2021.

[67] James E. Saal, Scott Kirklin, Muratahan Aykol, Bryce Meredig, and C. Wolverton. Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD). *JOM,*, 65(11):1501–1509, November 2013.

[68] Scott Kirklin, James E. Saal, Bryce Meredig, Alex Thompson, Jeff W. Doak, Muratahan Aykol, Stephan Rühl, and Chris Wolverton. The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies. *NPJ Comput. Mater.*, 1(1):1–15, December 2015.

[69] Claudia Draxl and Matthias Scheffler. NOMAD: The FAIR concept for big data-driven materials science. *MRS Bull.*, 43(9):676–682, September 2018.

[70] Claudia Draxl and Matthias Scheffler. The NOMAD laboratory: from data sharing to artificial intelligence. *J. Phys. Mater.*, 2(3):036001, July 2019.

[71] Stefano Curtarolo, Wahyu Setyawan, Shidong Wang, Junkai Xue, Kesong Yang, Richard H. Taylor, Lance J. Nelson, Gus L.W. Hart, Stefano Sanvito, Marco Buongiorno-Nardelli, Natalio Mingo, and Ohad Levy. AFLOWLIB.ORG: A distributed materials properties repository from high-throughput ab initio calculations. *Comput. Mater. Sci.*, 58:227–235, June 2012.

[72] Morten Niklas Gjerding, Alireza Taghizadeh, Asbjørn Rasmussen, Sajid Ali, Fabian Bertoldo, Thorsten Deilmann, Nikolaj Rørbæk Knøsgaard, Mads Kruse, Ask Hjorth Larsen, Simone Manti, Thomas Garm Pedersen, Urko Petralanda, Thorbjørn Skovhus, Mark Kamper Svendsen, Jens Jørgen Mortensen, Thomas Olsen, and Kristian Sommer Thygesen. Recent progress of the Computational 2D Materials Database (C2DB). *2D Mater.*, 8(4):044002, October 2021.

[73] Sten Haastrup, Mikkel Strange, Mohnish Pandey, Thorsten Deilmann, Per S Schmidt, Nicki F Hinsche, Morten N Gjerding, Daniele Torelli, Peter M Larsen, Anders C Riis-Jensen, Jakob Gath, Karsten W Jacobsen, Jens Jørgen Mortensen, Thomas Olsen, and Kristian S Thygesen. The Computational 2D Materials Database: high-throughput modeling and discovery of atomically thin crystals. *2D Mater.*, 5(4):042002, September 2018.

[74] Fabian Bertoldo, Sajid Ali, Simone Manti, and Kristian S. Thygesen. Quantum point defects in 2D materials: The QPOD database. *arXiv:2110.01961 [cond-mat, physics]*, October 2021.

[75] Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL Mater.*, 1(1):011002, July 2013.

[76] Leopold Talirz, Snehal Kumbhar, Elsa Passaro, Aliaksandr V. Yakutovich, Valeria Granata, Fernando Gargiulo, Marco Borelli, Martin Uhrin, Sebastiaan P. Huber, Spyros Zoupanos, Carl S. Adorf, Casper Welzel Andersen, Ole Schütt, Carlo A. Pignedoli, Daniele Passerone, Joost VandeVondele, Thomas C. Schulthess, Berend Smit, Giovanni Pizzi, and Nicola Marzari. Materials Cloud, a platform for open computational science. *Sci. Data*, 7(1):299, December 2020.

[77] Giovanni Pizzi, Andrea Cepellotti, Riccardo Sabatini, Nicola Marzari, and Boris Kozinsky. AiiDA: automated interactive infrastructure and database for computational science. *Comput. Mater. Sci.*, 111:218–230, January 2016.

[78] Sebastiaan P. Huber, Emanuele Bosoni, Marnik Bercx, Jens Bröder, Augustin Degomme, Vladimir Dikan, Kristjan Eimre, Espen Flage-Larsen, Alberto Garcia, Luigi Genovese, Dominik Gresch, Conrad Johnston, Guido Petretto, Samuel Poncé, Gian-Marco Rignanese, Christopher J. Sewell, Berend Smit, Vasily Tseplyaev, Martin Uhrin, Daniel Wortmann, Aliaksandr V. Yakutovich, Austin Zadoks, Pezhman Zarabadi-Poor, Bonan Zhu, Nicola Marzari, and Giovanni Pizzi. Common workflows for computing material properties using different quantum engines. *NPJ Comput. Mater.*, 7(1):136, December 2021.

[79] Heesoo Park, Adnan Ali, Raghvendra Mall, Halima Bensmail, Stefano Sanvito, and Fedwa El-Mellouhi. Data-driven enhancement of cubic phase stability in mixed-cation perovskites. *Mach. Learn.: Sci. Technol.*, 2(2):025030, June 2021.

[80] Chiho Kim, Ghanshyam Pilania, and Ramamurthy Ramprasad. From Organized High-Throughput Data to Phenomenological Theory using Machine Learning: The Example of Dielectric Breakdown. *Chem. Mater.*, 28(5):1304–1311, March 2016.

[81] Evgenii Tsymbalov, Zhe Shi, Ming Dao, Subra Suresh, Ju Li, and Alexander Shapeev. Machine learning for deep elastic strain engineering of semiconductor electronic band structure and effective mass. *NPJ Comput. Mater.*, 7(1):1–10, May 2021.

[82] Christopher J. Bartel, Christopher Sutton, Bryan R. Goldsmith, Runhai Ouyang, Charles B. Musgrave, Luca M. Ghiringhelli, and Matthias Scheffler. New tolerance factor to predict the stability of perovskite oxides and halides. *Sci. Adv.*, 5(2):eaav0693, February 2019.

[83] Aaron Gilad Kusne, Tieren Gao, Apurva Mehta, Liqin Ke, Manh Cuong Nguyen, Kai-Ming Ho, Vladimir Antropov, Cai-Zhuang Wang, Matthew J. Kramer, Christian Long, and Ichiro Takeuchi. On-the-fly machine-learning for high-throughput experiments: search for rare-earth-free permanent magnets. *Sci. Rep*, 4(1):6367, September 2014.

[84] Hideomi Koinuma and Ichiro Takeuchi. Combinatorial solid-state chemistry of inorganic materials. *Nat. Mate.*, 3(7):429–438, July 2004.

[85] Simone Manti, Mark Kamper Svendsen, Nikolaj R. Knøsgaard, Peder M. Lyngby, and Kristian S. Thygesen. Predicting and machine learning structural instabilities in 2D materials. *arXiv:2201.08091 [cond-mat]*, January 2022.

[86] Kyoungdoc Kim, Logan Ward, Jiangang He, Amar Krishna, Ankit Agrawal, and C. Wolverton. Machine-learning-accelerated high-throughput materials screening: Discovery of novel quaternary Heusler compounds. *Phys. Rev. Mater.*, 2(12):123801, December 2018.

[87] Koushik Pal, Cheol Woo Park, Yi Xia, Jiahong Shen, and Chris Wolverton. Scale-invariant Machine-learning Model Accelerates the Discovery of Quaternary Chalcogenides with Ultralow Lattice Thermal Conductivity. *arXiv:2109.03751 [cond-mat]*, September 2021.

[88] Martin Kuban, Santiago Rigamonti, Markus Scheidgen, and Claudia Draxl. Density-of-states similarity descriptor for unsupervised learning from materials data. *arXiv:2201.02187 [cond-mat]*, January 2022.

[89] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in neural information processing systems*, 30, 2017.

[90] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[91] Kevin Gurney. *Introduction to Neural Networks*. UCL Press Limited, London, 1997.

[92] Tian Xie and Jeffrey C. Grossman. Crystal Graph Convolutional Neural Networks for an Accurate and Interpretable Prediction of Material Properties. *Phys. Rev. Lett.*, 120(14):145301, April 2018.

[93] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[94] Christoph Molnar. *Interpretable Machine Learning*. Christoph Molnar, 2 edition, 2022.

[95] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[96] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *J. Mach. Learn. Res.*, 20(177):1–81, 2019.

[97] Saad Al-Askaar and Marek Perkowski. A new approach to machine learning based on functional decomposition of multi-valued functions. In *2021 IEEE 51st International Symposium on Multiple-Valued Logic (ISMVL)*, pages 128–135. IEEE, 2021.

[98] Radwa Elshawi, Mouaz H Al-Mallah, and Sherif Sakr. On the interpretability of machine learning-based model for predicting hypertension. *BMC medical informatics and decision making*, 19(1):1–32, 2019.

[99] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

[100] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[101] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021.

[102] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8827–8836, 2018.

[103] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 31, 2018.

[104] Luca M Ghiringhelli. Interpretability of machine-learning models in physical sciences. *arXiv preprint arXiv:2104.10443*, 2021.

[105] B. Meredig, A. Agrawal, S. Kirklin, J. E. Saal, J. W. Doak, A. Thompson, K. Zhang, A. Choudhary, and C. Wolverton. Combinatorial screening for new materials in unconstrained composition space with machine learning. *Phys. Rev. B*, 89(9):094104, March 2014.

[106] Logan Ward, Alexander Dunn, Alireza Faghaninia, Nils E. R. Zimmermann, Saurabh Bajaj, Qi Wang, Joseph Montoya, Jiming Chen, Kyle Bystrom, Maxwell Dylla, Kyle Chard, Mark Asta, Kristin A. Persson, G. Jeffrey Snyder, Ian Foster, and Anubhav Jain. Matminer: An open source toolkit for materials data mining. *Comput. Mater. Sci.*, 152:60–69, 2018.

[107] Logan Ward, Ankit Agrawal, Alok Choudhary, and Christopher Wolverton. A general-purpose machine learning framework for predictin g properties of inorganic materials. *Npj Comput. Mater.*, 2(1):16028, August 2016.

[108] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, , New York, NY 10013-2473, USA, Cambridge, 2014.

[109] Guido Van Rossum and Fred L Drake Jr. *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[110] Loriano Storchi. Open source code. `https://github.com/lstorchi/matinformatics`, 2022.

[111] Wolfram Research, Inc. Mathematica, Version 13.0.0. Champaign, IL, 2021.

[112] Loriano Storchi. Mathematica notebook. startingtest.nb within `https://github.com/lstorchi/matinformatics`, 2022.

[113] Fuchang Gao and Lixing Han. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Comput Optim Appl*, 51(1):259–277, January 2012.

[114] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins studies in the mathematical sciences. The Johns Hopkins University Press, Baltimore, fourth edition edition, 2013.

[115] C. G. Broyden. The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 03 1970.

[116] Ron dembo, Stanley Eisenstat, and Trond Steihaug. Inexact Newton Methods. *SIAM J. Numer. Anal.*, 19:400–408, April 1982.

[117] G. Kresse and J. Hafner. Ab initio molecular dynamics for liquid metals. *Phys. Rev. B*, 47:558–561, Jan 1993.

[118] G. Kresse and J. Furthmüller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Comput. Mater. Sci.*, 6(1):15 – 50, 1996.

[119] G. Kresse and J. Furthmüller. Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *Phys. Rev. B*, 54:11169–11186, Oct 1996.

[120] Volker Blum, Ralf Gehrke, Felix Hanke, Paula Havu, Ville Havu, Xinguo Ren, Karsten Reuter, and Matthias Scheffler. Ab initio molecular simulations with numeric atom-centered orbitals. *Computer Physics Communications*, 180(11):2175–2196, 2009.

[121] C. Eckhardt, K. Hummer, and G. Kresse. Indirect-to-direct gap transition in strained and unstrained $\mathrm{sn}_x\mathrm{ge}_{1-x}$ alloys. *Phys. Rev. B*, 89:165201, Apr 2014.

[122] L. Bellaiche and David Vanderbilt. Virtual crystal approximation revisited: Application to dielectric and piezoelectric properties of perovskites. *Phys. Rev. B*, 61:7877–7882, Mar 2000.

[123] Danila Amoroso, Andrés Cano, and Philippe Ghosez. First-principles study of $(\mathrm{Ba,Ca})\mathrm{tio}_3$ and $\mathrm{Ba(Ti,Zr)o}_3$ solid solutions. *Phys. Rev. B*, 97:174108, May 2018.

[124] Wilhelm Kirch, editor. *Pearson's Correlation Coefficient*, pages 1090–1091. Springer Netherlands, Dordrecht, 2008.

[125] Lars Vegard. Die konstitution der mischkristalle und die raumfüllung der atome. *Zeitschrift für Physik*, 5:17–26, 1920.

[126] William B Levy and Victoria G. Calvert. Computation in the human cerebral cortex uses less than 0.2 watts yet this great expense is optimal when considering communication costs. *bioRxiv*, 2020.

[127] C. Prasad, K. W. Park, M. Chahal, I. Meric, S. R. Novak, S. Ramey, P. Bai, H.-Y. Chang, N. L. Dias, W. M. Hafez, C.-H. Jan, N. Nidhi, R. W. Olac-vaw, R. Ramaswamy, and C. Tsai. Transistor reliability characterization and comparisons for a 14 nm tri-gate technology optimized for system-on-chip and foundry platforms. In *2016 IEEE International Reliability Physics Symposium (IRPS)*, pages 4B–5–1–4B–5–8, 2016.

[128] Amna Shahid and Malaika Mushtaq. A Survey Comparing Specialized Hardware And Evolution In TPUs For Neural Networks. In *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pages 1–6, November 2020. ISSN: 2049-3630.

[129] Bivas Rana and YoshiChika Otani. Towards magnonic devices based on voltage-controlled magnetic anisotropy. *Commun Phys*, 2(1):1–12, August 2019. Number: 1 Publisher: Nature Publishing Group.

[130] Michael Schneider, David Breitbach, Rostyslav O. Serha, Qi Wang, Alexander A. Serga, Andrei N. Slavin, Vasyl S. Tiberkevich, Björn Heinz, Bert Lägel, Thomas Brächer, Carsten Dubs, Sebastian Knauer, Oleksandr V. Dobrovolskiy, Philipp Pirro, Burkard Hillebrands, and Andrii V. Chumak. Control of the bose-einstein condensation of magnons by the spin hall effect. *Phys. Rev. Lett.*, 127:237203, Dec 2021.

[131] Michael Schneider, David Breitbach, Rostyslav O. Serha, Qi Wang, Morteza Mohseni, Alexander A. Serga, Andrei N. Slavin, Vasyl S. Tiberkevich, Björn Heinz, Thomas Brächer, Bert Lägel, Carsten Dubs, Sebastian Knauer, Oleksandr V. Dobrovolskiy, Philipp Pirro, Burkard Hillebrands, and Andrii V. Chumak. Stabilization of a nonlinear magnonic bullet coexisting with a bose-einstein condensate in a rapidly cooled magnonic system driven by spin-orbit torque. *Phys. Rev. B*, 104:L140405, Oct 2021.

[132] Morteza Mohseni, Qi Wang, Majid Mohseni, Thomas Brächer, Burkard Hillebrands, and Philipp Pirro. Propagating magnetic droplet solitons as moveable nanoscale spin-wave sources with tunable direction of emission. *Phys. Rev. Applied*, 13:024040, Feb 2020.

[133] Q. Wang, T. Brächer, M. Mohseni, B. Hillebrands, V. I. Vasyuchka, A. V. Chumak, and P. Pirro. Nanoscale spin-wave wake-up receiver. *Applied Physics Letters*, 115(9):092401, 2019.

[134] Hiroshige Onoda, Tomohiro Nozaki, Shingo Tamaru, Takayuki Nozaki, and Shinji Yuasa. Enhancing voltage-controlled magnetic anisotropy in $\mathrm{fe}_{80}\mathrm{b}_{20}/\mathrm{MgO}/\mathrm{hfo}_2$ thin films by dielectric constant modulation. *Phys. Rev. Materials*, 6:104406, Oct 2022.

[135] N.G. Pugach, M.O. Safonchik, V.I. Belotelov, T. Ziman, and T. Champel. Superconducting spin valves based on a single spiral magnetic layer. *Phys. Rev. Applied*, 18:054002, Nov 2022.

[136] Hongwei Tan, Yifan Zhou, Quanzheng Tao, Johanna Rosen, and Sebastiaan van Dijken. Bioinspired multisensory neural network with crossmodal integration and recognition. *Nat Commun*, 12(1):1120, December 2021.

[137] Roman Verba, Mario Carpentieri, Giovanni Finocchio, Vasil Tiberkevich, and Andrei Slavin. Amplification and stabilization of large-amplitude propagating spin waves by parametric pumping. *Applied Physics Letters*, 112(4):042402, 2018.

[138] Roman Verba, Mario Carpentieri, Giovanni Finocchio, Vasil Tiberkevich, and Andrei Slavin. Amplification and stabilization of large-amplitude propagating spin waves by parametric pumping. *Applied Physics Letters*, 112(4):042402, 2018.

[139] Abdulqader Mahmoud, Florin Ciubotaru, Frederic Vanderveken, Andrii V. Chumak, Said Hamdioui, Christoph Adelmann, and Sorin Cotofana. Introduction to spin wave computing. *Journal of Applied Physics*, 128(16):161101, 2020.

[140] Frank Heussner, Giacomo Talmelli, Moritz Geilen, Björn Heinz, Thomas Brächer, Thomas Meyer, Florin Ciubotaru, Christoph Adelmann, Kei Yamamoto, Alexander A. Serga, Burkard Hillebrands, and Philipp Pirro. Experimental realization of a passive gigahertz frequency-division demultiplexer for magnonic logic networks. *physica status solidi (RRL) – Rapid Research Letters*, 14(4):1900695, 2020.

[141] T. Brächer and P. Pirro. An analog magnon adder for all-magnonic neurons. *Journal of Applied Physics*, 124(15):152119, 2018.

[142] Logan Su, Dries Vercruysse, Jinhie Skarda, Neil V. Sapra, Jan A. Petykiewicz, and Jelena Vučković. Nanophotonic inverse design with spins: Software architecture and practical considerations. *Applied Physics Reviews*, 7(1):011407, 2020.

[143] Arne Vansteenkiste, Jonathan Leliaert, Mykola Dvornik, Mathias Helsen, Felipe Garcia-Sanchez, and Bartel Van Waeyenberge. The design and verification of MuMax3. *AIP Adv.*, 4(10):107133, October 2014.

[144] Komal Saini. Natural language processing ft. siri. NaturalLanguageProcessingft.Siri, September 2019.

[145] Ming Sun. how alexa recognizes sounds. https://www.amazon.science/blog/two-new-papers-discuss-how-alexa-recognizes-sounds, April 2019.

[146] Isaac Caswell and Bowen Liang. Recent advances in google translate. https://ai.googleblog.com/2020/06/recent-advances-in-google-translate.html, August 2020.

[147] David Petersson. Cnn vs. rnn: How are they different? https://www.techtarget.com/searchenterpriseai/feature/CNN-vs-RNN-How-they-differ-and-where-they-overlap, March 2021.

[148] Mohammad Yazdani-Asrami, Alireza Sadeghi, Wenjuan Song, Ana Madureira, João Murta-Pina, Antonio Morandi, and Michael Parizh. Artificial intelligence methods for applied superconductivity: material, design, manufacturing, testing, operation, and condition monitoring. *Superconductor Science and Technology*, 35(12):123001, October 2022.

[149] Karthik Padavala, Avaneesh Singh, and Joyjit Kundu. Machine learned phase transitions in a system of anisotropic particles on a square lattice, 2021.

[150] Alessandra Celletti, Catalin Gales, Victor Rodriguez-Fernandez, and Massimiliano Vasile. Classification of regular and chaotic motions in hamiltonian systems with deep learning. *Scientific Reports*, 12(1), February 2022.

[151] Gousia Habib and Shaima Qureshi. Optimization and acceleration of convolutional neural networks: A survey. *Journal of King Saud University - Computer and Information Sciences*, 34(7):4244–4268, July 2022.

[152] Anna Malanushenko, Natasha Flyer, and Sarah Gibson. Convolutional neural networks for predicting the strength of the near-earth magnetic field caused by interplanetary coronal mass ejections. *Frontiers in Astronomy and Space Sciences*, 7, September 2020.

[153] Duo Xu, Chi-Yan Law, and Jonathan C. Tan. Application of convolutional neural networks to predict magnetic fields' directions in turbulent clouds. *The Astrophysical Journal*, 942(2):95, January 2023.

[154] Akinori Tanaka and Akio Tomiya. Detection of phase transition via convolutional neural networks. *Journal of the Physical Society of Japan*, 86(6):063001, jun 2017.

[155] D-R Tan, C-D Li, W-P Zhu, and F-J Jiang. A comprehensive neural networks study of the phase transitions of potts model. *New Journal of Physics*, 22(6):063016, June 2020.

[156] Amit K. Choudhary, Andreas Jansche, Tvrtko Grubesa, Florian Trier, Dagmar Goll, Timo Bernthaler, and Gerhard Schneider. Grain size analysis in permanent magnets from kerr microscopy images using machine learning techniques. *Materials Characterization*, 186:111790, April 2022.

[157] Ruoqian Liu, Abhishek Kumar, Zhengzhang Chen, Ankit Agrawal, Veera Sundararaghavan, and Alok Choudhary. A predictive machine learning approach for microstructure optimization and materials design. *Sci. Rep.*, 5(1):11551, September 2015.

[158] Valentin Mateev and Iliana Marinova. Machine Learning in Magnetic Field Calculations. In *2019 19th International Symposium on Electromagnetic Fields in Mechatronics, Electrical and Electronic Engineering (ISEF)*, pages 1–2, Nancy, France, August 2019. IEEE.

[159] Lijia Jiang, Lingxiao Wang, and Kai Zhou. Deep learning stochastic processes with qcd phase transition. *Phys. Rev. D*, 103(11):116023, June 2021.

[160] Amit K. Choudhary, Andreas Jansche, Tvrtko Grubesa, Florian Trier, Dagmar Goll, Timo Bernthaler, and Gerhard Schneider. Grain size analysis in permanent magnets from Kerr microscopy images using machine learning techniques. *Mater. Charact.*, 186:111790, April 2022.

[161] S Courtin and S Padovani. Analysis of magnetic domain patterns by a perceptron neural network. *Europhys. Lett.*, 50(1):94–100, April 2000.

[162] Naoya Mamada, Masaichiro Mizumaki, Ichiro Akai, and Toru Aonishi. Obtaining Underlying Parameters from Magnetic Domain Patterns with Machine Learning. *J. Phys. Soc. Jpn.*, 90(1):014705, January 2021.

[163] Tim Matthies, Alexander F. Schäffer, Thore Posske, Roland Wiesendanger, and Elena Y. Vedmedenko. Topological characterization of dynamic chiral magnetic textures using machine learning. Technical Report arXiv:2201.01629, arXiv, April 2022. arXiv:2201.01629 [cond-mat] type: article.

[164] D. B. Lee, H. G. Yoon, S. M. Park, J. W. Choi, H. Y. Kwon, and C. Won. Estimating the effective fields of spin configurations using a deep learning technique. *Sci. Rep.*, 11(1):22937, December 2021.

[165] Hee Young Kwon, Han Gyu Yoon, Sung Min Park, Doo Bong Lee, Jun Woo Choi, and Changyeon Won. Magnetic State Generation using Hamiltonian Guided Variational Autoencoder with Spin Structure Stabilization. *Adv. Sci.*, 8(11):2004795, June 2021.

[166] Pengju Wang, Jianpei Xing, Xue Jiang, and Jijun Zhao. Transition-metal interlink neural network: Machine learning of 2d metal–organic frameworks with high magnetic anisotropy. *ACS Appl. Mater. Interfaces*, 2022.

[167] Masashi Kawaguchi, Kenji Tanabe, Keisuke Yamada, Takuya Sawa, Shun Hasegawa, Masamitsu Hayashi, and Yoshinobu Nakatani. Determination of the Dzyaloshinskii-Moriya interaction using pattern recognition and machine learning. *npj Comput. Mater.*, 7(1):20, December 2021.

[168] Lars Lien Ankile, Morgan Feet Heggland, and Kjartan Krange. Deep Convolutional Neural Networks: a survey of the foundations, selected improvements, and some current applications. Technical Report arXiv:2011.12960, arXiv, November 2020. arXiv:2011.12960 [cs] type: article.

[169] H. Y. Kwon, H. G. Yoon, C. Lee, G. Chen, K. Liu, A. K. Schmid, Y. Z. Wu, J. W. Choi, and C. Won. Magnetic Hamiltonian parameter estimation using deep learning techniques. *Sci. Adv.*, 6(39):eabb0872, September 2020.

[170] Martin Stärk, Frank Schlickeiser, Dennis Nissen, Birgit Hebler, Philipp Graus, Denise Hinzke, Elke Scheer, Paul Leiderer, Mikhail Fonin, Manfred Albrecht, Ulrich Nowak, and Johannes Boneberg. Controlling the magnetic structure of co/pd thin films by direct laser interference patterning. *Nanotechnology*, 26(20):205302, 2015.

[171] LALE Landau and Evgeny Lifshitz. On the theory of the dispersion of magnetic permeability in ferromagnetic bodies. In *Perspectives in Theoretical Physics*, pages 51–65. Elsevier, 1992.

[172] Thomas L Gilbert. A lagrangian formulation of the gyromagnetic equation of the magnetization field. *Phys. Rev.*, 100:1243, 1955.

[173] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data*, 8(1):53, December 2021.

[174] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[175] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org.

[176] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. Technical Report arXiv:1608.06993, arXiv, January 2018. arXiv:1608.06993 [cs] version: 5 type: article.

[177] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, editors, *Adv Neural Inf Process Syst.*, volume 25. Curran Associates, Inc., 2012.

[178] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Technical Report arXiv:1905.11946, arXiv, September 2020. arXiv:1905.11946 [cs, stat] version: 5 type: article.

[179] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[180] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[181] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[182] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[183] Jeremy Howard and Sylvain Gugger. fastai: A layered api for deep learning. 2020.

[184] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[185] Maren Mahsereci, Lukas Balles, Christoph Lassner, and Philipp Hennig. Early stopping without a validation set, 2017.

[186] Suki Lau. Learning rate schedules and adaptive learning rate methods for deep learning. https://towardsdatascience.com/learning-rate-schedules-and-adaptive-learning-rate-methods\-for-deep-learning-2c8f433990d1, July 2017.

[187] Chi-Feng Wang. The vanishing gradient problem the problem, its causes, its significance, and its solutions. https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484, January 2019.

[188] Dev Rajnarayan and David Wolpert. *Bias-Variance Trade-offs: Novel Applications*, pages 101–110. Springer US, Boston, MA, 2010.

Appendix A

Additional Information

.1 Analysis of the best 1D, 2D and 3D formulas

Table-1 reports other verification parameters calculated for the best formulas of each generator, including those presented in Chapter-2. The relevance of calculating avg(RMSE train) and avg(RMSE test) lies in analysing the bias-variance tradeoff[188] in LR. Max_E and Min_E indicate the maximum and minimum error in the prediction for the specific formula.

	Details	avg(RSME train) (eV)	avg(RMSE test) (eV)	RMSE (eV)	R^2	Pearson_coeff	success rate	Max_E	Min_E	Std_deviation
0	Ref_1D[4]	0.1420	0.1455	0.1422	0.89	0.947	89%	0.0523	0.0041	0.0081
1	1D_GEN1	0.1186	0.1296	0.1192	0.92	0.963	90%	0.0743	0.0014	0.0083
2	1D_GEN2	0.1305	0.1367	0.1309	0.91	0.956	91%	0.0676	0.0027	0.0105
3	1D_GEN3	0.0961	0.0995	0.0963	0.95	0.976	94%	0.0234	0.0016	0.0032
4	1D_GEN4	0.1055	0.1103	0.1058	0.94	0.971	96%	0.0330	0.0012	0.0044
5	Ref_2D[4]	0.0983	0.1041	0.0987	0.95	0.975	96%	0.0323	0.0019	0.0044
6	2D_GEN1	0.0941	0.0988	0.0943	0.95	0.977	89%	0.0419	0.0020	0.0040
7	2D_GEN2	0.1095	0.1163	0.1099	0.93	0.969	87%	0.0489	0.0011	0.0083
8	2D_GEN3	0.0875	0.0911	0.0878	0.96	0.980	88%	0.0178	0.0014	0.0026
9	2D_GEN4	0.0951	0.0995	0.0954	0.95	0.977	93%	0.0221	0.0016	0.0033
10	Ref_3D[4]	0.0751	0.0814	0.0755	0.97	0.985	93%	0.0185	0.0009	0.0031
11	3D_GEN1	0.0929	0.1003	0.0933	0.95	0.978	90%	0.0282	0.0024	0.0038
12	3D_GEN2	0.1200	0.1300	0.1205	0.92	0.963	91%	0.1119	0.0021	0.0103
13	3D_GEN3	0.0832	0.0874	0.0834	0.96	0.982	98%	0.0199	0.0015	0.0026
14	3D_GEN4	0.0915	0.0989	0.0919	0.96	0.978	93%	0.0227	0.0013	0.0035

Table 1 Different verification parameters for 1D, 2D and 3D descriptors calculated in the present work and descriptors presented in Ref.[4]. Here, avg(RMSE train), avg(RMSE test) and RMSE indicate the root mean squared error for training data, test data and full dataset, respectively. R^2 and Pearson coeff are goodness parameters. Max_E and Min_E show the maximum and minimum absolute error in prediction.

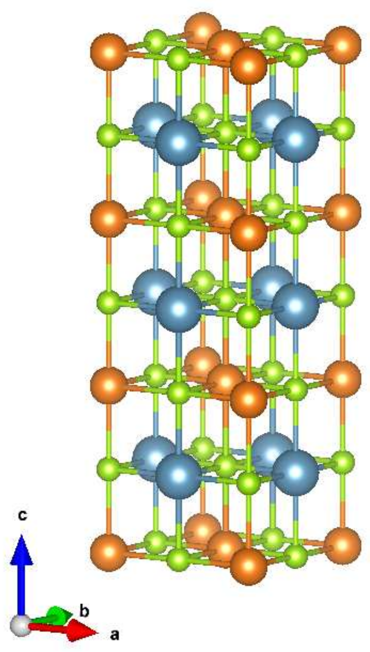


Fig. 1 $Mg_{0.5}Ca_{0.5}Se$ rocksalt supercell: Mg is reported in orange, Ca in blue and Se in green. The supercell is obtained alternating layers of Mg and Ca in the cation sub-lattice along the c primitive vector.

A	B	DFT Classification	ΔE	IP(A)	EA(A)	HOMO(A)	LUMO(A)	r_s(A)	r_p(A)	r_d(A)	IP(B)	EB(B)	HOMO(B)	LUMO(B)	r_s(B)	r_p(B)	r_d(B)
Li	F	RS	-0.059	-5.329	-0.698	-2.874	-0.978	1.652	1.995	6.930	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Li	Cl	RS	-0.038	-5.329	-0.698	-2.874	-0.978	1.652	1.995	6.930	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Li	Br	RS	-0.033	-5.329	-0.698	-2.874	-0.978	1.652	1.995	6.930	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Li	I	RS	-0.022	-5.329	-0.698	-2.874	-0.978	1.652	1.995	6.930	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Be	O	ZB	0.430	-9.459	0.631	-5.600	-2.098	1.078	1.211	2.877	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Be	S	ZB	0.506	-9.459	0.631	-5.600	-2.098	1.078	1.211	2.877	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Be	Se	ZB	0.495	-9.459	0.631	-5.600	-2.098	1.078	1.211	2.877	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Be	Te	ZB	0.466	-9.459	0.631	-5.600	-2.098	1.078	1.211	2.877	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
B	N	ZB	1.713	-8.190	-0.107	-3.715	2.248	0.805	0.826	1.946	-13.585	-1.867	-7.239	3.057	0.539	0.511	1.540
B	P	ZB	1.020	-8.190	-0.107	-3.715	2.248	0.805	0.826	1.946	-9.751	-1.920	-5.596	0.183	0.826	0.966	1.771
B	As	ZB	0.879	-8.190	-0.107	-3.715	2.248	0.805	0.826	1.946	-9.262	-1.839	-5.341	0.064	0.847	1.043	2.023
C	C	ZB	2.638	-10.852	-0.872	-5.416	1.992	0.644	0.630	1.631	-10.852	-0.872	-5.416	1.992	0.644	0.630	1.631
Na	F	RS	-0.146	-5.223	-0.716	-2.819	-0.718	1.715	2.597	6.566	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Na	Cl	RS	-0.133	-5.223	-0.716	-2.819	-0.718	1.715	2.597	6.566	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Na	Br	RS	-0.127	-5.223	-0.716	-2.819	-0.718	1.715	2.597	6.566	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Na	I	RS	-0.115	-5.223	-0.716	-2.819	-0.718	1.715	2.597	6.566	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Mg	O	RS	-0.178	-8.037	0.693	-4.782	-1.358	1.330	1.897	3.171	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Mg	S	RS	-0.087	-8.037	0.693	-4.782	-1.358	1.330	1.897	3.171	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Mg	Se	RS	-0.055	-8.037	0.693	-4.782	-1.358	1.330	1.897	3.171	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Mg	Te	RS	-0.005	-8.037	0.693	-4.782	-1.358	1.330	1.897	3.171	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
Al	N	ZB	0.072	-5.780	-0.313	-2.784	0.695	1.092	1.393	1.939	-13.585	-1.867	-7.239	3.057	0.539	0.511	1.540
Al	P	ZB	0.219	-5.780	-0.313	-2.784	0.695	1.092	1.393	1.939	-9.751	-1.920	-5.596	0.183	0.826	0.966	1.771
Al	As	ZB	0.212	-5.780	-0.313	-2.784	0.695	1.092	1.393	1.939	-9.262	-1.839	-5.341	0.064	0.847	1.043	2.023
Al	Sb	ZB	0.150	-5.780	-0.313	-2.784	0.695	1.092	1.393	1.939	-8.468	-1.847	-4.991	0.105	1.001	1.232	2.065
Si	C	ZB	0.668	-7.758	-0.993	-4.163	0.440	0.938	1.134	1.890	-10.852	-0.872	-5.416	1.992	0.644	0.630	1.631
Si	Si	ZB	0.275	-7.758	-0.993	-4.163	0.440	0.938	1.134	1.890	-7.758	-0.993	-4.163	0.440	0.938	1.134	1.890
K	F	RS	-0.146	-4.433	-0.621	-2.426	-0.697	2.128	2.443	1.785	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
K	Cl	RS	-0.165	-4.433	-0.621	-2.426	-0.697	2.128	2.443	1.785	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
K	Br	RS	-0.166	-4.433	-0.621	-2.426	-0.697	2.128	2.443	1.785	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
K	I	RS	-0.168	-4.433	-0.621	-2.426	-0.697	2.128	2.443	1.785	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Ca	O	RS	-0.266	-6.428	0.304	-3.864	-2.133	1.757	2.324	0.679	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Ca	S	RS	-0.369	-6.428	0.304	-3.864	-2.133	1.757	2.324	0.679	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Ca	Se	RS	-0.361	-6.428	0.304	-3.864	-2.133	1.757	2.324	0.679	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Ca	Te	RS	-0.350	-6.428	0.304	-3.864	-2.133	1.757	2.324	0.679	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
Cu	F	RS	-0.019	-8.389	-1.638	-4.856	-0.641	1.197	1.680	2.576	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Cu	Cl	ZB	0.156	-8.389	-1.638	-4.856	-0.641	1.197	1.680	2.576	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Cu	Br	ZB	0.152	-8.389	-1.638	-4.856	-0.641	1.197	1.680	2.576	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Cu	I	ZB	0.203	-8.389	-1.638	-4.856	-0.641	1.197	1.680	2.576	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Zn	O	ZB	0.102	-10.136	1.081	-6.217	-1.194	1.099	1.547	2.254	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Zn	S	ZB	0.275	-10.136	1.081	-6.217	-1.194	1.099	1.547	2.254	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Zn	Se	ZB	0.259	-10.136	1.081	-6.217	-1.194	1.099	1.547	2.254	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Zn	Te	ZB	0.241	-10.136	1.081	-6.217	-1.194	1.099	1.547	2.254	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
Ga	N	ZB	0.433	-5.818	-0.108	-2.732	0.130	0.994	1.330	2.163	-13.585	-1.867	-7.239	3.057	0.539	0.511	1.540
Ga	P	ZB	0.341	-5.818	-0.108	-2.732	0.130	0.994	1.330	2.163	-9.751	-1.920	-5.596	0.183	0.826	0.966	1.771
Ga	As	ZB	0.271	-5.818	-0.108	-2.732	0.130	0.994	1.330	2.163	-9.262	-1.839	-5.341	0.064	0.847	1.043	2.023
Ga	Sb	ZB	0.158	-5.818	-0.108	-2.732	0.130	0.994	1.330	2.163	-8.468	-1.847	-4.991	0.105	1.001	1.232	2.065
Ge	Ge	ZB	0.202	-7.567	-0.949	-4.046	2.175	0.917	1.162	2.373	-7.567	-0.949	-4.046	2.175	0.917	1.162	2.373
Rb	F	RS	-0.136	-4.289	-0.590	-2.360	-0.705	2.240	3.199	1.960	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Rb	Cl	RS	-0.161	-4.289	-0.590	-2.360	-0.705	2.240	3.199	1.960	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Rb	Br	RS	-0.164	-4.289	-0.590	-2.360	-0.705	2.240	3.199	1.960	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Rb	I	RS	-0.169	-4.289	-0.590	-2.360	-0.705	2.240	3.199	1.960	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Sr	O	RS	-0.221	-6.032	0.343	-3.641	-1.379	1.911	2.548	1.204	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Sr	S	RS	-0.369	-6.032	0.343	-3.641	-1.379	1.911	2.548	1.204	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Sr	Se	RS	-0.375	-6.032	0.343	-3.641	-1.379	1.911	2.548	1.204	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Sr	Te	RS	-0.381	-6.032	0.343	-3.641	-1.379	1.911	2.548	1.204	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
Ag	F	RS	-0.156	-8.058	-1.667	-4.710	-0.479	1.316	1.883	2.968	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Ag	Cl	RS	-0.044	-8.058	-1.667	-4.710	-0.479	1.316	1.883	2.968	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Ag	Br	RS	-0.030	-8.058	-1.667	-4.710	-0.479	1.316	1.883	2.968	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Ag	I	ZB	0.037	-8.058	-1.667	-4.710	-0.479	1.316	1.883	2.968	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Cd	O	RS	-0.087	-9.581	0.839	-5.952	-1.309	1.232	1.736	2.604	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Cd	S	ZB	0.070	-9.581	0.839	-5.952	-1.309	1.232	1.736	2.604	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Cd	Se	ZB	0.083	-9.581	0.839	-5.952	-1.309	1.232	1.736	2.604	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Cd	Te	ZB	0.113	-9.581	0.839	-5.952	-1.309	1.232	1.736	2.604	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
In	N	ZB	0.150	-5.537	-0.256	-2.697	0.368	1.134	1.498	3.108	-13.585	-1.867	-7.239	3.057	0.539	0.511	1.540
In	P	ZB	0.170	-5.537	-0.256	-2.697	0.368	1.134	1.498	3.108	-9.751	-1.920	-5.596	0.183	0.826	0.966	1.771
In	As	ZB	0.122	-5.537	-0.256	-2.697	0.368	1.134	1.498	3.108	-9.262	-1.839	-5.341	0.064	0.847	1.043	2.023
In	Sb	ZB	0.080	-5.537	-0.256	-2.697	0.368	1.134	1.498	3.108	-8.468	-1.847	-4.991	0.105	1.001	1.232	2.065
Sn	Sn	ZB	0.016	-7.043	-1.039	-3.866	0.008	1.057	1.344	2.030	-7.043	-1.039	-3.866	0.008	1.057	1.344	2.030
B	Sb	ZB	0.581	-8.190	-0.107	-3.715	2.248	0.805	0.826	1.946	-8.468	-1.847	-4.991	0.105	1.001	1.232	2.065
Cs	F	RS	-0.112	-4.006	-0.570	-2.220	-0.548	2.464	3.164	1.974	-19.404	-4.273	-11.294	1.251	0.406	0.371	1.428
Cs	Cl	RS	-0.152	-4.006	-0.570	-2.220	-0.548	2.464	3.164	1.974	-13.902	-3.971	-8.700	0.574	0.679	0.756	1.666
Cs	Br	RS	-0.158	-4.006	-0.570	-2.220	-0.548	2.464	3.164	1.974	-12.650	-3.739	-8.001	0.708	0.749	0.882	1.869
Cs	I	RS	-0.165	-4.006	-0.570	-2.220	-0.548	2.464	3.164	1.974	-11.257	-3.513	-7.236	0.213	0.896	1.071	1.722
Ba	O	RS	-0.095	-5.516	0.278	-3.346	-2.129	2.149	2.632	1.351	-16.433	-3.006	-9.197	2.541	0.462	0.427	2.219
Ba	S	RS	-0.326	-5.516	0.278	-3.346	-2.129	2.149	2.632	1.351	-11.795	-2.845	-7.106	0.642	0.742	0.847	2.366
Ba	Se	RS	-0.350	-5.516	0.278	-3.346	-2.129	2.149	2.632	1.351	-10.946	-2.751	-6.654	1.316	0.798	0.952	2.177
Ba	Te	RS	-0.381	-5.516	0.278	-3.346	-2.129	2.149	2.632	1.351	-9.867	-2.666	-6.109	0.099	0.945	1.141	1.827
Ge	C	ZB	0.808	-7.567	-0.949	-4.046	2.175	0.917	1.162	2.373	-10.852	-0.872	-5.416	1.992	0.644	0.630	1.631
Sn	C	ZB	0.450	-7.043	-1.039	-3.866	0.008	1.057	1.344	2.030	-10.852	-0.872	-5.416	1.992	0.644	0.630	1.631
Ge	Si	ZB	0.264	-7.567	-0.949	-4.046	2.175	0.917	1.162	2.373	-7.758	-0.993	-4.163	0.440	0.938	1.134	1.890
Sn	Si	ZB	0.136	-7.043	-1.039	-3.866	0.008	1.057	1.344	2.030	-7.758	-0.993	-4.163	0.440	0.938	1.134	1.890
Sn	Ge	ZB	0.087	-7.043	-1.039	-3.866	0.008	1.057	1.344	2.030	-7.567	-0.949	-4.046	2.175	0.917	1.162	2.373

Table 2 Values related to 82 AB binaries: total energy difference between Rock-Salt and Zinc-Blende ($\Delta E = E^{RS} - E^{ZB}$) (calculated using DFT) and seven atomic properties of corresponding A and B atom[4].

.2 Script for the alphabet identifier

```
Nx:=512
Ny:=128
Nz:=1
cx:=100e-9
cy:=100e-9
cz:=100e-9
SetGridsize(Nx, Ny, Nz)
SetCellsize(cx, cy, cz)
Mid_rect := rect(10000e-9, 10000e-9)
arms := rect(20600e-9, 400e-9)
in_arm1 := arms.transl(-15300e-9,-44cy,0)
in_arm2 := arms.transl(-15300e-9,-33cy,0)
in_arm3 := arms.transl(-15300e-9,-22cy,0)
in_arm4 := arms.transl(-15300e-9,-11cy,0)
in_arm5 := arms.transl(-15300e-9, 0 ,0)
in_arm6 := arms.transl(-15300e-9, 11cy,0)
in_arm7 := arms.transl(-15300e-9, 22cy,0)
in_arm8 := arms.transl(-15300e-9, 33cy,0)
in_arm9 := arms.transl(-15300e-9, 44cy,0)
out_arm1:= arms.transl( 15300e-9,-33cy,0)
out_arm2 := arms.transl( 15300e-9, 0 ,0)
out_arm3 := arms.transl( 15300e-9, 33cy,0)

total_struct := Mid_rect.add(in_arm1).add(in_arm2).add(in_arm3)
.add(in_arm4).add(in_arm5).add(in_arm6).add(in_arm7)
.add(in_arm8).add(in_arm9)

total_struct = total_struct.add(out_arm1).add(out_arm2).add(out_arm3)
demultiflexer := zrange(-50e-9, 50e-9).intersect(total_struct)
setgeom(total_struct)

//defining regions
defRegion(1, xrange( -Nxcx/2, (-Nxcx/2)+1e-6) ) // left 1-
```

```
um damped YIG
defRegion(2, xrange((-Nxcx/2)+1e-6, ( Nxcx/2)-1e-6) ) //
middle YIG
defRegion(3, xrange(( Nxcx/2)-1e-6, Nxcx/2 ) ) // right 1-
um damped YIG
save(regions)

Msat.setRegion(0,0)
Aex.setRegion(0,0)
alpha.setRegion(0,10000)

Msat.setRegion(1, 14e4)
Aex.setRegion(1, 3.5e-12)
alpha.setRegion(1, 0.5)

Msat.setRegion(2, 14e4)
Aex.setRegion(2, 3.5e-12)
alpha.setRegion(2, 2e-4)

Msat.setRegion(3, 14e4)
Aex.setRegion(3, 3.5e-12)
alpha.setRegion(3, 0.5)

T_loc := 1e-9
B_ext = vector(0, 0, 200e-3)
f1 := 2.6e+9
pos1 := Nx/4
B_exc := 1e-2
mask1 := newVectorMask(Nx, Ny, Nz)
for i := pos1; i<pos1+1; i++ {
  for j := 0; j< Ny; j++
  { for k := 0; k<Nz; k++
  { r := index2coord(i,j,k)
  x := r.X()
```

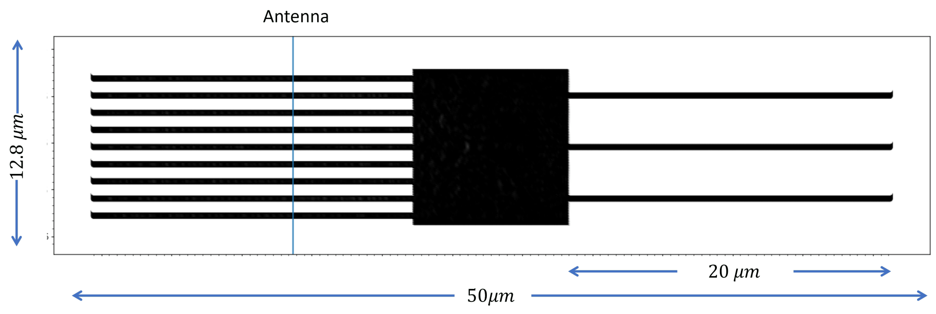


Fig. 2 Representation of 12 alphabets possible to visualize by the 3×3 matrix.

```
  y := r.Y()
  z := r.Z()
  mask1.setVector(i,j,k, vector(1.0, 0, 0)) } } }
B_ext.add(mask1, B_excsin(2pif1(t-T_loc)))

m = uniform(0, 1, 0)
relax()
OutputFormat = OVF2_TEXT save(m)
save(B_demag)
save(B_eff)
autosave(m, 10e-10)
tableautosave(10e-15)
B_ext = vector(0, 0, 200E-3)
run(100e-9)
```

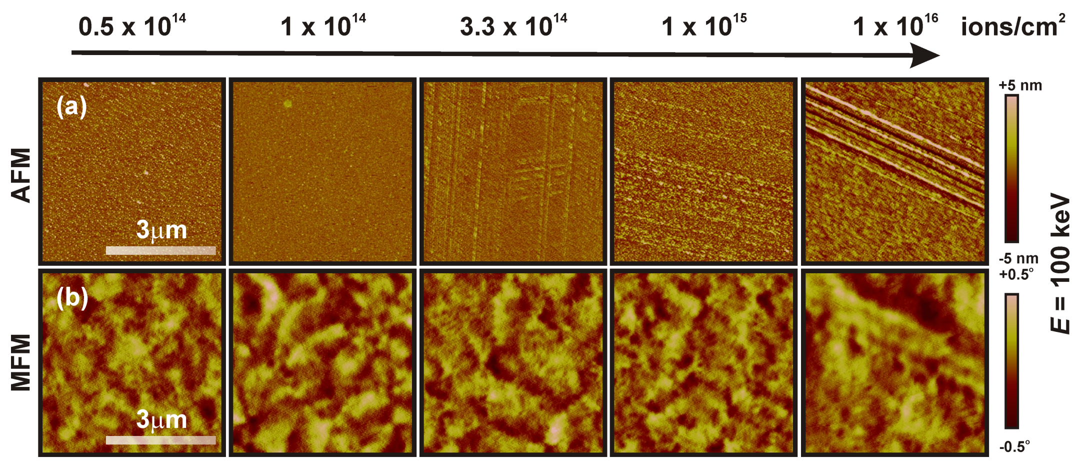


Fig. 3 Series of normalized (a) AFM images with simultaneously captured corresponding (b) MFM images for the films irradiated with 100 *kev* of Ar^+ ions at various fluences (mentioned on top of the image). The scale bar is same for all the images.

.3 Understanding Domain Modification Using Machine Learning

.3.1 Atomic and Magnetic Force Microscopy

Fig. 3 (a) and (b) show the AFM and MFM images, respectively, obtained after exposing the sample to increasing fluences ranging from 0.5×10^{14} to 10^{16}. These images illustrate the changes in magnetic domain evolution resulting from the fluence.

.3.2 Mumax3 script

```
Nx := 256
Ny := 256
Nz := 4
sizeX := 1024e-9
sizeY := 1024e-9
sizeZ := 16e-9
SetGridSize(Nx, Ny, Nz)
SetCellsize(sizeX/Nx, sizeY/Ny, sizeZ/Nz)

//input from the python Msat = 1200000.0
Aex = 2.3e-10
alpha = 0.925
Ku1 = 5000000
anisU = vector(0,0,1)
Temp = 287
//end input from the python

m = RandomMag()
FixDt = 1e-12
run(100e-9)
saveas(m, Domain_CoPd_K_1E6_Ms_1p2)
snapshot(m)

tableadd(Edens_total)
tableadd(Edens_anis)
tableadd(Edens_demag)
tableadd(Edens_exch)
tablesave()
```

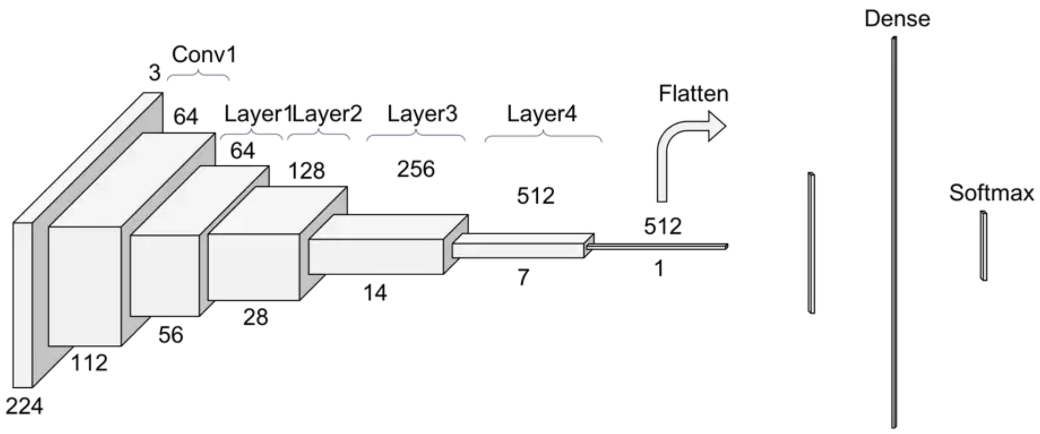


Fig. 4 Lateral dimensions decrement in the ResNet-34 [9]. Picture taken from ref-[10]

.3.3 ResNet Models

The ResNet34 model's conv and pooling layers are depicted in Fig. 4, revealing the structure of each layer. The convolutional process results in a decrease in the lateral dimensions of the image, as evident from the figure.

PhD Activities

❖ **Publications:**

- Gajera, U.; Storchi, L.; Amoroso, D.; Delodovici, F.; Picozzi, S. *"Toward machine learning for microscopic mechanisms: A formula search for crystal structure stability based on atomic properties"*, AIP Journal of Applied Physics, published date: 07th June 2022
- Talapatra, A.; Gajera, U.; Prasad, P S.; Mohanty, J.,*" Understanding magnetic microstructure through neural network and experiments"*, ACS Applied Materials and Interface, Published date: 21st October 2022
- Gajera, U.; Qin, H.; Picozzi, S.; Dijken, S.; "*Design and fabrication of magnonic logistic device using Genetic Algorithm"*, In preparation.

❖ **Period Abroad:**

- Aalto University: *Simulation and prediction of spin waves in demultiplexer,* 16th September 2021 - 31st October 2021 (6 weeks).
- Aalto University: *Simulation and prediction of spin waves in demultiplexer,* 12th June 2022 - 25th June 2022 (2 weeks).
- Voxalytic GmbH: *Simulation of magneto ionic device using COMSOL multi-physics software*, 25th June 2022 - 6th August 2022 (6 weeks)

❖ **Presentations:**

- U. Gajera, A. Talapatra, "Fundamentals, Applications and Advancement in Machine Learning" 13th December 2022, Innsbrück
- U. Gajera, S. Picozzi, "*Toward machine learning for microscopic mechanisms*" BeMagic workshop, 14th June 2022, Aalto University, Finland.
- U. Gajera, S. Picozzi, "*Applications of Machine Learning in Material Science"* Second year university review report, 24th February 2022, Unito-Turin, Italy (online presentation).
- U. Gajera, H. Qin, H. Tan, S. Picozzi, S. Dijken, "*Magnonic Neural Network*"- Magnetoelectricity in biomedicine: healthcare for the 21st century, 12th November 2021, ETH-Zurich, Switzerland.

❖ **Poster presentation at Conference:**

- U. Gajera, A. Talapatra*, "Understanding magnetic microstructure through neural network"* IEEE, Around-the-Clock Around-the-Globe Magnetics Conference, 31st August 2022 (Online Event).

❖ **PhD courses**

Course title	Professor	University/ Department	Hours	CFU
From biomass to chemicals and biopolymers towards a sustainable future	Silvia Tabasso	Chimica Industriale	12	3
Introduction to scientific programming	Alessandro Erba	Chimica Fisica	20	5
Chemical Sensors for Scientific Research and Everyday Life	Ornella Abollino	Chimica Analitica	8	2
Electrochemical Energy Storage and Conversion Systems	Mauro Sgroi	Materials Sciences	12	3
The Magic of DFT	Bartolomeo Civalleri	Material Science	12	3

❖ **Outreach activities:**

- Lecture at workshop by, U. Gajera, "*Fundamentals, applications and advancement of Machine Learning*"- Advance Magnetic materials and Applications, 29th July 2022, IIT Hyderabad, India (Online lecture).

❖ **Awards:**

- Best video presentation award to U.Gajera, M. De H-óra, Z. Zhao "*Possible application of the nano-Magneto electric composites for cell therapies*" organized by BeMagic training management on 20th December 2021.

❖ **Attended Congresses and Workshops**

- "Innovation, entrepreneurship and marketable products using ME technology" organized by G.TECH, 13-16 December 2022, CFU=4
- "Fundamentals of magnetoelectricity and how it can boost energy efficiency" organized by CNR, 9-10 September 2020, CFU=3.5
- "Training on synthetic and Magnetoelectric characterization methodologies" organized by the Leibniz Institute for Solid State and Materials Research Dresden (IFW), 7-11 December 2020, CFU=4

❖ **Summer and winter schools**

- Summer school "*From fundamental properties of matter to magnetic materials and applications*" organized online by EMA (European magnetism Association) on 6th September 2021–17th September 2021.
- MaX School on "*Advanced Materials and Molecular Modeling with Quantum ESPRESSO",* on 17th May 2021-28th May 2021

❖ **Online training courses:**

- Raman Days-2021, Prof. Anna Maria Ferrero, PhD doctorate in Earth Science, 28-29 January 2021, Torino, CFU=2
- "From Stimuli Responsive and Shape Memory Materials to Algorithmic Materials that Mimic Behaviorists Classical Conditioning ", prof. Olli Ikkala, Virtual DCMIC Seminar, 22 May 2020. CFU= 1
- "Machine Learning Meets Chemistry" organized by Department of Chemistry, University of Torino, 17-18 February 2020, CFU=2