Classification of Magnetic Configurations using Machine Learning Algorithms

by

Saffat Bokul
16301001
Samiha Sabrin Md Abdus Shukur
16201037
Saquib Ahmed
17301181

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
August 2019

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Saffat Bokul
Student ID: 16301001

Samiha Sabrin Md Abdus Shukur
Student ID: 16201037

Saquib Ahmed
Student ID: 17301181

Approval

The thesis/project titled "Classification of Magnetic Configurations using Machine Learning Algorithms" submitted by

1. Saffat Bokul (16301001)
2. Samiha Sabrin Md Abdus Shukur (16201037)
3. Saquib Ahmed (17301181)

Of Summer, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on August 7, 2019.

Examining Committee:

Supervisor:
(Member)

Dr. Md. Ashraful Alam
Assistant Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Jia Uddin
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Mahbubul Alam Majumdar
Professor and Chairperson
Department of Computer Science and Engineering
BRAC University

Abstract

Machine learning is used to carry out efficient studies and analyses in the field of condensed matter physics. We propose comprehensive machine learning approaches that would classify between magnetic structures. We propose models that are trained on data that has been generated on 3D lattices of Heisenberg model using the physical properties of respective magnetic structures. Models are designed based on three types of classifications, first classification is done between topologically-protected structures, second on non-topologically-protected structures, thirdly on all structures collectively. To achieve this, convolutional neural network (CNN) and support vector machine (SVM) with principle component analysis (PCA) algorithms have been used. We then make a comparative analysis and find the most optimal solution. The results show that CNN provides the highest accuracy in the classification of topological and non-topological magnetic configurations.

Keywords: Machine Learning; Convolutional Neural Network; Support Vector Machine; Principle Component Analysis; Skyrmion; Ferromagnetic; Spin-spiral; Anti-skyrmion; Anti-ferromagnetic; Topological

Dedication (Optional)

By the grace of the Almighty, we have finally completed our thesis. We would be delighted to dedicate this to our loving parents and supportive faculties.

Acknowledgement

First of all, all praise to the Most Merciful for providing us with the capabilities of completing our thesis without any major issue in our ways. This thesis would not have been possible without our supervisor Dr. Md. Ashraful Alam who immensely helped us with all his resources and guidance. We would also like to express our gratitude to our co-supervisor, Tonmoy Kumar Bhowmick of University of California, Riverside who has been a great mentor through out our thesis period. Also, the department deserves a praise without whom this thesis would not have initiated in the first place. In addition, we would like to show appreciation to Dr. Md. Ashraful Alam who initiated the research and CVIS labs facilities which have greatly helped us complete our thesis. Lastly, we are grateful to our families for constantly supporting us. With the help of their love and support, we are finally on the brink of graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

$AnFe$ Antiferromagnetic

$AnSk$ Antiskyrmion

CNN Convolutional Neural Network

DM Dzyaloshinskii-Moriya

Fe Ferromagnetic

FNN Feed Forward Network

$k-NN$ k-nearest neighbor

$LTEM$ Lorentz Transmission Electron Microscopy

ML Machine Learning

NN Neural Network

PCA Principle Component Analysis

RBF Radial Basis Function

$ReLU$ Rectified Linear Unit

SEM Scanning Electron Microscopy

$SEMPA$ Scanning Electron microscopy with Polarised Analysis

Sk Skyrmion

Sp Spin Spiral

STM Scanning Tunneling Microscopy

SVM Support Vector Machine

TEM Transmission Electron Microscopy

Chapter 1

Introduction

1.1 Motivation

Nowadays, computer algorithms can gain an understanding of data and forecast the outcome using machine learning. Most of the times, such algorithms are applied on areas which require classification between data to be performed on large scale data sets and the evaluation of results is done in the absence of human guidance [1]. In the fields of physics, chemistry and material science, working out the solution of molecular structures, materials and their interfaces play a fundamental role [2]. Therefore classifying various phases and transitional stages of matter is always a critical topic in the area of condensed matter in physics [3]. The task of identifying between several phases and configurations is extremely difficult using conventional methods. Consequently, this field requires machine learning to aid in the classification of many-body stages of classical and quantum models. These algorithms can efficiently identify the magnetic configurations without the need for any tiresome microscopy techniques.

1.2 Literature Review

Iakovlev, Sotnikov and Mazurenko (2018) used machine learning techniques to differentiate between skyrmion, paramagetic, ferromagnetic and spin spiral phases along with their transitional states in two-dimensional materials at different temperature and in the presence of various magnetic fields. They used two approaches here and made a quantitative analysis. The first approach consists of neural network with a single-hidden layer. However, they faced difficulties in establishing concise boundaries between skyrmion and spin spiral phases. They used a feed-forward network (FFN) on magnetic skyrmion and spin spiral states to achieve a non-collinear configuration which comes from the Monte Carlo simulation. The neural network takes this magnetic configuration as input and distinguishes between the magnetic configuration efficiently. They use a minimum distance method for their second approach. For the minimum distance method, they use k-NN and nearest centroid classifier. They achieved higher performance when they use nearest centroid method as opposed to the neural network and k-NN approaches. Their training data

set consisted of big skyrmions, high temperature spin spirals, ferromagnetic, spin spiral, skyrmion and paramagnetic phases [4]. Successful quantitative classification resulted between complex and mixed ferromagnetic-skyrmion and skyrmion-spiral configurations which had not been done before. Speaking of the performance of the algorithms used, it was discovered that both neural network and nearest centroid classifier demonstrate similar detection accuracy for big skyrmions at 94% and 100%. Also accuracy for high temperature spirals was at 75% and 78%, whereas both provided accuracy of 100% for ferromagnetic, and 91% and 100% accuracy for skyrmion configurations on triangular lattice. In addition to that, they found out nearest centroid algorithm provided excellent performance for paramagnetic phase at 90%. However, k-NN provided 100% accuracy for ferrromagnetic and paramagnetic phases but worked poorly on the rest of the data [4].

Singh and Han (2018) used machine learning approaches to support phases of skyrmion in two-dimensions. They used many layers of convolutional neural network associated with several layers of neural network. Their work basically consisted of feeding in a lattice model made of Heisenberg, DM interaction and Zeeman term. Their model could successfully predict both the mechanical quantities of the input configuration such as chirality, magnetization of the structure as well as physical quantities such as temperature and magnetic field at which images were generated. In addition to that, they train their network using three different configurations consisting of z, xy and xyz components of magnetization and yet got no differences among the accuracy of the three types of training and testing data sets [5].

1.3 Objective

We apply machine learning approaches to perform an effective classification of ferromagnetic, anti-ferromagnetic, skyrmion, antiskyrmion and spin spiral configurations via supervised learning and unsupervised learning. The input sample is data achieved from simulations of physical properties of the various magnetic configurations. First CNN is used to classify between the images. Image classifications are mostly carried out using neural networks where data is placed in a graphical structure which contains a number of nodes and edges [1], [3], [5], [6]. Also SVM algorithm is used, once in association with PCA and once without PCA. The proposed approaches can successfully classify amongst magnetic configurations in real time with data obtained from spin-polarized scanning tunneling and Lorentz transmission electron microscopy.

1.4 Thesis Outline

The rest of the paper is structured as follows:

- Chapter 2: Discussion on Fundamentals of Algorithms used
- Chapter 3: Discusses Fundamental of Magnetism
- Chapter 4: Discusses the Proposed Methodology

- Chapter 5: Discussion on Experiments and Results
- Chapter 6: Conclusion and Future Works
- Chapter 7: References

Chapter 2

Algorithms Used

2.1 Supervised Learning Algorithm

The theory of learning in machine learning algorithms usually follow two methods, supervised or unsupervised. When it comes to supervised learning, the model is given a set of labelled data so that model is able to generalize according to those labels on new data [7]. In this type of learning, data set needs to be provided. Also the sample has to be a good representation of real-world data to ensure good accuracy. Some of the most used supervised learning algorithms are neural networks, CNN, SVM, linear regression, logistic regression, naive Bayes, decision trees.

2.1.1 Neural Network

Neural network is a concept of machine learning created upon the principle of biological neural networks. The concepts were discovered in an attempt while simulating the brain processes by Warren McCulloch and Walter Pitts [8]. A classic neural network consists of a number of artificial neurons knows as units which are organized in layers. There are connections between the neurons from one layer to the layers on either sides. Figure 2.1 elaborates the breakdown of the steps taking place in the neural network. The first layer known as input layer takes in information that is needed to recognize or classify. On the other end of the network, there are output nodes in the output layer which gives a decision based on the information. There is one or several layers of hidden neurons between the output and the input layers that produces majority of the artificial brain. All individual layers are connected to each other via links knows as weights. These are particular numbers that decide if one unit is influenced by the previous unit. Information goes through the network in two methods, either during training period or while testing [9].

During training, the network makes use of a feedforward network. Aim of feedforward network is to estimate a function f^*. For instance, $y = f^*(x)$ is a classifier which maps input x to a category y. The feedforward network describes a mapping $\mathbf{y} = \mathrm{f}(\mathrm{x}; \boldsymbol{\theta})$ and studies the values of parameters θ. The parameter is to be studied to provide the best function estimation [10]. After obtaining results from network, the difference between predicted labels and original results are analyzed. We define

a performance metric based on the ability of the NN to achieve its goal of providing predictions as close as possible to the actual results using a loss function. One of the most famous loss functions is the sum of squares of the absolute errors [11].

To optimize the weights of NN, a concept known as differentiation is applied. It basically works with derivative of loss function. Derivative is a measure of the rate at which errors change with a change in weights. Then the errors are back-propagated from the end to the beginning. Back-propagation is used to efficiently calculate weight updates to enhance the performance of the network recursively until it reaches its desired output [10], [12]. Several methods of updating weight exists. These methods are widely known as optimizers. After applying an optimizer, the procedure keeps iterating until it reaches convergence. Number of iterations required to reach convergence depends on the learning rate applied, meta-parameters of network, the optimisation method used, random initialisation of network and quality of training set [11].

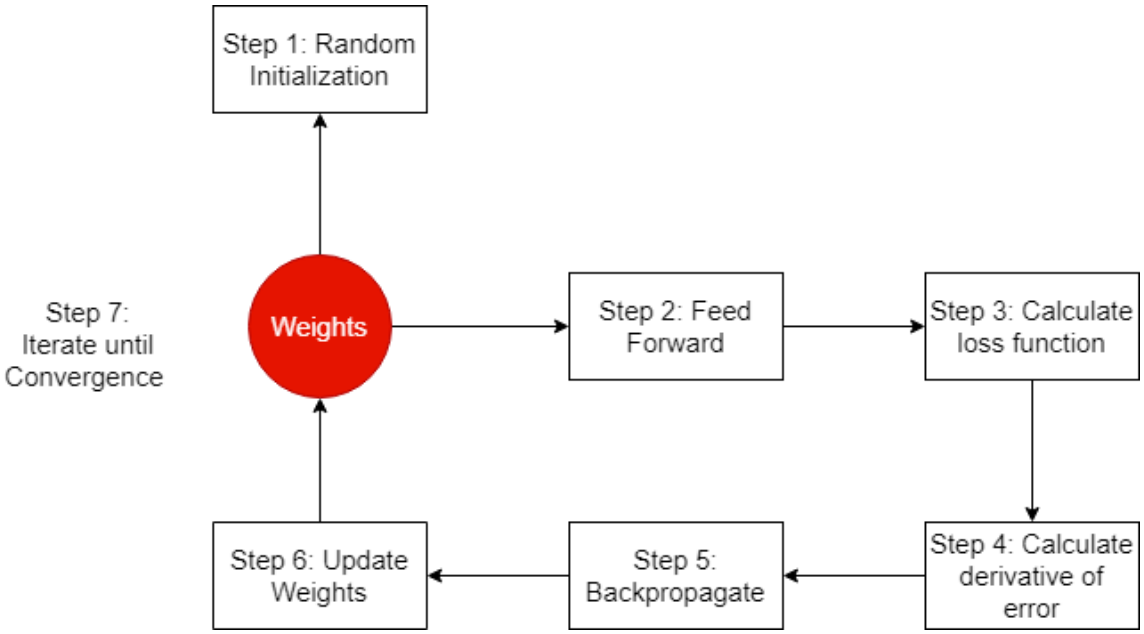

Figure 2.1: Step-by-step breakdown of a neural network.

2.1.2 How Convolutional Neural Networks Work

Convolutional neural network is an architecture of artificial neural networks, which uses some features resembling the visual cortex. CNN has at least one convolution layer and it is designed in a way to reduce computational complexity when it comes to tasks like image recognition. The image passes through a series of convolutional, nonlinear, pooling and fully connected layers to generate an output as is demonstrated in figure 2.2.

Before the CNN is provided with an image, the computer converts this image into an array of pixels known as matrix, depending on dimension of image and image type. If the image is a gray-scale image, there is only one pixel. However, for an RGB image, there are three color planes, thus 3 pixels. Similarly there are wide ranges of color

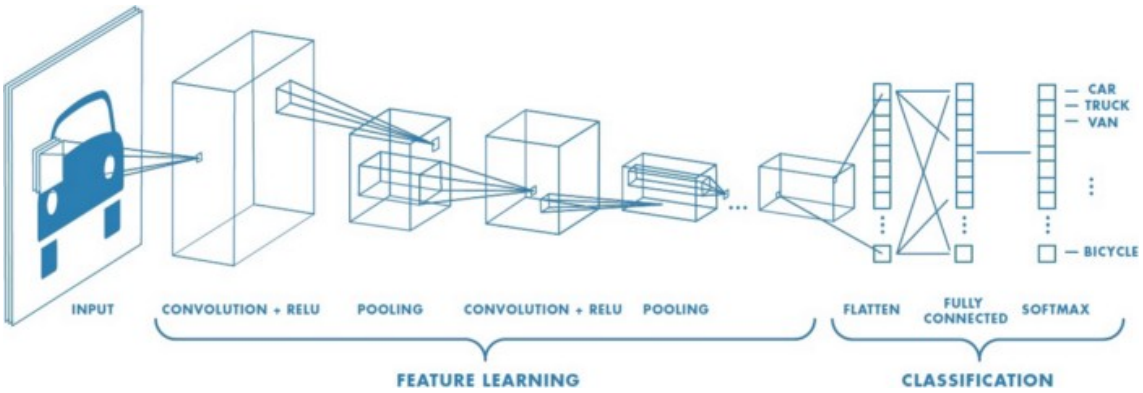

Figure 2.2: Neural network containing multiple convolutional layers.

spaces for HSV, CMYK etc. Therefore, a convolutional layer is needed to decrease the matrix size and generate a general form which would be easier to process and significant features would not be lost. This also comes handy for designing network which is not conformed to only one type of images but can be scaled to larger data sets.

Convolutional Layer

The matrix first goes through the convolutional layer. A smaller matrix or filter, known as feature detector is used which moves across the input matrix starting from the top left. The filter's job is to multiply its value with the image pixel values. These multiplications are then summed up to generate a single number at the end. The filter continues doing this by moving further right by 1 unit every time, if the stride is 1. After the filter passes through all positions, a smaller matrix is achieved known as a feature map.

The input matrix will be in a dimension of $W_1 \times H_1 \times D_1$ where W_1 is the width, H_1 is the height and D_1 is the depth of the matrix. After convolution the output produced would be $W_2 \times H_2 \times D_2$ where:

$$W_2 = (W_1 - F + 2P)/S + 1 \tag{2.1}$$

$$H_2 = (H_1 - F + 2P)/S + 1 \tag{2.2}$$

$$D_2 = K \tag{2.3}$$

where K is the number of filters, F is the size of filter, S is the stride size and P is the amount of zero padding [13].

Using this procedure we lose a large amount of features but it does not leave the important features, rather keeps features which are to be used in future layers. The feature maps we get from this procedures consist of important features that is needed to recognize and classify various types of images with the convolutional neural network (CNN).

Non-Linearity

This layer is added after every convolutional operation. The objective of this layer is to add an activation function that would bring non-linearity in the images. There are several activations functions such as sigmoid, tanh, ReLU,Leaky ReLU, Parametric ReLU and softmax. Without non-linearity, the network can not model the class label efficiently.

Figure 2.3 shows the sigmoid function. It resembles an S-shaped curve [14]. It provides a range from 0 to 1. The function can be differentiated, thus slope can be found at any two points.

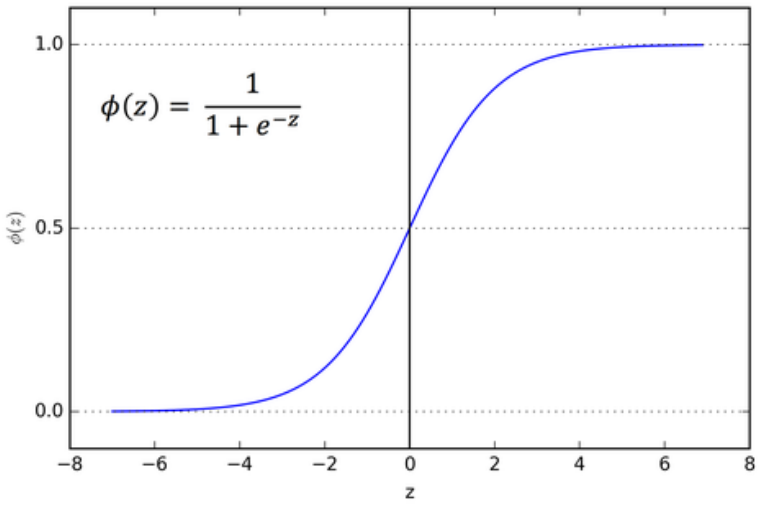

Figure 2.3: Sigmoid function.

Figure 2.4 displays the rectified linear unit function (ReLU) [14]. Conventionally ReLU activation function is used while dealing with classification problem using deep neural networks due to its high computational speed [15]. The function and derivatives are both monotonic. Also all the neurons are not activated at the same time, thus it is efficient.

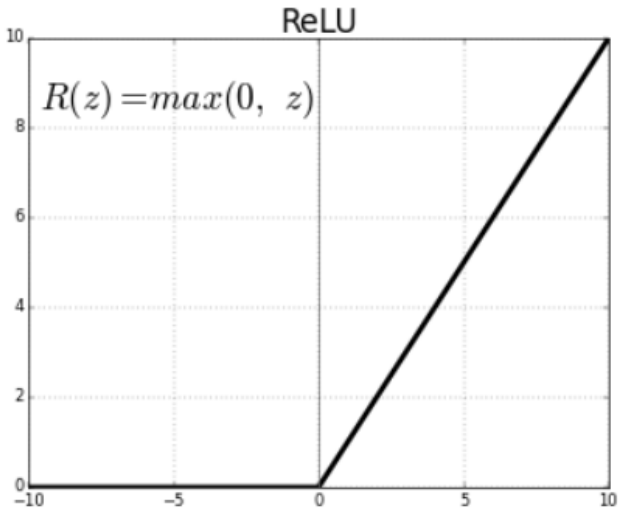

Figure 2.4: ReLU function.

Equation 2.4 displays the softmax function [16] and figure 2.5 shows the graphical representation of the function. Softmax can manage multiple classes and find probability of input belonging to a specific category.

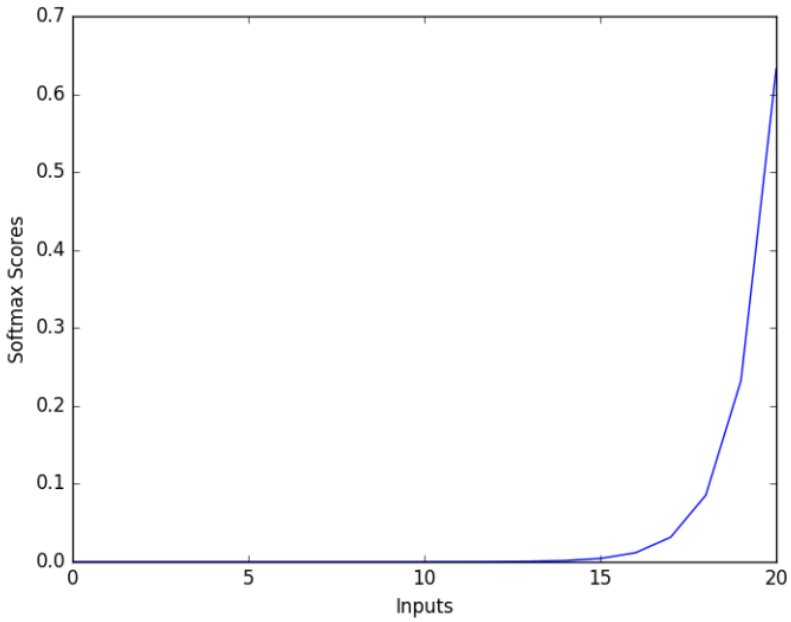

Figure 2.5: Softmax function.

$$\sigma(z)j = \frac{ezj}{\sum k_{k=1ezk}} \quad \text{for } j = 1 \ldots k \tag{2.4}$$

Pooling Layer

The next layer is a pooling layer. Pooling is done to get most significant features and to get rid of most of the unwanted features. This will result in getting a smaller sized feature map which is easier to process. This is equivalent to picking activation in a random manner based on a distribution which is multinomial at the time of training [17]. There are several types of pooling, Max Pool, Sum Pool and Average Pool. Thus, the layer takes input of feature maps generated from convolution layer and generates pooled feature map as output.

Fully Connected Layer

After going through a series of convolutional, nonlinear and pooling layers, a fully connected layer is required. Figure 2.6 shows the fully connected layer [18]. The output from the pooled layer is flattened into a long vector which is input to the artificial neural network. The fully connected layer is attached to the end of the network which has N number of nodes, where N is the number of classes to be classified. After the full connection step, learning rate is specified. Over a series of epochs, model can distinguish between the images and classify.

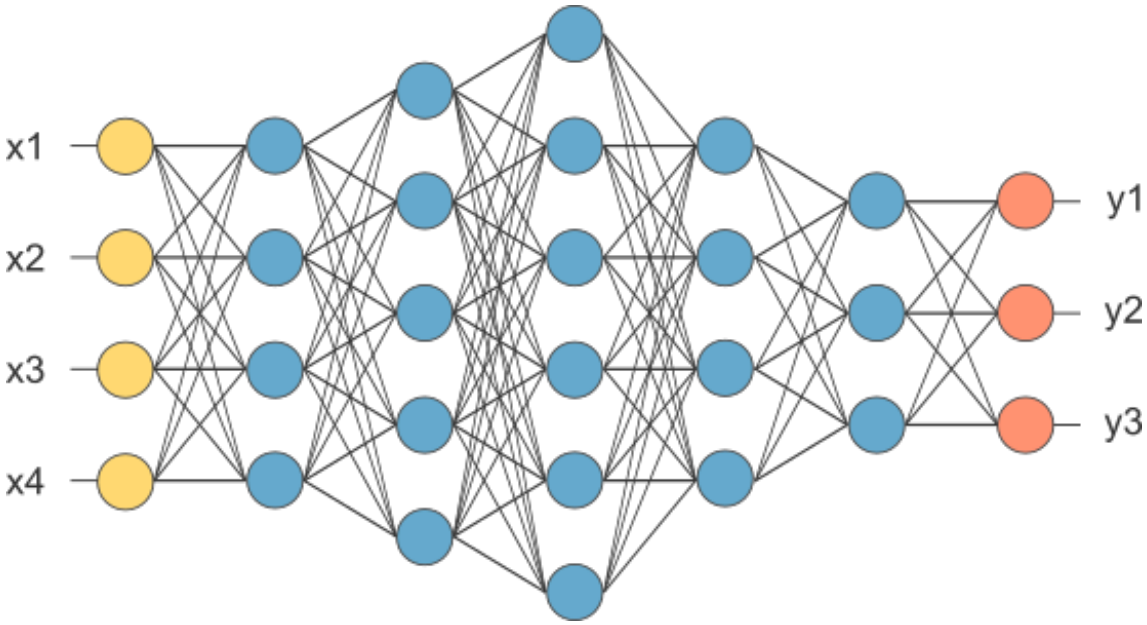

Figure 2.6: Fully connected layer.

2.1.3 Support Vector Machine

Objective of support vector machine (SVM) algorithm is to find a decision boundary in an N-dimensional space (where N represent the quantity of features) to classify between points [19].

To draw a decision boundary between the two classes, a hyperplane is drawn. The classes are separated based on the hyperplane. Dimension of hyperplane is chosen

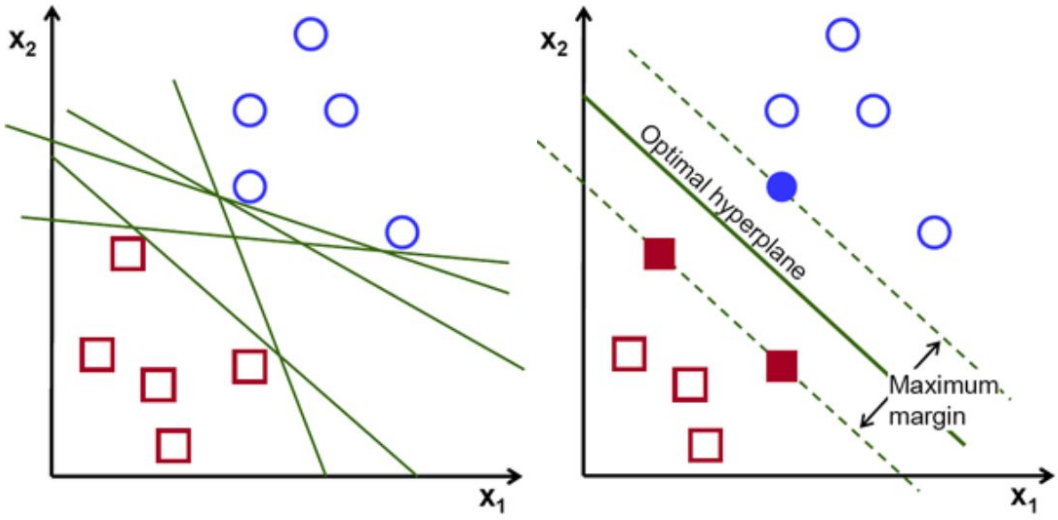

Figure 2.7: Figure showing possible hyperplanes vs an optimal hyperplane.

depending on the number of features. For instance, hyperplane is a simple line if there are 2 input features. Hyperplane becomes a two-dimensional plane if quantity of input feature is 3. As number of features increase, complexity of hyperplane increases. However, there are many possible options for the hyperplane as shown in Figure 2.7, but SVM aims to find an optimal hyperplane where the margin distance from the hyperplane to data points on both sides are maximum. The optimal hyperplane is drawn in the second grid on figure 2.7.

Support vectors are the data points which fall nearer to hyperplane and they have an impact on the positioning of the hyperplane. In the second grid on figure 2.7, the filled circle and squares are the support vectors.

Aim of SVM is to maximize distance or margin between the hyperplane and data points. There is a loss function that is used to maximize the margin, known as hinge loss. A regularization parameter is added to the cost function to create a balance between loss and margin maximization. Partial derivatives are taken from the loss function with respect to weight to calculate the gradients. The weights are then updated using the gradients [19].

Multiclass SVM

SVMs are by default binary classifier. However, multiclass classification can be performed using 2 approaches. They are one-against-one and one-against-all methods [20]. One-against-all approach considers one class as positive and rest all as negative and trains the classifier. Thus, if there are n classes, it has to train n number of classifiers. In the classification stage, the n-classifiers predict probability of each class with respect to new data and class with highest probability is chosen [21].

For the one-against-one approach, it takes every 2 classes and trains classifiers on data containing those classes. This way it trains a sum of $n^*(n-1)/2$ classes. During the classification stage, each classifier makes prediction of one class and class that has been suggested most for a particular data is the solution [21].

Kernel Functions

SVM algorithms make use of some mathematical functions known as kernel. Kernels aim to take input data and change it into a desired form. Different algorithms use different kernel functions. Some of the functions are gaussian radial basis bunction (RBF), polynomial and sigmoid.

RBF is a kernel used when no prior knowledge is given. Equation 2.5 shows the rbf equation.

$$k\left(\mathbf{x_i}, \mathbf{x_j}\right) = \exp\left(-\gamma \left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2\right) \tag{2.5}$$

2.2 Unsupervised Learning Algorithm

Unsupervised learning is a type of machine learning that does not required data to be labelled, categorized or classified. It does not need feedback to learn. Unsupervised learning learns by recognizing common elements among the data and decides based on those common elements in a new piece of data [22]. The goal of unsupervised learning is to model the underlying structure or distribution of data to get a good knowledge of the data. Some of the widely used unsupervised learning algorithms are k-means clustering and PCA algorithms.

2.2.1 Principle Component Analysis

Principle Component Analysis (PCA) is a mechanism that is used to decrease the dimensionality when it comes to very big data sets. It is done by changing a variable set into a smaller one including maximum of the information from the big data set [23]. A step wise breakdown of PCA is demonstrated below.

Standardization

First of all, initial variables need to be standardized to prevent biased results occurring from big differences in the range of variables. This is usually done using equation 2.6 [23].

$$z = \frac{\text{value} - \text{mean}}{\text{standard deviation}} \tag{2.6}$$

In the above equation, mean is calculated from the training sample. To standardize, the mean is subtracted from each value and the result is divided by standard deviation.

Covariance Matrix Computation

This step is performed to see how variables vary from mean with respect to one another, or to find correlation between the variables. This is done to find any un-

derlying property or redundant information among the points. Thus, the covariance matrix is calculated.

Covariance matrix is a $m \times m$ matrix where m represents the number of dimensions. The matrix has covariances with all possible pairs of the input variables. Equation 2.7 shows the covariance matrix for a 3 dimensional data with x, y and z.

$$\begin{bmatrix} \mathrm{Cov}(x,x) & \mathrm{Cov}(x,y) & \mathrm{Cov}(x,z) \\ \mathrm{Cov}(y,x) & \mathrm{Cov}(y,y) & \mathrm{Cov}(y,z) \\ \mathrm{Cov}(z,x) & \mathrm{Cov}(z,y) & \mathrm{Cov}(z,z) \end{bmatrix} \tag{2.7}$$

The formula of covariance is as follows:

$$\mathrm{Cov}(x,y) = \frac{\Sigma\,(x_i - \overline{x})\,(y_i - \overline{y})}{N} \tag{2.8}$$

where x_i and y_i are the coordinates of the points and $\overline{x}$ and $\overline{y}$ are the average points of x and y points respectively. N represent number of total points. After achieving the covariance matrix, the signs of the covariances are recorded. If it is positive, the two variables are correlated, else they are inversely related.

Computation of Eigenvectors and Eigenvalues

Principal components are variables newly formed as linear combinations or mixtures from the starting variables. Eigenvectors and eigenvalues are calculated from covariance matrix to figure out the principal components.

The eigenvectors and eigenvalues are always in pairs and the eigenvectors computed are basically the axes direction containing highest variance, otherwise known as the principle components [23].

Feature vector

This step enables us to either keep all components or discard ones with lower significance of eigenvalues. Thus, the feature vector is a matrix that contains as many columns as the number of eigenvectors we desire to keep. This step is the first step to reducing dimensionality.

Recast Data Along Principle Components Axes

This is the final step where the feature vector from previous step is used to reorient data from initial axes to ones shown by principle components. This is done by taking products between transpose of original data to transpose of feature vector as shown in equation 2.9 [23].

$$\text{Final Data Set} = \text{Feature Vector}^T * \text{Standardized Original Data Set}^T \tag{2.9}$$

2.3 Tools, Softwares and Libraries

2.3.1 TensorFlow and Keras

TensorFlow is an open-source platform that is easily accessible and free for users. It basically is a math library and is widely used in machine learning approaches especially neural networks. It is composed of an inclusive package of tools, libraries and community resources that makes deployment of ML applications easier.

Keras is a high level API used for neural networks. It has the ability to run on top of Theano or TensorFlow. Keras allows for quick prototyping and supports CNN and RNN including their combinations.

2.3.2 Scikit-learn

Scikit-learn is a library in Python that enables using several unsupervised and supervised learning algorithms. It is built on top of numpy, pandas and matplotlib. Scikit-learn provides a number of functionality including regression, classification, clustering, model selection and preprocessing.

2.3.3 Jupyter Notebook

Jupyter Notebook is a web based application which is open-source and enables creation of documents which consist of live codes that can be shared as well. It is most primarily used in simulation, statistical modelling, machine learning and much more.

2.3.4 Mayavi

Mayavi is a data visualizer used to produce computational grids, vector and scalar data. This library was used during data set generation.

2.3.5 Numpy

Numpy is a Python package used for scientific computations. It also consists of N-dimensional array objects to work with. It also comes in handy when using linear algebra, Fourier transformations and various random value abilities. In addition to that, Numpy also acts as a multi-dimensional container for all sorts of data and objects. Arbitrary types of data can also be described making it easier for Numpy to integrate with databases.

2.3.6 Matplotlib

Matplotlib is a Python library used for 2D plotting. It is used in the creation of quality figures and graphs. Plots, histograms, power spectra, scatter plots, bar charts can be created using Matplotlib.

Chapter 3

Fundamentals of Magnetic Structures

3.1 Magnetism

Magnetism has opened doors to a number of new inventions. It plays a major role in the advancement of technological devices including data storage etc. In addition to that, the world is currently producing huge amount of data everyday. We need to catch up on the rapidly increasing amount of information as well as manage this data by storing and processing it in decreasing size of storage devices [24]. Thus, hard drives play a very important role in the field of Computer Science. To get a deeper understanding of what happens inside hard disks, we also need to understand Magnetism. So, hard drives use magnetism for the storage of information in a layer of magnetic material that is situated under the surface of the platter of a typical hard disk. Small magnetic domains are created in the magnetic layer that enables storage of information in this material. Regions of the material that represent same direction as the magnetic moment are the magnetic domains. Magnetism is basically derived from the electron spin and so is the creation of magnetic moments in ferromagnetic materials.

3.2 Classical Spin Models

Spin models are basically mathematical models which are essentially applicable in physics to demonstrate magnetism. Spin models could be of 2 types; Classical and Quantum spin models [25]. Classical spin models connect the gap between an electronic description of magnetic material and a conventional micromagnetism. In micromagnetism, magnetic properties are evaluated based on a continuum theory. As spin models take account of magnetic configurations on atomic levels, it becomes possible to study ferromagnets, antiferromagnets and even heterostructures that contain both of them. In the previous days, magnetism was studied using thermal equilibrium properties but nowadays thermally activated spin dynamics are more advanced in this field of research [25].Some of the most prevalent classical spin

models are Ising model, XY model and Heisenberg model. Ising model contains of a lattice of spin variables that only have values of +1 or -1. Therefore, the spins are discreet. An XY model consists of planar vector spins, usually just in 2 dimensions. A Heisenberg model has spins of 3-dimensional vector. Figure 3.1 demonstrates the 3 models.

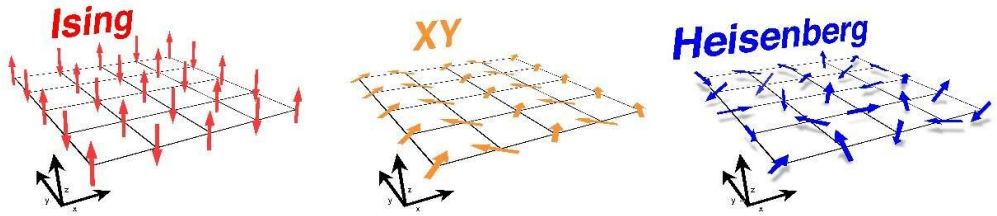

Figure 3.1: 3 Classical Spin Models.

3.3 Magnetic Configurations

3.3.1 Topological and Non-topological structures

Topological structures, or topologically-protected structures are the structures associated with topological invariants, i.e, they do not change when subjected to outer physical conditions such as external magnetic fields or high pressure or temperature. Skyrmions and antiskyrmions are such structures. On the other hand, non-topological or non-topologically-protected structures are ones which can get influenced by external physical changes. Such structures are ferromagnetic, antiferromagnetic and spin spiral configurations.

3.3.2 Ferromagnetic Properties

A ferromagnet is a substance which demonstrates instant magnetization even when there is no magnetic field applied to it [26]. Although, all materials are magnetic to some degree, these materials have very strong magnetic properties. Ferromagnetic materials not exceeding their Curie temperature, represent magnetic domains where their atomic dipoles are uniformly aligned, thus creating non-zero magnetic field. The Ising model of ferromagnetism claims it is created due to an interaction amongst spins of specific electrons present in the atoms of the consisting crystal [27]. Ferromagnets have resulted into the discovery of compasses, electromagnets and generators that are a quintessential part of today's world [28].

3.3.3 Anti-Ferromagnetic Properties

In antiferromagnetism, adjacent particles arrange themselves in opposite directions to counteract the overall magnetism. This alignment takes place instantaneously when the temperature is less than a critical temperature, called Neel temperature [26].

3.3.4 Skyrmions

There has been a lot of hype around skyrmions in the recent years and many researches are taking place over this subject [29]. Ferromagnetic materials have been the prime material used in the development of storage devices, however further advancements can not be made due to size constrains. For example, when we develop magnetic hard disks, nanoscale ferromagnetic particles store all the data but their magnetic states happen to lose stability when they are shrunk below a particular size. This causes data loss. To combat this issue, magnetic skyrmions can be utilized [24]. They are basically considered as knots in a magnetization distribution with major topological properties [30]. Looking from a topological viewpoint, an integer invariant is used to describe a magnetic skyrmion, known as the topological charge. This charge is created in the map starting from the physical two-dimensional space to the target space S_2. The formula that calculates the topological charge is as follows:

$$Q = \frac{1}{4\pi} \int d^2r \, (\partial_x m \times \partial_y m) \cdot m \tag{3.1}$$

where m is a unit vector which points in the direction of magnetization. Here, the topological charge signifies the number of turns around a unit sphere by the magnetic moments in the mapping. Skyrmions are said to uphold topological protection because erasing a skyrmion would require global amendment of the system [31].

Due to their topological properties, they uphold high stability when used in the development of skyrmion-based storage devices due to their topological protection [24], [30]. The spins in skyrmions are organized in such a way that the spins at the edges are in the opposite direction to spins in the centre [32].

3.3.5 Antiskyrmion

Antiskyrmions resemble skyrmions, they are tiny structures with topological protection. However, topological charge of antiskyrmion is opposite to that of a skyrmion's charge. Also, antiskyrmions provide greater advantages since their stability is higher than that of skyrmion due to dipolar interactions [31] .

3.3.6 Spin Spiral Phases

Spin spirals are another type of magnetic configuration in which magnetic moment of an atom is rotated by a fixed angle starting from one unit cell to the following cells

along a specific direction. When the spin-orbiting coupling is missing, the rotation axis can be described as a universal z-axis. Then we get a magnetization direction as:

$$\hat{e}^{n,\mu} = \begin{pmatrix} \cos\left(\mathbf{q}\cdot(\mathbf{R}^n + \boldsymbol{\tau}^\mu) + \alpha^\mu\right)\sin\theta^\mu \\ \sin\left(\mathbf{q}\cdot(\mathbf{R}^n + \boldsymbol{\tau}^\mu) + \alpha^\mu\right)\sin\theta^\mu \\ \cos\theta^\mu \end{pmatrix} \tag{3.2}$$

where R^n is the lattice vector which basically directs from origin to unit cell n, τ^μ is the basis vector of atom μ in unit cell, q is the spin spiral vector, θ^μ is the angle between magnetic moment of atom, μ and rotation axis. α^μ is a phase shift that is dependent on atoms [33]. Figure 3.2 shows the spins at various angles.

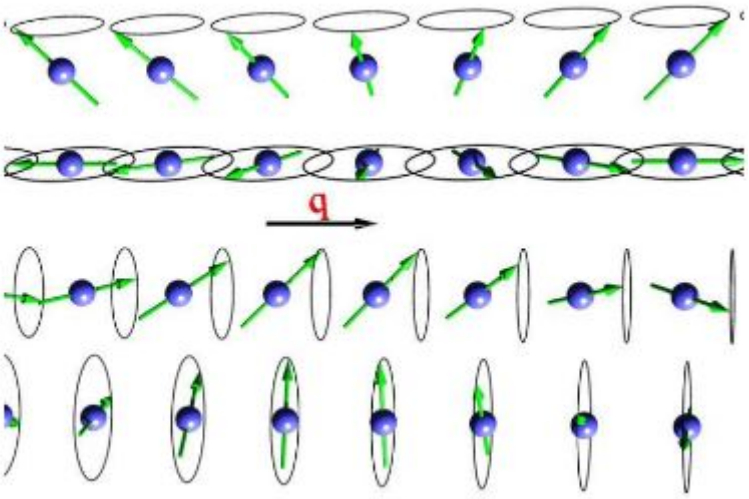

Figure 3.2: Spin spirals.

3.4 Microscopy Techniques

Scientists use 3 modes of microscopy to see objects at the molecular levels; Electron microscope, optical microscope and scanning probe microscope. Electron microscope is a technique that aids in exposing the internal structure of smaller materials and thus measure the magnetic properties held in magnetic materials [34]. There are several electron microscopy techniques, however the scanning electron microscope (SEM) and transmission electron microscope (TEM) are prevalent in use. Generally both microscopes consist of 3 main components, a source which consists of either a filament or a field-emission gun, lenses which could be electromagnetic lenses to diverge the beam of electron and detectors which is either an electron detector or a fluorescent screen. Since electrons require to move freely within the microscope, vacuum is needed inside the microscopes [34]. Scanning probe microscopy consists of scanning tunneling microscopy.

3.4.1 Scanning Electron Microscopy

Scanning Electron microscopy with Polarised Analysis

Figure 3.3 shows a scanning electron microscopy with polarised analysis (SEMPA). It takes the measurement of spin polarization for the secondary electrons which are released from the sample when a sharp, unpolarized beam of scanning electron microscope scans over the sample. These secondary electrons are highly responsive to topography, thus the total current measured from these electrons provide topographic details and information.No external field needs to be applied and it is a surface technique since it requires secondary electrons which are released from a depth of 1nm [34].

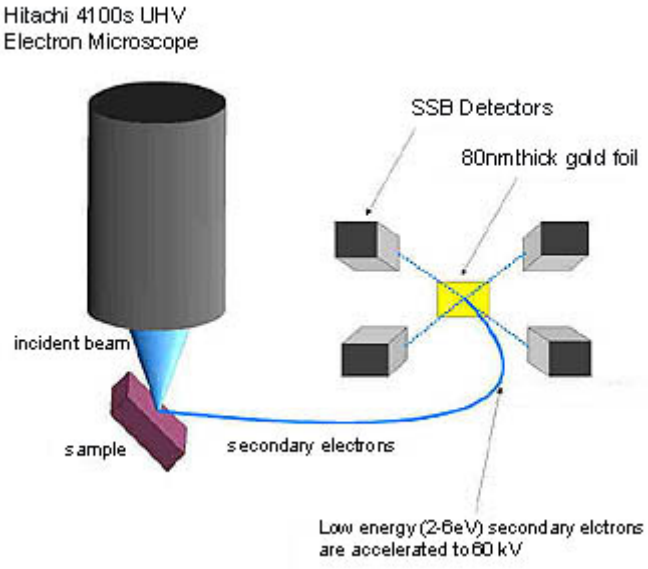

Figure 3.3: Working of a SEMPA.

3.4.2 Transmission Electron Microscopy

Lorentz Transmission Electron Microscopy

Lorentz force is a phenomenon where an electron feels a force in the presence of an electric field and a magnetic field. Due to this force, the electron faces a deviation. The deflection angle is related to the in-plane magnetization and also detects the thickness of the sample. Lorentz Transmission Electron Microscopy (LTEM) is based on this technique [34]. Domain walls and domains are studied easily using Lorentz microscopy at high resolutions [35]. Figure 3.4 shows an imaging of skyrmion crystal under LTEM microscopy. Lorentz microscopy has two modes: the Fresnel mode that provides observations of magnetisation ripples and domain walls, and the Foucault mode in which the domains are captured as an image. Figure 3.5 shows schematic of electron path in Fresnel mode [34].

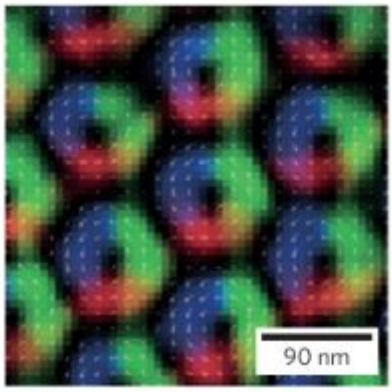

Figure 3.4: Real-space observation of skyrmion crystal under LTEM microscope.

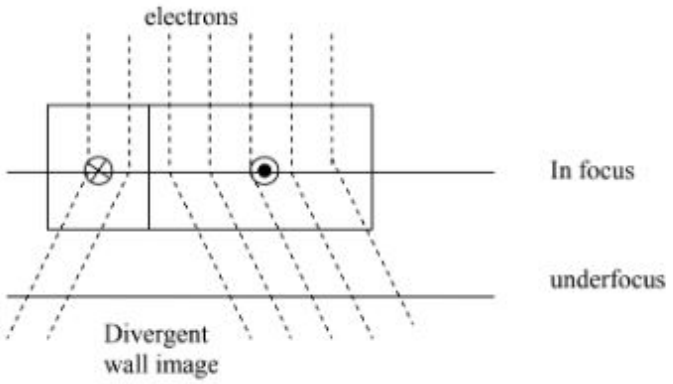

Figure 3.5: Schematic of electron path through a magnetic sample in fresnel model.

Electron Holography

Electron beams which belong to the energy range of 10^3 to 10^7 eV, there is a particular electron-matter interaction which is basically the deviation of electron wavelength that happens due to the presence of magnetic and electrostatic fields. This change in wavelength results into a phase change in the wave that is relative to the phase of the wave in vacuum. Compared to other transmission electron microscopy, the images do not provide any information on phases. However, phase information is provided clearly by the holography technique. Electron holography basically operates based on the creation of an interference pattern between the scattered electron wave of the specimen and a standard wave of known form [34].

3.4.3 Scanning Tunneling Microscopy

Figure 3.6 shows a set up of a scanning tunneling microscope (STM). It has a very simple working methodology. Here, a bias voltage is given between a metal tip and the conducting sample which is to be explored. When the tip and sample surface are close to each other, a tunneling current is produced due to quantum mechanical

tunneling effect. This current is dependent on distance, therefore it is highly precise while measuring tip to sample proximity using the tunneling current. A piezoelectric actuator is placed with the tip of the STM which are externally coordinated by electronics and have the ability to move in three dimensions. The tip is moved over the sample surface keeping the tunneling current unchanged. Positions of the sample from the tip is measured and provides a three-dimensional image of the surface of the sample. Thus, this techniques uses 3 concepts: scanning, point probing and vacuum tunneling [36], [37]. Figure 3.7 shows spin spirals under an STM [38].

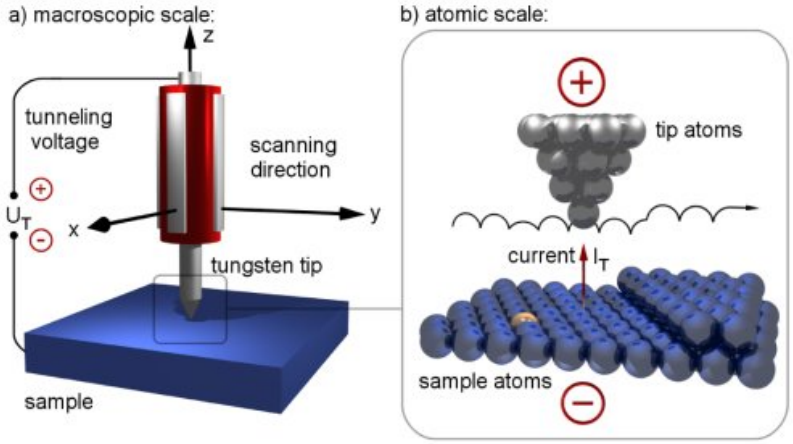

Figure 3.6: Scanning tunneling microscopy a) Macroscopic scale b) atomic scale.

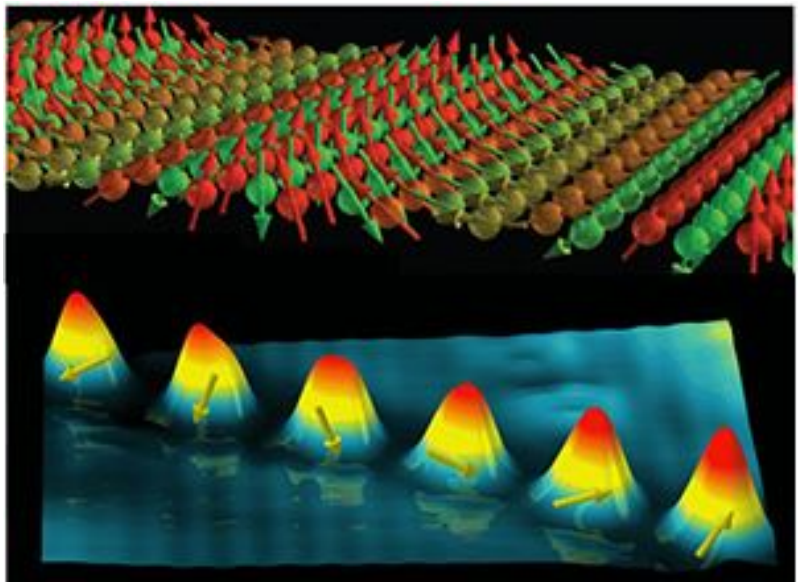

Figure 3.7: Spin spirals under spin-polarized scanning tunneling microscopy.

Chapter 4

Proposed Methodology

4.1 Architecture of Proposed Model

Figure 4.1 shows the overall work plan that has been executed through out the thesis period. First of all, data sets have been generated due to their unavailability on repositories. After generation of data sets, pre-processing is done on them. Then, a classifier is applied on the data sets. Three different classification approaches have been used for which the results are analyzed and classifier that provides the highest accuracy is chose as the best one.

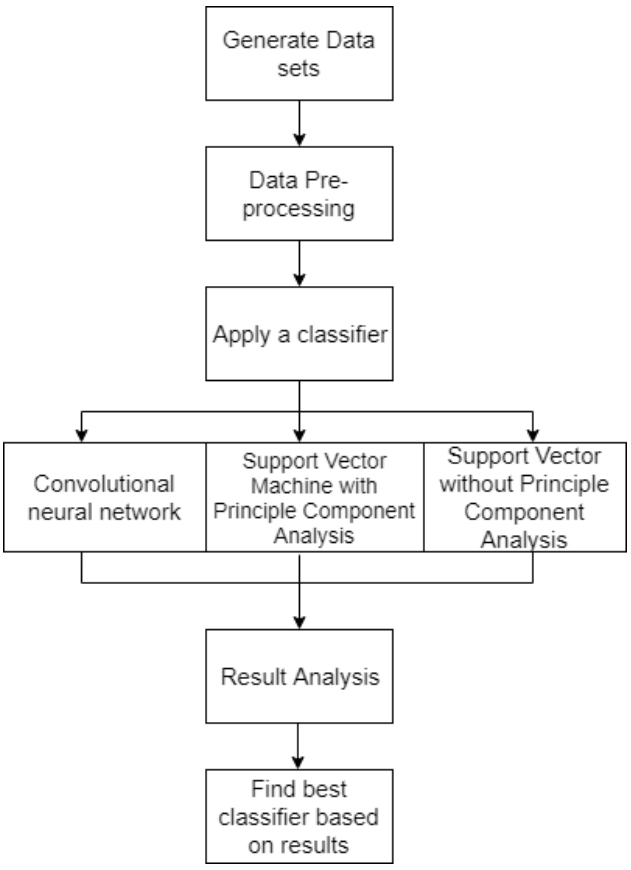

Figure 4.1: Diagram presenting the proposed model.

4.2 Data Set Generation

The validity of a machine learning approach completely depends on the data and how it was obtained because it translates into the transferability and universality of the output model. One of the challenges of using an ML approach in this field is the availability or lack thereof of data sets. Considering the novelty of our approach, it was not possible to find a data set that would suit our specific needs. Since we do not have access to an LTEM or similar microscopy equipment, it wasn't feasible for us to collect our own data from physical observations. The solution to this problem was to simulate own data from the physical properties of the magnetic configurations. This approach has precedent in existing literature [4], [5], [39]–[43] and is actually preferred in many cases because 1) numerous 'exotic' magnetic phases or excitations of matter like skyrmions are harder to generate physically in materials, 2) it is challenging to manipulate these phases at room temperature [44] and 3) we might not able to create as many variations in them physically because of this challenge.

The workflow of Figure 4.2 shows how the data set was generated. For a basic idea, all the structural formulae of the interested magnetic configurations were collected firstly. Then vectors were created using those formulae. Variations and deformities have been added by slightly changing certain variables in the formulae. Vectors were then plotted. The azimuth and elevation angles were also changed to add more variation to the images. Finally the generated images were stored.

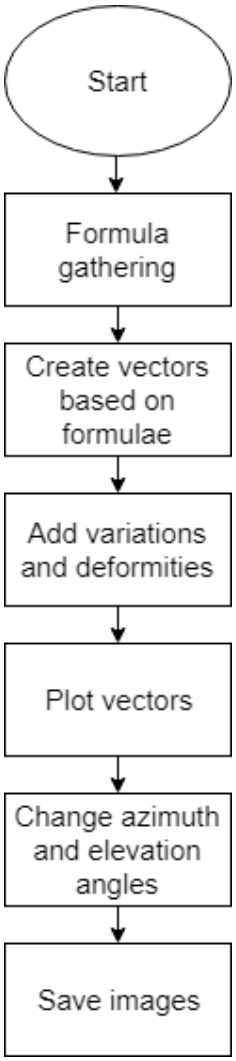

Figure 4.2: Workflow diagram of data set generation.

4.2.1 Our Approach

Since the problem we wanted to solve at the core was an image classification problem, we had to generate images. In other words, we needed to generate images of different magnetic configurations with enough small variations in each one of them so that we can detect a slightly changed or deformed phase. We already knew the mathematical formulations of all the magnetic configurations as described in chapter 3. We had to translate them into code and use plotting to generate our data set.

The general code structure was to initialize all the variables first. Then we created a loop which would iterate through all the x, y, z positions in our lattice. Then for each position we would calculate the x, y, z and s_x, s_y, s_z values using formulations of each different magnetic configuration. We could set the number of spins in each direction of the rectangle which would let us increase or decrease the number of individual simulated spins. The following paragraphs describe how different configurations were generated.

4.2.2 Skyrmion & Antiskyrmion

Skyrmions and antiskyrmions are similar in the sense that only their vorticity or 'm' changes as described by the groundbreaking work done in [45]. For every position in the lattice, we had to calculate the value of Φ in eq (4.1).

$$\Phi(\varphi) = m\varphi + \gamma \tag{4.1}$$

We would get this value with $m = \pm 1$ and different γ values. What is important here is the 'm' value, when we keep the 'm' value +1 we get skyrmions and when we keep the 'm' as -1 we get antiskyrmions as demonstrated in Fig 4.3 [45]. The helicity γ value was chosen as 0, π, -$pi/2$ and $pi/2$ for different variations. Lastly, we would give φ as input which would the positional argument of x, y which would be turned into a single value $\theta \ni -\pi < \theta \leq \pi$ and, for some $r > 0$.

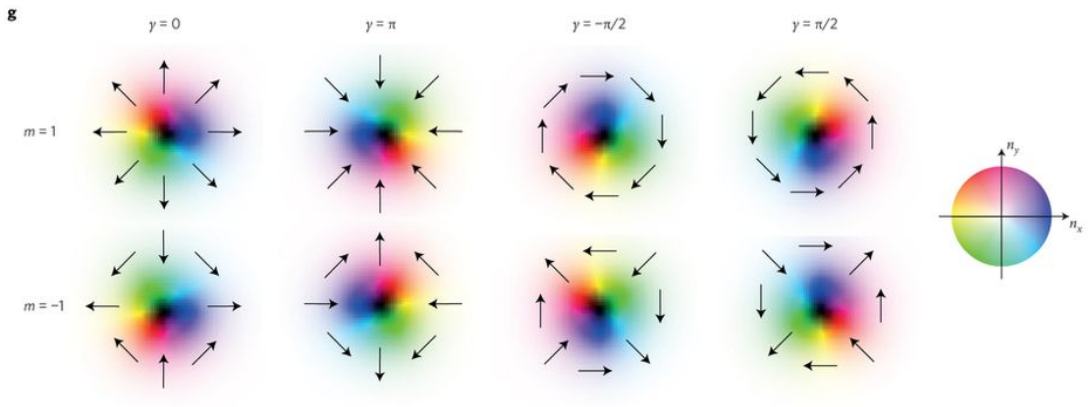

Figure 4.3: Using different gamma and m = (-1, 1) we generated different configurations of skyrmion and antiskyrmion.

Finally we would get three components of the directional vector when we insert the value from eq (4.1) into (4.2).

$$\mathbf{n}(\mathbf{r}) = (\cos\Phi(\varphi)\sin\Theta(r), \sin\Phi(\varphi)\sin\Theta(r), \cos\Theta(r)) \tag{4.2}$$

We would get three vectors s_x, s_y, s_z for each position in the axis. We use lattice size of 25 x 25 which means we get 25 spins while plotting skyrmions or antiskyrmions. The final result is shown in Fig 4.4.

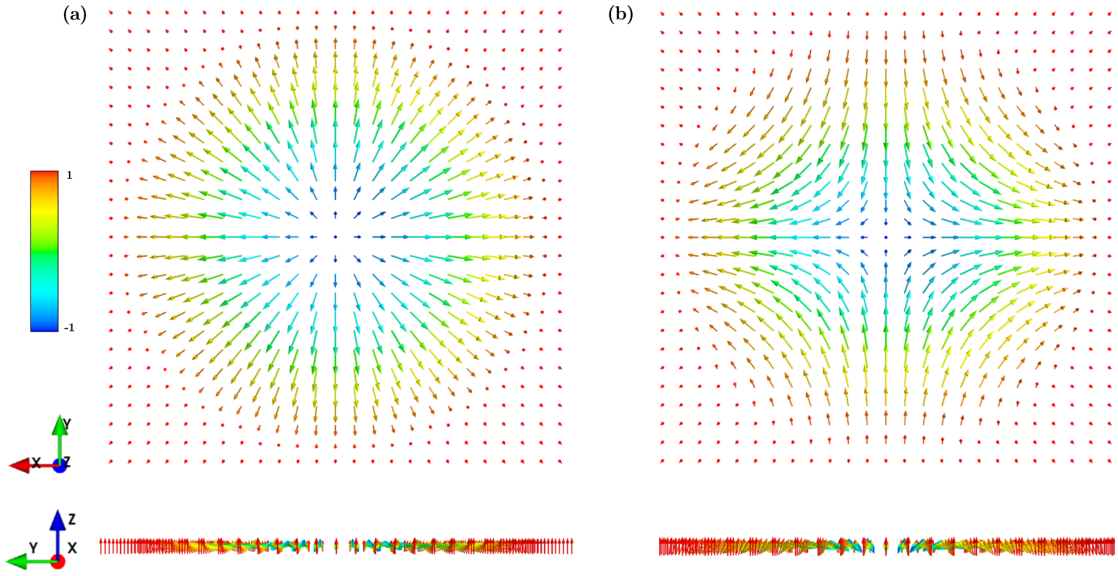

Figure 4.4: Images show (a) skyrmion with m=+1, γ=0 and (b) antiskyrmion with m=-1, γ=0. The colorbar represents the direction of the spins. At the top we have a top view and at the bottom we have a cross-sectional view of the lattice.

To get a representative data set of real magnetic interactions, we had to introduce some deformities in the magnetic configurations. The way we did it was simple, we multiplied each vector component by a random number between 0 and 1. This would give each individual spin different random values and in turn would generate a deformed magnetic configuration. Fig 4.5 shows the results. This contributed to the robustness and universality of our data set.

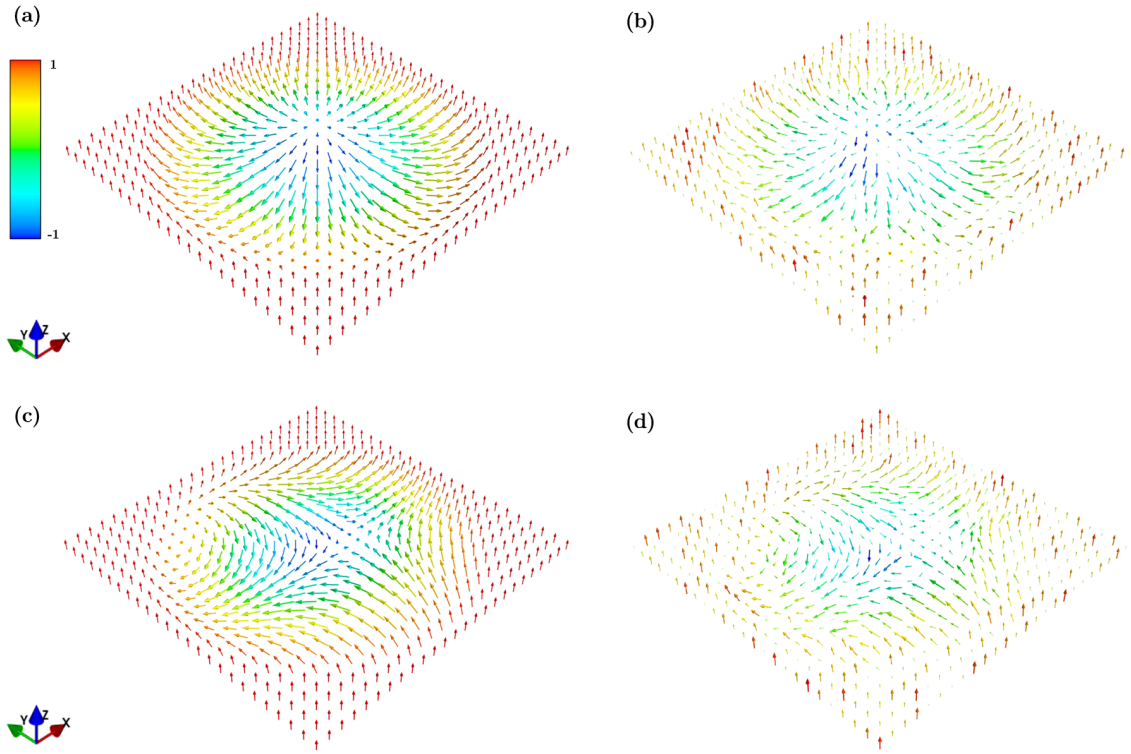

Figure 4.5: Figures show (a) magnetic skyrmion, (b) deformed skyrmion, (c) anti-skyrmion, (d) deformed antiskyrmion configurations that were used in training.

4.2.3 Spin Spiral

Spin spiral as described in chapter 3 is a type of magnetic configuration where the magnetization keeps rotating throughout the surface by a constant angle. From eq (3.2) we can get the numeric values of three components of the directional vector. Where the θ^{μ} value denotes the direction of the vector with respect to the x and y axis. For our generated images, we used an 45° as the value of θ^{μ}. The $\mathbf{q} \cdot (\mathbf{R}^n + \boldsymbol{\tau}^{\mu})$ value is basically the starting angle of the first spin. We increment α^{μ} value to get the "swirling" shape throughout the spins. This is the constant angle by which the spins keep rotating. For our data set generation we used different values for both of them so that we can get different variations. Fig 4.6 shows the end result.

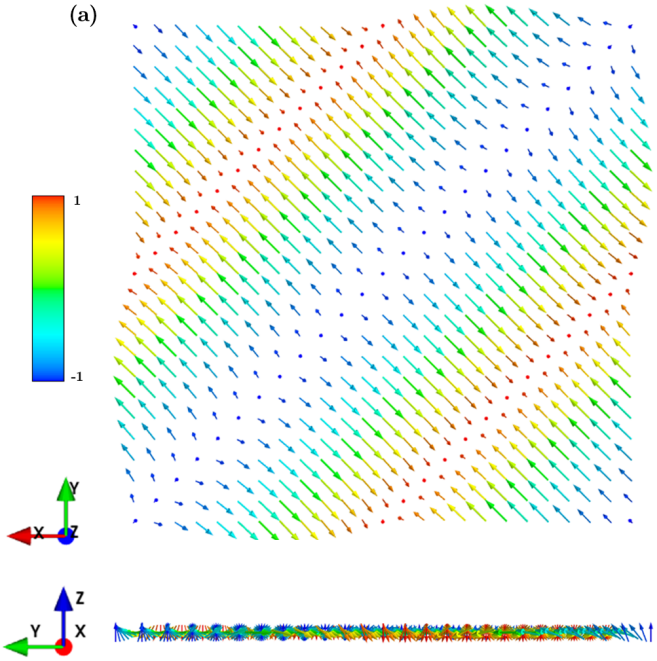

Figure 4.6: Images shows a spin spiral configuration. The color bar represents the direction of the spins. At the top we have a top view and at the bottom we have a cross-sectional view of the lattice.

Similar to how deformities in skyrmions were generated, we multiply each of the vector component by a random value from (0,1) to get random deformities in spin spirals as demonstrated in Fig 4.7.

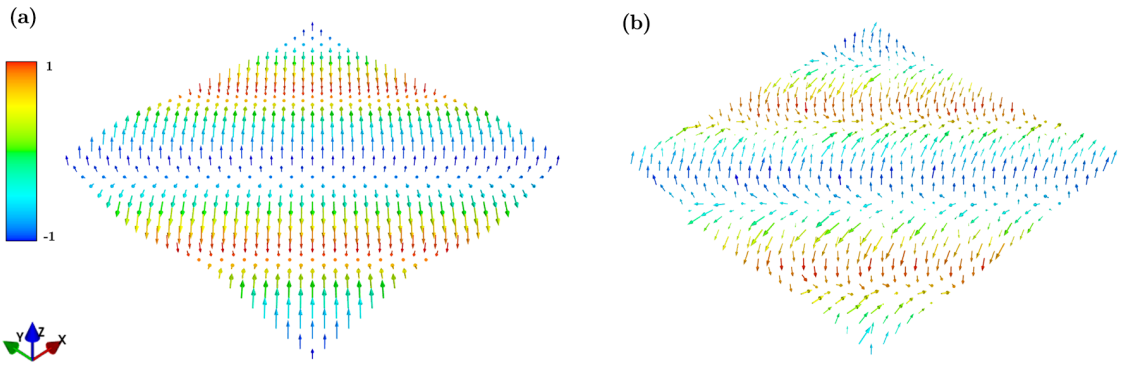

Figure 4.7: Images show (a) spin spiral and (b) deformed spin spiral configurations.

4.2.4 Ferromagnetic & Anti-Ferromagnetic

Ferromagnets are magnets where their atomic dipoles are uniformly arranged. For ferromagnetic configurations, the approach is simple. Direction of all the vectors would be same. Here we introduced some variations by changing the direction of the vector but the fundamental rule was followed which is that all the vectors will have

same directions. On the other hand for the anti-ferromagnetic images the value of the s_z alternates from +1 to -1 or vice versa depending on the neighbouring spin. This change or direction was translated into the code with very simple implementation. The results are in Fig 4.8.

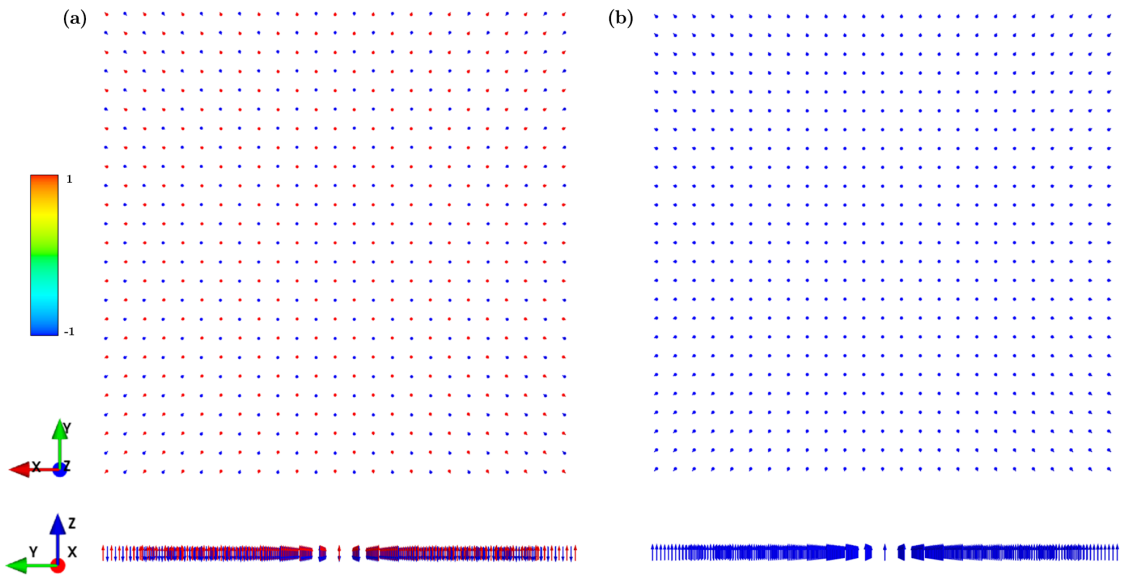

Figure 4.8: Images shows (a) Anti-ferromagnet and (b) Ferromagnet. The colorbar represents the direction of the spins. At the top we have a top view and at the bottom we have a cross-sectional view of the lattice.

Similar to previous configurations, deformities were introduced in the range of (0,1). Fig 4.9 demonstrates the generated images.

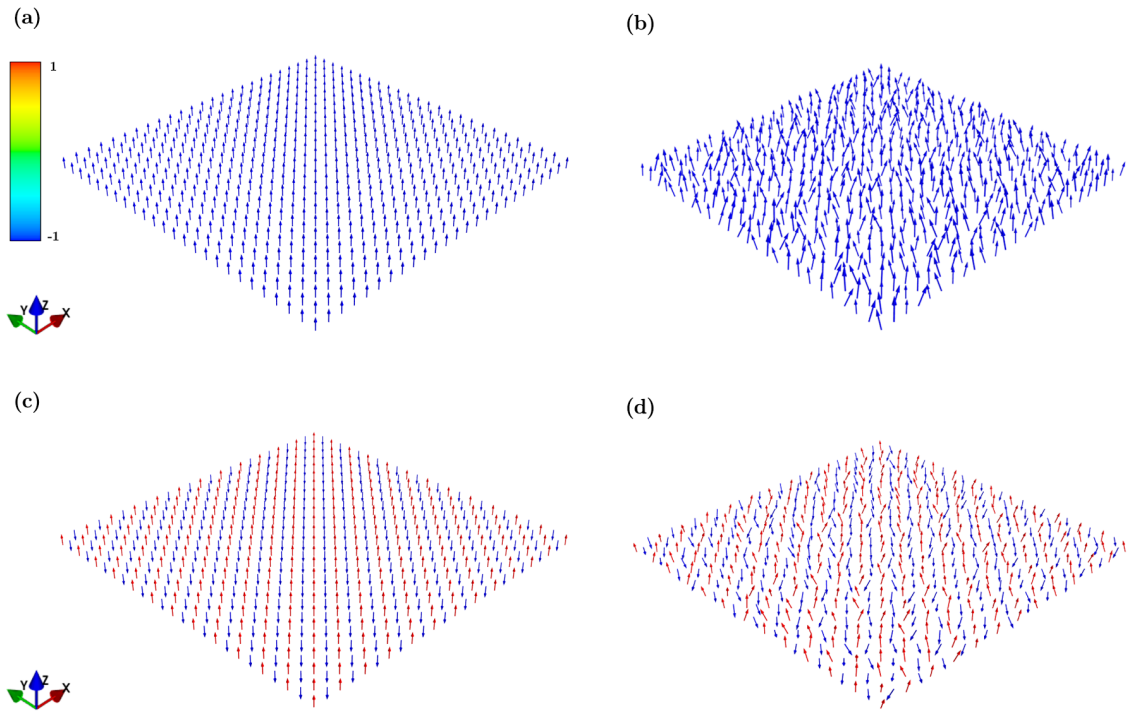

Figure 4.9: Figures show (a) ferromagnetic, (b) deformed ferromagnetic, (c) anti-ferromagnetic, (d) deformed anti-ferromagnetic configurations that were used in training.

4.2.5 Plotting Vectors

One of the challenges in generating the data set was finding a suitable programming language and library that would be able to plot vectors that would resemble a physical scan. We experimented with different programming languages and found python and a library called "mayavi" good for what we needed. After we had the values of all vector components, we had to plot then. We supplied the mayavi "quiver3d" method with arguments u, v, w which were vector components in different axis and x, y, z positional arguments which were the positions of each individual vectors in the lattice.

Another important thing we should mention is the color of the different spins. This was important because the color of the spin helps us to determine the direction of each spin. To achieve this we had to use a color by scalar approach where the spins will be colored depending on their 'w' vector component.

4.2.6 Saving Images

All that was left for us to do was to save images. For this we ran a simple loop. The loop iterated through different azimuth and elevation values to save the configuration from different angles. This variation of viewpoint along with the deformations

discussed earlier gave us a pretty robust data set. Using the "savefig" function we saved all the images in corresponding sub directories for each category.

4.3 Image Pre-processing

In ML, image pre-processing is done to enhance or improve the image data. Since our approach was to use a data set that was computer generated, our images retained all the color or pixel information very well. So we used very simple image pre-processing methods on the image like resize and rescale.

4.3.1 Use of "ImageDataGenerator"

Using this class we can generate image data in batches. We used rescaling to 1/.255 which scales the array of original image pixel values in a range from 0 to 1. Figure 4.10 shows how an input image is transformed into a matrix of pixels.

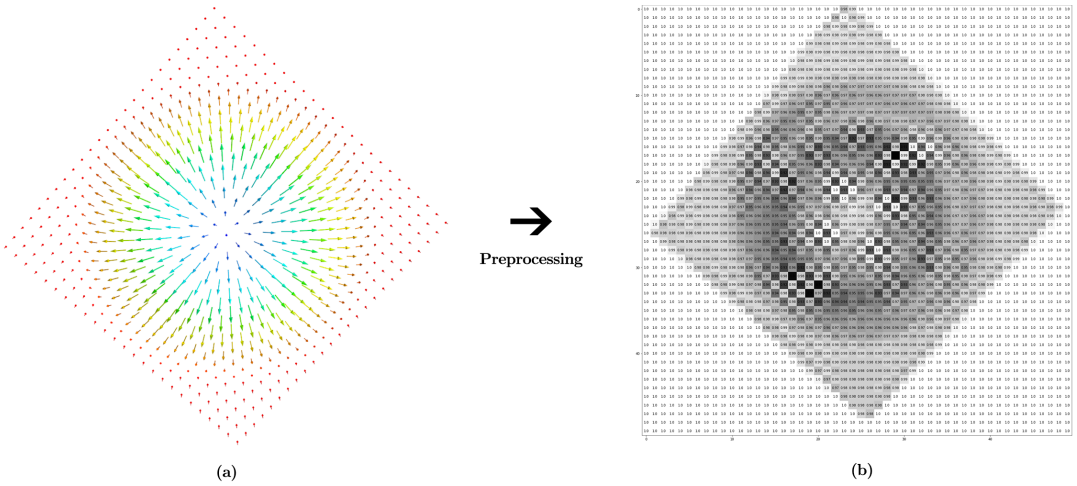

Figure 4.10: Transformation of image into a matrix of pixels a) Original image of a skyrmion b) Pixel matrix of the original image.

4.3.2 Use of "flow_from_directory"

This method is used so that it can take path to a directory and batches of data which is augmented are generated. The directory will be the target directory where all the data set will be present and there are sub directory for each class. The sub directory tree of our data set will be included in the generator which will generate pixels from our image data. Using target size attribute we resized all the images of our data set so that all the images will be of the same size which we input to the network. We used 250×250 as our target size. We learned through trial and error that this image size works best for us when training a convolutional neural network

in terms of training time and accuracy. For the batch size we defined 128 as batch size which defines the batch size of our data. We used color mode as RGB and so the channels will be converted to 3 channels. For Grayscale color mode, the images will be converted to only 1 channel. Categorical class mode is used in our model as we are classifying more than 2 types of images that will return 2D one-hot encoded labels.

4.4 Implementation of Convolutional Neural Network

4.4.1 First Convolutional Layer

This is the first convolutional layer of our network. A matrix of pixels generated from the images is taken as the input. We used 16 filters and stride of 3 to get feature maps using the filters. All input images were resized to 250 * 250 pixels so that all the input remain same. No important feature is left alone as we used a large input size as researches show that while using a large data set, this approach can give higher accuracy at the end [46]. At the last "use bias" is declared false to exclude bias vector from this layer. ReLU has been used in this layer since its efficient when it comes to classification problems and converges quickly [47], [48].

4.4.2 Second Convolutional Layer

In our second convolutional layer, 32 filters are used. Stride of 3 is used to generate feature maps. ReLU is used in this layer as well.

4.4.3 Max Pooling Layer

Here network takes input of feature maps that is generated from the convolutional layer and gives smaller feature maps containing very significant features to work with. We used two max pooling layers accompanying the two convolutional layers with pool size of 3. Pool size is the size of the window of our max pooling.

4.4.4 Flattening Layer

We used one flatten layer after the second and last max pooling layer to form the column matrix. This is given as input to the next layer, known as fully connected layer.

4.4.5 Fully Connected Layer

We used two hidden layers using Dense package, where we used 128 neurons in the first layer and 256 neurons in the second layer. We increased the number of units in the second layer as it can increase reliability of the architecture. Thus using two hidden layers makes the network deeper. And "use bias" is made false so that bias vector is not present in the layers. Batch normalization is applied after the first dense layer, which normalizes the activation from the previous layer for each and every batch and maintains the mean of the activation close to 0 and standard deviation of activation close to 1. Batch normalization beats the original model by very significant margin as it achieves the accuracy better than the original one in less steps [49]. Both the hidden layers are activated using ReLU activator. The last layer consists of another Dense layer with 5 units as we have 5 categories to classify. In this layer, softmax activator is used as it maps the outputs to a range from 0 to 1 which leads to a probability distribution. Figure 4.11 shows the architecture that has been used in this paper.

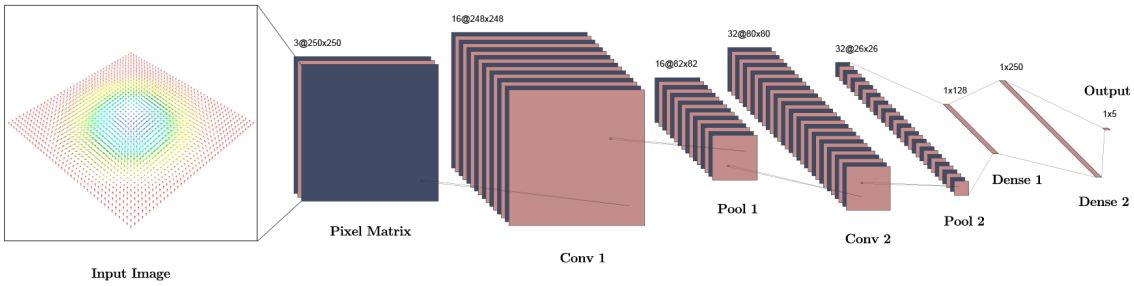

Figure 4.11: The CNN architecture proposed in this paper.

4.4.6 Saving The Model

Using "model.save", the model is saved for future predictions. It contains the whole architecture of the model which can re-create the model while doing further predictions, saves the weights and training configurations such as loss and optimizer of the model and gives optimizer the permission to resume its work from where it stopped the last time.

4.4.7 Optimizer

We used Adagrad as optimizer accompanied with a specific learning rate and it adapts with how frequently the parameter gets updated. The more frequently the parameter gets updated, the learning rate gets smaller [50]. We used 0.001 learning rate as our initial learning rate.

4.4.8 Loss Function

We used categorical crossentropy as loss function when we classified 3 to 5 images using our networks. This loss function is used as this works for multi-class classification. This is also named as "Softmax Loss". We used binary crossentropy as loss function when we classified only 2 types of images using our networks. This is a sigmoid activation and also used for multi-label classification. It is called binary as it works for only two types of classes in data set.

4.5 Implementation of PCA & SVM

4.5.1 Loading Data

Firstly an array for images and another array for the labels are created, where images and labels are stored in their respective arrays. When the images are loaded in the image array, it is resized to 40×40 and are also defined as RGB. The smaller image size is selected because of memory constraints while loading a big data set like ours and the infeasibility of using batches in scikit-learn.

4.5.2 Standardization of Data & PCA

"StandardScaler" is a function of sklearn class and it is used to standardize the features with the help of removing the mean of the features and lastly scaling the features to a unit variance. Standardization is performed using equation 2.6.

PCA is used to speed up the machine learning algorithm and visualize the data [51]. As redundant data from the images can affect the performance of learning, we wanted to find the optimal number of components using PCA. Variance is a statistical term which depicts exactly how faraway the set of data and numbers is spread out. Explained variance is the inconsistency between actual data and model. In Fig 4.12 it is evident that for first 100 components, we have expected variance of more than 5. So for General model (Sk,AnSk,Fe,AnFe,Sp) we used 100 components for PCA.

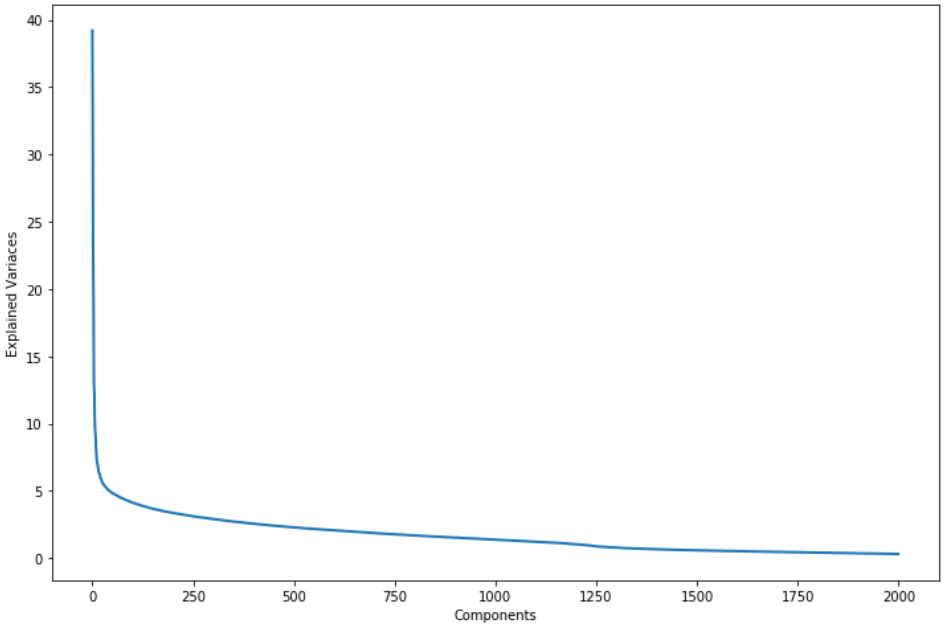

Figure 4.12: Explained variance vs number of components graph for General model (Sk,AnSk,Fe,AnFe,Sp)

In Fig 4.13 it is evident that for first 50 components have expected variance of more than 5. So for the model of non-topologically-protected images we used 50 components for PCA.

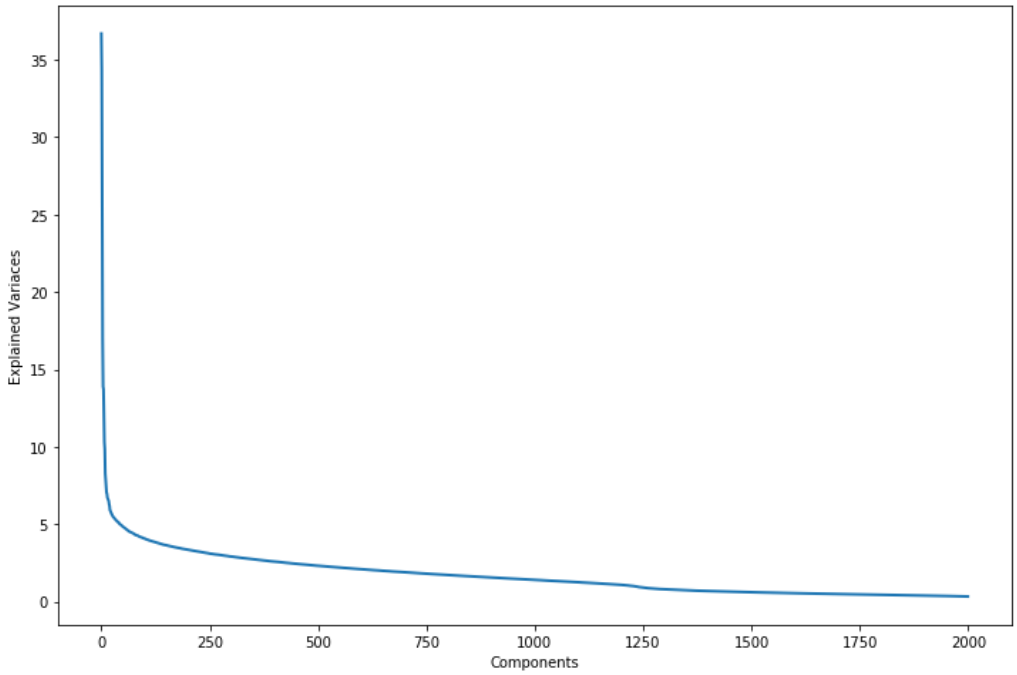

Figure 4.13: Explained variance vs number of components graph for non-topological model

In Fig 4.14 it is evident that for first 400, we get the higher variance. So for the model for topological images we used 400 components for PCA.

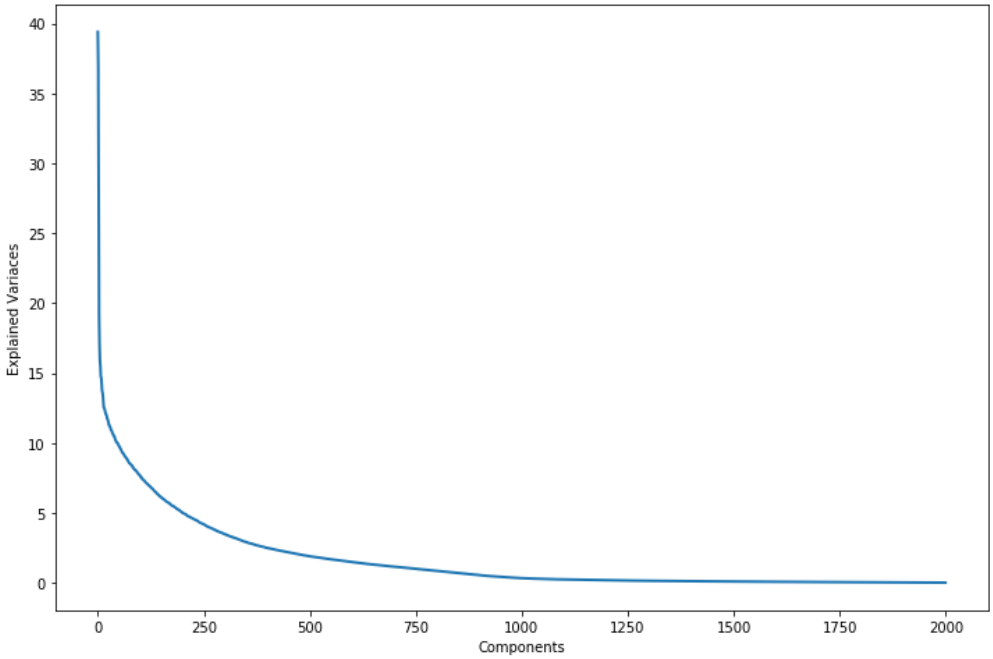

Figure 4.14: Explained variance vs number of components graph for topological model

This gives a clear view of the number of components for the models so that classification can be done faster using SVM. Though SVM can be implemented without PCA, but it would require more time as opposed to running SVM with PCA since the former will take all significant as well as insignificant components for classifying the images given to the model. Then all the images which are scaled with "StandardScaler" are fitted to the model with fit_transform method.

4.5.3 Splitting The Data Set

Next step is to split the data in test and train set. For both cases of SVM with PCA & SVM without PCA, there are 10000 images for each type of image of train set gets 5250 images, 1750 images in test set for each type and 3000 images for validation set. When PCA is used it contains (40 X 40 X (components used in the model)) components, where in SVM without PCA it contains 40 X 40 X 3 components. So it means PCA reduces the components by getting rid of insignificant components, whereas SVM contains all the components. After splitting the data set, we feed the images into an SVM classifier.

Chapter 5

Result and Analysis

5.1 CNN

5.1.1 Training

For CNN, we have a total of 50000 images where each category contains 10000 images. For every category, we split data into ratio of 70-15-15 where 70% of data belongs to training set, 15% in testing set and rest 15% in validation set. The last 15% is kept for validation. Figure 5.1 demonstrates the accuracy and loss vs epoch graphs for topological models in both RGB and grayscale modes. Also, both the train and test accuracy have been plotted. Accuracy for RGB model is highest at 2 epochs and for grayscale, it is at 6 epochs. The training was stopped at 6 epochs for the grayscale model because we reached the optimum accuracy and any further training leads to the training loss increasing. In RGB, loss for train data is somewhat constant while loss for test data heavily fluctuates.

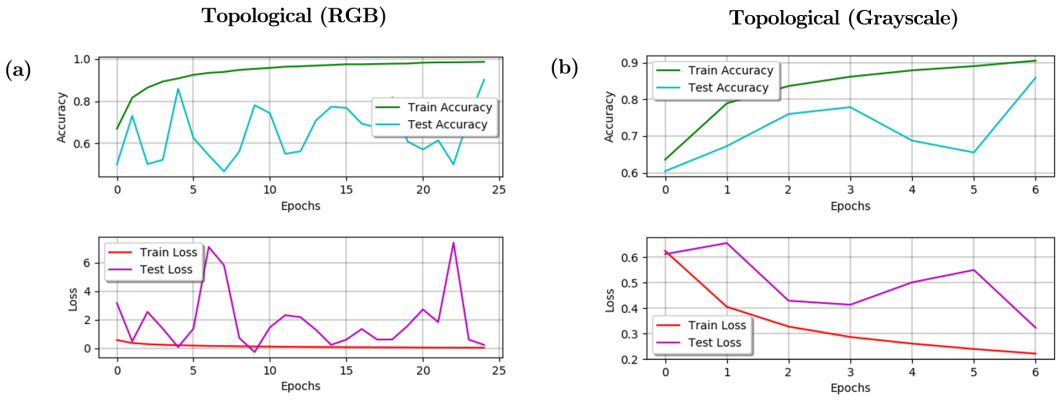

Figure 5.1: a) Accuracy vs Epoch and Loss vs Epoch graphs for topological model in RGB mode b) Accuracy vs Epoch and Loss vs Epoch graphs for topological model in grayscale

Figure 5.2 demonstrates the accuracy and loss vs epoch graphs for non-topological models in both RGB and grayscale modes. In this case, accuracy was highest with 1 epoch in RGB and with close to 40 epochs in grayscale. Loss was constantly falling in RGB for test data but remained somewhat constant for train data. In grayscale, there was an exponential decrease in loss for both test and train losses.

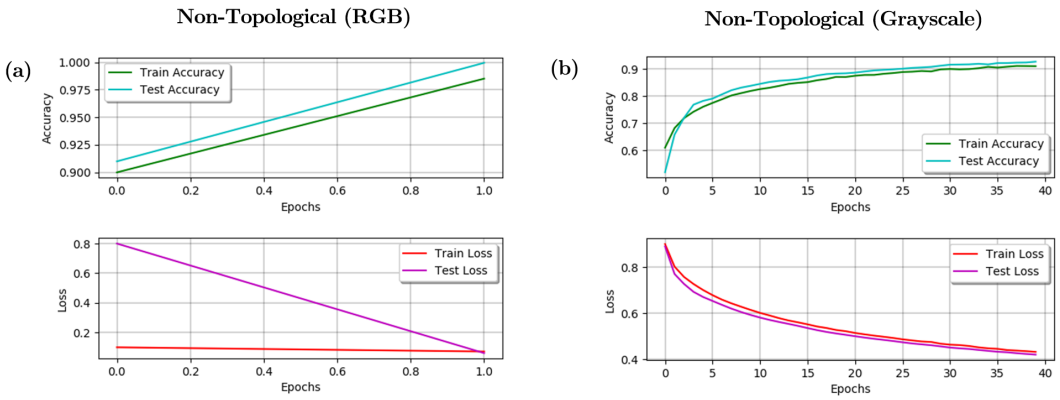

Figure 5.2: a) Accuracy vs Epoch and Loss vs Epoch graphs for non-topological model in RGB mode b) Accuracy vs Epoch and Loss vs Epoch graphs for non-topological model in grayscale

Figure 5.3 shows the accuracy and loss vs epoch graphs for train and test data for the general model. In both RGB and grayscale, accuracy for both train and test data is highest around 40 epochs. Loss tends to decrease over these number of epochs for both cases except fluctuations occur in loss for test data in grayscale.

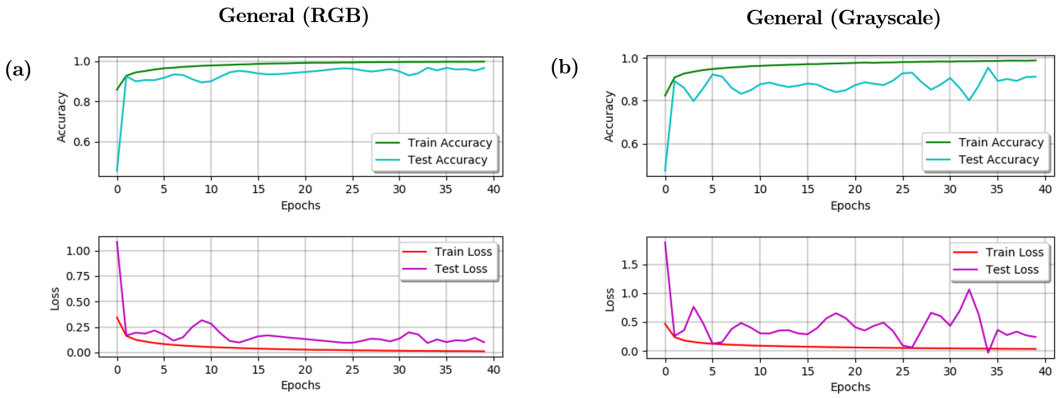

Figure 5.3: a) Accuracy vs Epoch and Loss vs Epoch graphs for general model in RGB mode b) Accuracy vs Epoch and Loss vs Epoch graphs for general model in grayscale

5.1.2 Results

In the scope of our thesis, we mainly divided the classification task into three different parts. Our initial interest was to classify different magnetic configurations from scanning microscopy methods. We train our models separately on topological and non-topological magnetic structures. This approach led us to the discovery that machine learning approach is not only suitable for classifying topologically protected magnetic structures and non-topological structures separately but can also efficiently classify when both types of structures are mixed together. Validation is done on unseen data to estimate how well the model will perform on new data. The goal is to see if the model is overfitting on the training data and whether it would be able to classify on actual real world data. We set aside 15% of our data set (randomly) for validation of each model.

First, we trained two different models namely Topological (Sk,AnSk) and Non-topological (Fe,AnFe,Sp) using respective data sets. For Topological (Sk,AnSk) model, we attained 98.66% accuracy on train data, 90.20% accuracy on test data and 0% accuracy on validation data. For Non-topological (Fe,AnFe,Sp) model, we got 98.53% accuracy on training data, 99.96% accuracy on test data and 99.97% accuracy on the validation data. In our last model we attempted to classify five different categories of magnetic configurations. We were able to attain 99.82% accuracy on train data, 96.67% accuracy on test data and 96.87% accuracy on validation data with this General (Sk,AnSk,Fe,AnFe,Sp) model.

CNN Model	Accuracy%					
	RGB			Grayscale		
	Train	Test	Val	Train	Test	Val
Topological (Sk,AnSk)	98.66	90.20	89.23	90.53	85.90	85.63
Non-topological (Fe,AnFe,Sp)	98.51	99.95	99.97	90.99	92.64	92.31
General (Sk,AnSk,Fe,AnFe,Sp)	99.82	96.67	96.87	98.52	94.76	94.16

Table 5.1: CNN Classification Accuracy

In addition to using the original images in RGB to train our model, we also experimented using a single color channel or grayscale images. Although imaging techniques like Lorentz transmission electron microscopy produces a scan where different colors correspond to different spin directions, we figured it would be insightful to find out if a convolutional neural network can classify the direction of the spin without having the color information of the spins. The high accuracy achieved from grayscale images demonstrate the reliability of our architecture when it comes to classifying different magnetic structures.

5.1.3 Confusion Matrix

F_1 score is used to depict the balance between precision and recall as this is the function of recall and precision where uneven distribution in class can also be found. Precision is the ratio of predicting something "yes" and being correct at the same time. Recall is also called as "Rate of True Positive" which means predicting an

image as Skyr while it is actually Skyr type. Recall is the ratio of predicting a type while it is actually of that type.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5.1}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive+False Positive}} \tag{5.2}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{5.3}$$

In equation 5.1 it multiplies both recall and precision which is found from 5.3 and 5.2. Then it is doubled multiplying by 2 and divided with the sum of recall and precision. That is how F_1 can be found for each type mentioned in each confusion matrix. In 5.3 True positive value for each type is divided by the addition of true positive and false negative to find recall value. On the other hand in 5.2 true positive is divided by the addition of true positive and false negative to find precision value.

In Fig 5.4 in the RGB confusion matrix Anti_Skyr has the highest F_1 score which is 0.90 having the highest recall value of 0.99 and lowest precision of 0.83. For Skyr the F_1 score comes down to 0.88 but it has highest precision of 0.98. The scenario changes for the grayscale confusion matrix as the color of the spins go away. Here Anti_Skyr gets the highest F_1 value of 0.87 having highest recall value of 0.95 and precision of 0.80 which is lowest between these two types. On the other hand Skyr has the lowest F_1 score of 0.84 having lowest recall value of 0.77 and highest precision value of 0.93.

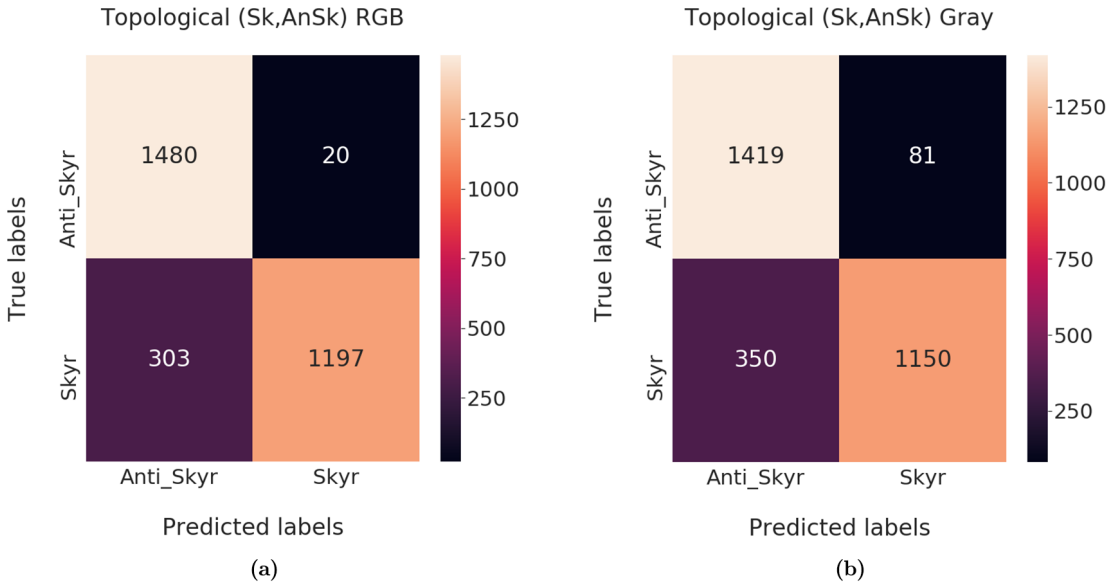

Figure 5.4: Topological confusion matrix (a) RGB images (b) Grayscale images

In Fig 5.5 in the RGB confusion matrix all of the types FM, Anti_FM and Spin_Spiral has the same F_1 score which is perfect 1 given that all have the same precision and

recall value which is perfect 1 too. But in the gray confusion matrix the scenarion changes as the spins do not have any color to distinguish the change in spin. Here Spin_Spiral has the highest F_1 score of 0.96 having highest recall value of perfect 1 and precision value of 0.92. For FM F_1 score goes down to 0.92 having precision of 0.95 and recall of 0.89. The lowest F_1 score is 0.89 achieved by Anti_FM having lowest precision value of 0.90 and lowest recall value of 0.88. In the latter case FM and Anti_FM have less F_1 value as they both are pretty close.

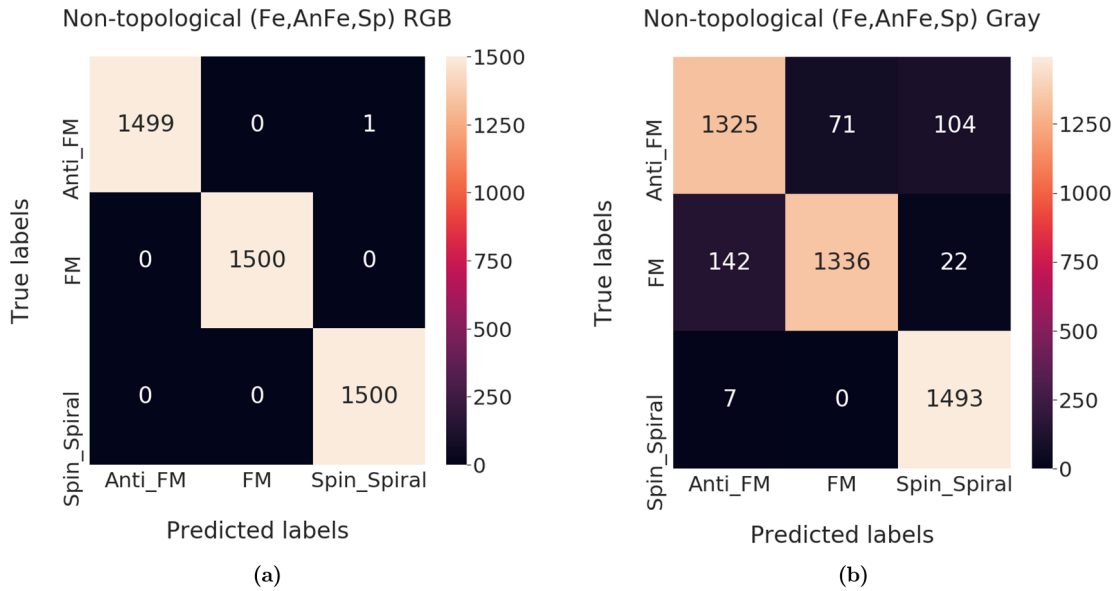

Figure 5.5: Nontopological confusion matrix (a) RGB images (b) Grayscale images

In Fig 5.6 we can see that in the RGB confusion matrix that Anti_FM, FM and Spin_Spiral have highest F_1 score which is perfect 1 and precision and recall is also 1 for these types. For Anti_Skyr and Skyr the F_1 score is lowest which is 0.92, given that Anti_Skyr has precision of 0.94 and recall of 0.90 and Skyr has precision of 0.90 and recall of 0.94. The scenario changes for the grayscale as the color of the spin does not play a role like RGB. For the grayscale confusion matrix F_1 score is highest for Spin_Spiral which is perfect 1 and very slight change for FM and Anti_FM which is 0.99 for both types. For Spin_Spiral recall value is 0.99 and precision value is perfect 1. For Anti_FM and FM the recall value is same which is 0.99 just like Spin_Spiral, but precision value is 0.99 and perfect 1 respectively. Skyr and Anti_Skyr are complex images which reflects in their F_1 scores which are 0.80 the lowest and 0.85 second lowest among the five types in the model. The precision values of Skyr and Anti_Skyr are 0.97 and 0.75 respectively where 0.75 is the lowest among the five types. On the other hand 0.98 is the recall value for the Anti_Skyr and 0.69 is the lowest value for Skyr.

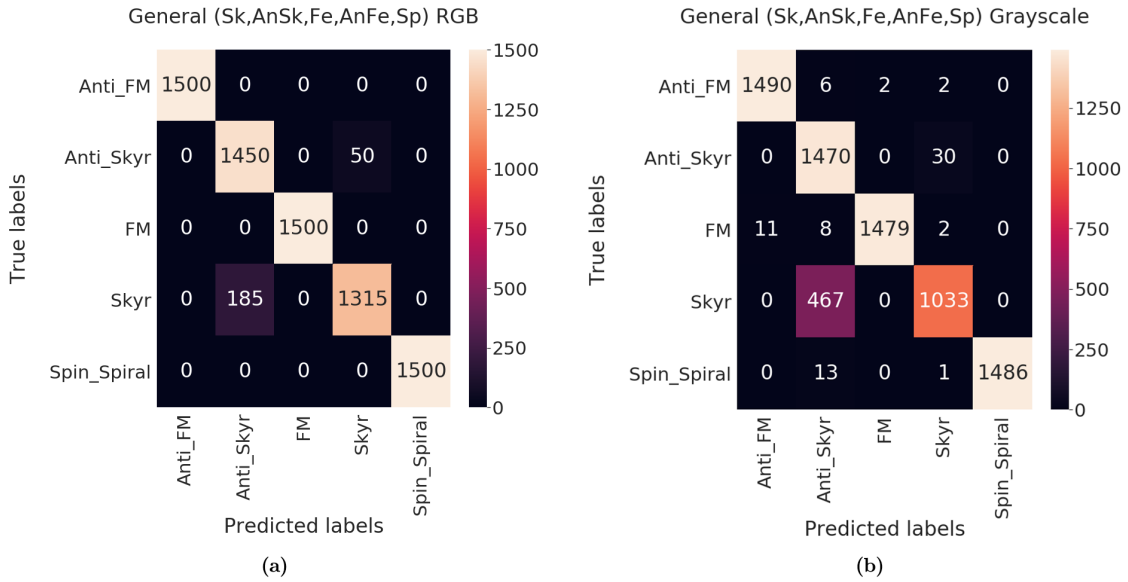

Figure 5.6: All Type confusion matrix (a) RGB images (b) Grayscale images

5.1.4 Analyzing The Network

Convolutional Neural networks are complex computational graphs often with large number of parameters. Interpreting what features the CNN is extracting can be tough and abstract at times [52]. CNN models work on low level characteristics like the intensity value of pixels which can be hard for humans to comprehend or visualize. The difference in workings of high level and low level features in a CNN can be contrasting [53]. It is important to understand what is happening in every step of the way to optimize performance and increase classification success rate.

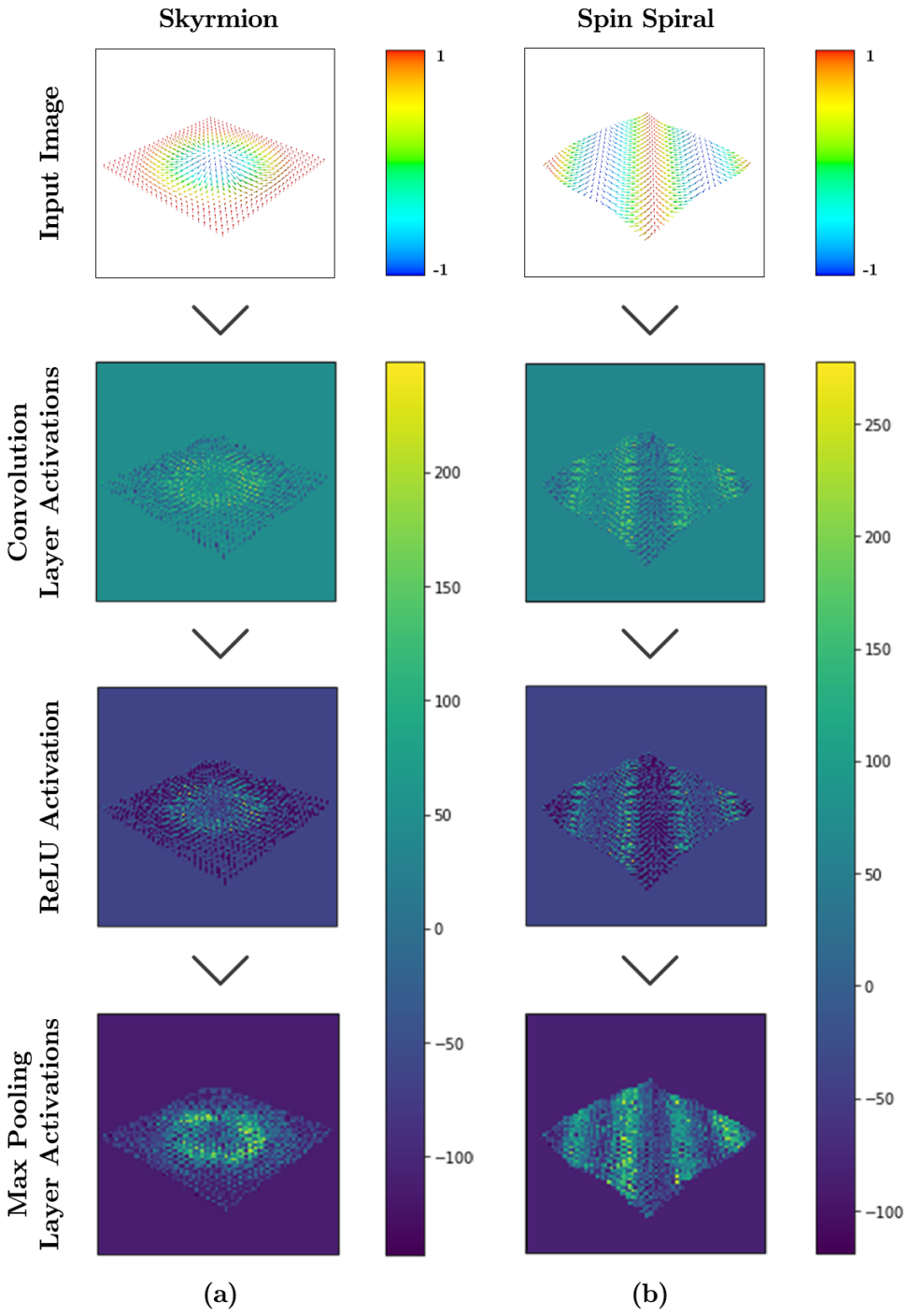

Figure 5.7: Examples of activations in the convolution layers (a) top panel is an example of topological magnetic configuration (skyrmion) and (b) is an example of non-topological magnetic configuration(spin spiral). For both, the image gets passed through a convolution layer, ReLU layer and a max pooling layer where features are extracted.

To visualize what our model is observing, we run the images from our data set through the General (Sk,AnSk,Fe,AnFe,Sp) classifier to figure out what each layer of the model is seeing. In other words we tried to visualize the activations or outputs

and gradients for each layer of the network. Generally in a CNN, low level filters work as edge detectors and high level filters detect complex shapes like objects or faces. Fig 5.7 shows what features a filter has extracted at each level. Note that we used 16 filters of size 3x3 in the convolution layer when training our model, for this demonstration we are showing what features only one of the filters is extracting. According to the color bar, yellow color regions are where the network is putting the most emphasis on or where it is getting 'activated' the most. If we look at the images and where the filters are getting activated, we can see that the region of interest for the convolution, ReLU and max pool layer is where the direction of the spins change. For skyrmions, the region of interest is a circular shape in the middle where the spins change from 1 to -1. For spin spiral, the network gets activated where the magnetic spins change direction but it is a different pattern because in spin spirals, the direction continuously changes from 1 to -1 and vice-versa. The takeaway from this is that the accuracy of our models are reliable because CNN learns the classification task similarly as humans would. It looks at the transitional region and tries to figure out what type of magnetic configuration it is.

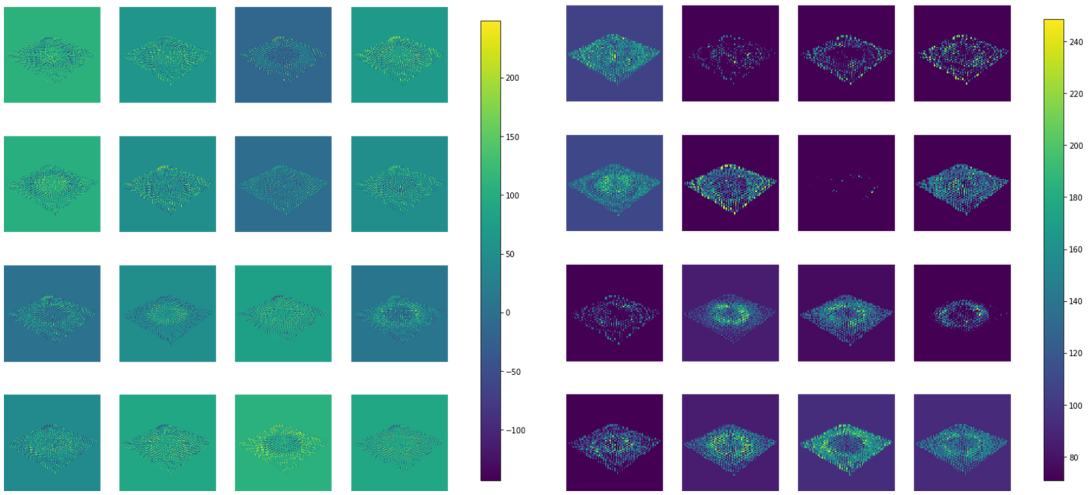

Figure 5.8: Features extracted from the image

Fig 5.8 shows the features extracted at convolution and max-pooling layers of the network. We used 16 filters at the first convolution layer which genrated 16 feature maps from the input. Some of these filters are detecting edges, shapes and different color channels. Then we use a layer of max pooling with 16 pooled feature maps. We add one more convolution and max-pooling layer and then feed the outputs into a series of fully connected layers.

5.2 SVM

5.2.1 Training

During the training of the SVM classifier we used "GridSearchCV" for parameter tuning. With "GridSearchCV" we were able to test all the combinations of parameters to get the optimal parameters for our specific problem. For the multi-class classifications in our data set (Non-topological (Fe,AnFe,Sp) & General (Sk,AnSk,Fe,AnFe,Sp)), we used "LinearSVC" wrapped in "OneVsRestClassifier" and for the lone binary classification (Topological (Sk,AnSk)), we used "SVC" with 'rbf' kernel. We then see the accuracy of our classifier using "accuracy_score".

5.2.2 Results

For SVM, we divided the classification task into three different parts. We train our models separately on topological, non-topological and then on all of the magnetic structures. Additionally, we repeat the process associating PCA algorithm and check the results once for SVM with PCA and once without PCA. Table 5.2 show the accuracy for both of the approaches. For original data (SVM without PCA), for the Topological (Sk,AnSk) model, we attained 79.97% accuracy on train data, 67.60% on test data, and 68.55% on validation data. For Non-topological (Fe,AnFe,Sp) model, we got 99.88% on train data, 99.58% on test data, and 99.75% on validation data. In our last model we attained 89.79% on train data, 83.22% on test data, and 83.38% on validation data.

For data obtained from PCA, for the Topological (Sk,AnSk) model, we attained 87.44% on train data, and 80.14% on test data, and 70.52% on validation data. For Non-topological (Fe,AnFe,Sp) model, we got 99.78% on train data, 99.12% on test data, and 96.51% on validation data. In our last model we attained 78.64% on train data, 75.12% on test data, and 67.56% on validation data. SVM is used

SVM Model	Accuracy%					
	Original Data			PCA		
	Train	Test	Val	Train	Test	Val
Topological (Sk,AnSk)	79.97	67.60	68.55	87.44	80.14	70.52
Non-topological (Fe,AnFe,Sp)	99.88	99.58	99.75	98.78	99.12	96.51
General (Sk,AnSk,Fe,AnFe,Sp)	87.97	83.22	82.38	78.64	75.12	67.56

Table 5.2: SVM Classification Accuracy

because it finds optimal boundaries among all possible number of outputs. In other words, SVM can efficiently classify with the help of complex data transformations. In addition to that non-linear SVM using non-linear kernel gives a high reliable output. SVM was used with PCA as one of our proposed methodologies. PCA is a technique used to downsample very high-dimensional data sets. This is done to reduce the number of unnecessary components and thus make computations much faster. SVM with PCA gave better results for topological model only. SVM without

PCA had higher accuracy for the general model. However, for non-topological both approaches gave similar results.

5.2.3 Confusion Matrix

In Fig 5.9 in the SVM confusion matrix Anti_Skyr has the highest F_1 score of 0.69, highest recall value of 0.70 and lowest precision value of 0.68. On the other hand Skyr has the lowest F_1 score of 0.68, highest precision of 0.69 and lowest recall value of 0.67. In the SVM (PCA) confusion matrix Skyr both has F_1 score of 0.81 having same recall and precision value which is also of 0.81. Here it is evident that using PCA we got better result as PCA used the significant components losing most of the insignificant components.

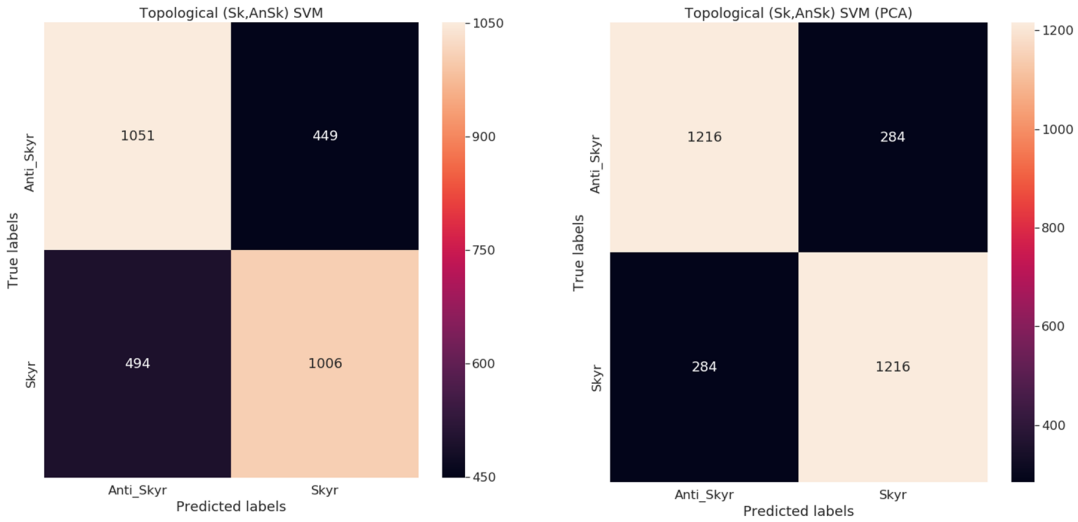

Figure 5.9: Topological confusion matrix (a) SVM (b) SVM (PCA)

In Fig 5.10 in the SVM confusion matrix F_1 scores for Anti_FM, FM & Spin_Spiral all are the same and that is perfect 1 having recall value of perfect 1 also. For the precision value FM & Spin_Spiral have the perfect 1 value, but for Anti_FM it slightly changes and get value of 0.99. On the other hand for SVM (PCA) confusion matrix Spin_Spiral gets the F_1 score as perfect 1 while for Anti_FM & FM it slightly changes and gets value of 0.99 for both. Same type of value is found for their recall and precision value just like their F_1 score. So it is clear that for both SVM & SVM (PCA) F_1 scores remain same and performs likewise.

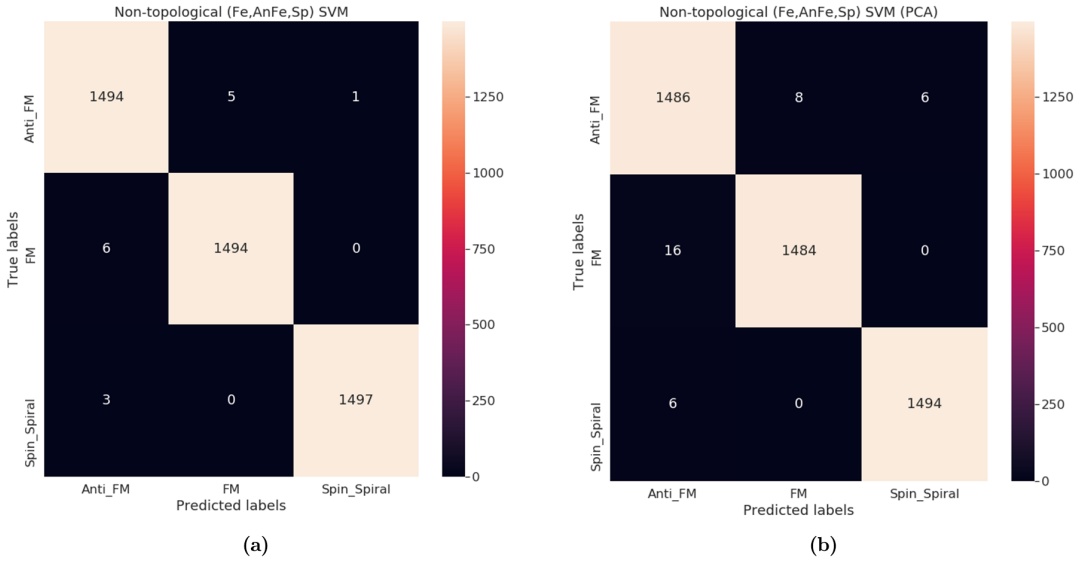

Figure 5.10: Nontopological confusion matrix (a) SVM (b) SVM (PCA)

In Fig 5.11 in SVM confusion matrix FM & Spin_Spiral got highest F_1 score of 0.98. On the other hand Anti_FM got F_1 score of 0.97 and the lowest two scores are achieved by Skyr & Anti_Skyr which are 0.60 & 0.59 respectively. The values plays likewise role for precision as well where Spin_Spiral got the highest value of perfect 1, FM got the second highest value of 0.99 and Anti_FM achieved value of 0.98. But Anti_Skyr & Skyr got lowest value which is 0.59 for both. For recall value FM, Anti_FM & Spin_Spiral got value of 0.98, 0.95 & 0.96 respectively where FM got the highest value. Like precision & F_1 score Skyr & Anti_Skyr got lowest two recall values of 0.62 & 0.60. On the other hand in SVM (PCA) confusion matrix scenarion remains almost same where FM achieves highest F_1 score of 0.99 with highest recall & precision value of 0.99 among the five types. Anti_FM & Spin_Spiral got F_1 score of 0.91 & 0.87 respectively. Anti_FM got recall value of 0.92 & precision value of 0.89. While Spin_Spiral gets recall value of 0.90 & precision value of 0.83. For Anti_Skyr the F_1 score falls down to 0.58 with precision of 0.50 & recall value of 0.68. The lowest F_1 score is achieved by Skyr which is 0.34 with the recall value of 0.26 & precision value of 0.49. Here it is evident in both cases that for topological types results are not good enough, which shows result as worst in SVM (PCA) as it removes the insignificant components which may play a role in SVM where all the components are present. Thus for classifying all five types together SVM plays a better role than SVM (PCA).

5.2.4 Comparative Analysis Of Different Approaches

Analysing the results of our approaches we can see that the CNN approach performs better on the harder classification of topological configurations. This is harder because skyrmions and antiskyrmions are very similar in terms of their spin configurations. The SVM classifier does surprisingly well on the classification on non-topological magnetic structures. This is due to the fact that the ferromagnetic,

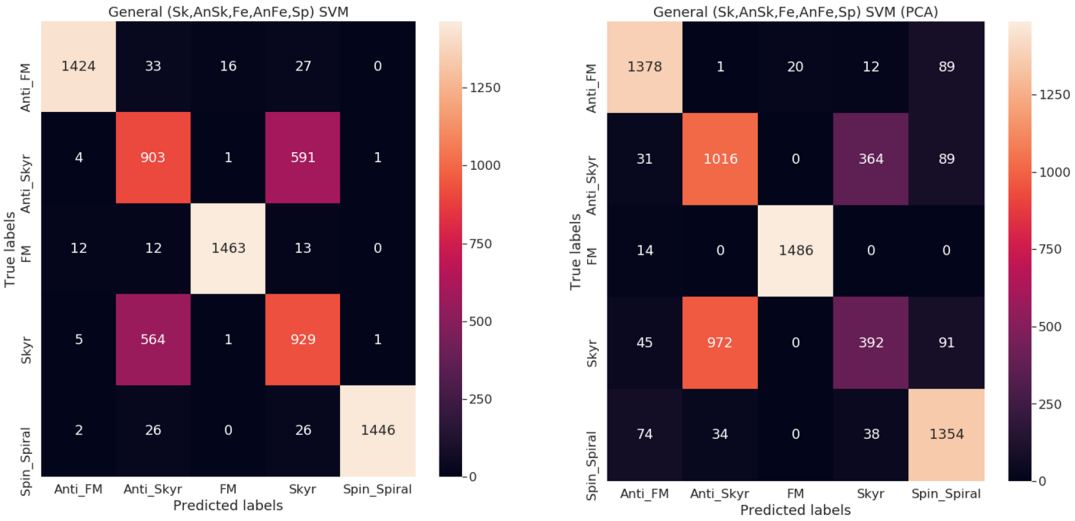

Figure 5.11: General confusion matrix (a) SVM (b) SVM (PCA)

anti-ferromagnetic and spin spirals are visually not very similar. Now for the general model, the CNN classifier outperforms the SVM classifier.

Chapter 6

Conclusion and Future Works

6.1 Concluding Remarks

In this paper, we investigated multiple machine learning approaches to identify different magnetic structures from image data. The proposed system achieved remarkable accuracy in detecting topological and non topological magnetic structures. We applied several machine learning algorithms such as CNN, SVM and PCA. Different models provided different accuracy. In our experimental results we found that CNN provided the highest accuracy compared to other approaches. In addition to that, we developed a method to generate 2D images of different magnetic configurations from their respective physical properties. The generated data would be used as input for the models.

6.2 Future Works

For future works, we plan on improving the model so that it can classify among even higher number of categories existing in magnetism including their transitional states. Moreover, other neural network architectures like capsule network (CapsNet) [54] can be used to classify since it is more accurate than a typical CNN as it can detect size, orientation, directions of shapes or vectors.

Bibliography

[1] G. Torlai and R. G. Melko, "Learning thermodynamics with boltzmann machines", *Phys. Rev. B*, vol. 94, p. 165 134, 16 Oct. 2016. DOI: 10.1103/PhysRevB.94.165134. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.94.165134.

[2] K. Mills, M. Spanner, and I. Tamblyn, "Deep learning and the schrödinger equation", *Physical Review A*, vol. 96, no. 4, p. 042 113, 2017.

[3] L. Wang, "Discovering phase transitions with unsupervised learning", *Phys. Rev. B*, vol. 94, p. 195 105, 19 Nov. 2016. DOI: 10.1103/PhysRevB.94.195105. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.94.195105.

[4] I. A. Iakovlev, O. M. Sotnikov, and V. V. Mazurenko, "Supervised learning approach for recognizing magnetic skyrmion phases", *Physical Review B*, vol. 98, Nov. 2018. DOI: 10.1103/PhysRevB.98.174411.

[5] V. K. Singh and J. H. Han, "Machine Learning Application to Two-Dimensional Dzyaloshinskii-Moriya Ferromagnets", *arXiv e-prints*, arXiv:1806.03749, arXiv:1806.03749, Jun. 2018. arXiv: 1806.03749 [cond-mat.dis-nn].

[6] W. X. Gang, *Quantum field theory of many-body systems: from the origin of sound to an origin of light and electrons*. Oxford University Press, 2007.

[7] A. Al-Masri, *What are supervised and unsupervised learning in machine learning?* [Online]. Available: https://towardsdatascience.com/what-are-supervised-and-unsupervised-learning-in-machine-learning-dc76bd67795d.

[8] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[9] Chris Woodford, *Neural networks*. [Online]. Available: https://www.explainthatstuff.com/introduction-to-neural-networks.html.

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

[11] Assaad Moawad, *Neural networks and back-propagation explained in a simple way*. [Online]. Available: https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e.

[12] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *et al.*, "Learning representations by back-propagating errors", *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[13] *Cs231 convolutional neural network for visual recognition.* [Online]. Available: http://cs231n.github.io/convolutional-networks/.

[14] *Topic dl01: Activation functions and its types in artifical neural network.* [Online]. Available: https://medium.com/@abhigoku10/activation-functions-and-its-types-in-artifical-neural-network-14511f3080a8.

[15] A. F. Agarap, "Deep learning using rectified linear units (relu).", *CoRR*, vol. abs/1803.08375, 2018. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1803.html#abs-1803-08375.

[16] Saimadhu Polamuri, *Difference between softmax function and sigmoid function.* [Online]. Available: http://dataaspirant.com/2017/03/07/difference-between-softmax-function-and-sigmoid-function/.

[17] H. Wu and X. Gu, "Max-pooling dropout for regularization of convolutional neural networks", *CoRR*, vol. abs/1512.01400, 2015. arXiv: 1512.01400. [Online]. Available: http://arxiv.org/abs/1512.01400.

[18] Raghav Pravu, *Understanding of convolutional neural network (cnn) — deep learning.* [Online]. Available: https://medium.com/@RaghavPrabhu?source=post_page----------.jpg.

[19] R. Gandhi, *Support vector machine — introduction to machine learning algorithms.* [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

[20] G. Anthony, H. Gregg, and M. Tshilidzi, "Image classification using svms: One-against-one vs one-against-all", *arXiv preprint arXiv:0711.2914*, 2007.

[21] A. Vora, *Classification – one vs rest and one vs one.* [Online]. Available: https://datastoriesweb.wordpress.com/2017/06/11/classification-one-vs-rest-and-one-vs-one/.

[22] E. Saslow, *Unsupervised machine learning.* [Online]. Available: https://towardsdatascience.com/unsupervised-machine-learning-9329c97d6d9f.

[23] Z. Jaadi, *A step by step explanation of principal component analysis.* [Online]. Available: https://towardsdatascience.com/a-step-by-step-explanation-of-principal-component-analysis-b836fb9c97e2.

[24] S. Krause and R. Wiesendanger, "Spintronics: Skyrmionics gets hot", *Nature materials*, vol. 15, no. 5, p. 493, 2016.

[25] U. Nowak, "Classical spin models", in *Handbook of Magnetism and Advanced Magnetic Materials*. American Cancer Society, 2007, ISBN: 9780470022184. DOI: 10.1002/9780470022184.hmm205. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470022184.hmm205. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470022184.hmm205.

[26] S. Blundell, *Magnetism in condensed matter*, 2003.

[27] G. F. Newell and E. W. Montroll, "On the theory of the ising model of ferromagnetism", *Reviews of Modern Physics*, vol. 25, no. 2, p. 353, 1953.

[28] L. Duò, M. Finazzi, and F. Ciccacci, *Magnetic properties of antiferromagnetic oxide materials: surfaces, interfaces, and thin films.* John Wiley & Sons, 2010.

[29] E. Balkind, A. Isidori, and M. Eschrig, "Magnetic skyrmion lattice by fourier transform method", *arXiv preprint arXiv:1901.02459*, 2019.

[30] R. Wiesendanger, "Nanoscale magnetic skyrmions in metallic films and multilayers: A new twist for spintronics", *Nature Reviews Materials*, vol. 1, no. 7, p. 16 044, 2016.

[31] A. A. Kovalev and S. Sandhoefner, "Skyrmions and antiskyrmions in quasi-two-dimensional magnets", *Frontiers in Physics*, vol. 6, p. 98, 2018, ISSN: 2296-424X. DOI: 10.3389/fphy.2018.00098. [Online]. Available: https://www.frontiersin.org/article/10.3389/fphy.2018.00098.

[32] Y. Zhou, "Magnetic skyrmions: intriguing physics and new spintronic device concepts", *National Science Review*, vol. 6, no. 2, pp. 210–212, Oct. 2018, ISSN: 2095-5138. DOI: 10.1093/nsr/nwy109. eprint: http://oup.prod.sis.lan/nsr/article-pdf/6/2/210/28498387/nwy109.pdf. [Online]. Available: https://doi.org/10.1093/nsr/nwy109.

[33] B. Zimmermann, "Calculation of the dzyaloshinskii-moriya interaction in ultrathin magnetic films: Cr/w (110)", Master's thesis, RWTH Aachen, 2010.

[34] B. Warot-Fonrose, "Magnetic imaging and microscopy electron microscopies",

[35] K. Kirk, S. McVitie, and J. Chapman, "Lorentz imaging of sub-micron patterned elements", in *Magnetic Storage Systems Beyond 2000*, Springer, 2001, pp. 145–152.

[36] R. Wiesendanger and W. Roland, *Scanning probe microscopy and spectroscopy: methods and applications*. Cambridge university press, 1994.

[37] S. Gottardi, *Designing molecular nano-architectures on metals and on graphene*. University of Groningen, 2015.

[38] *Non-collinear spins*. [Online]. Available: http://www.nanoscience.de/HTML/research/noncollinear_spins.html?fbclid=IwAR1IQvNbZ-1D8ZVIiTJz_iPjJhRdWkfEjg7c3kxY
jpg.

[39] P. Zhang, H. Shen, and H. Zhai, "Machine learning topological invariants with neural networks", *Physical review letters*, vol. 120, no. 6, p. 066 401, 2018.

[40] S. J. Wetzel and M. Scherzer, "Machine learning of explicit order parameters: From the ising model to su (2) lattice gauge theory", *Physical Review B*, vol. 96, no. 18, p. 184 410, 2017.

[41] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter", *Nature Physics*, vol. 13, no. 5, p. 431, 2017.

[42] P. Broecker, J. Carrasquilla, R. G. Melko, and S. Trebst, "Machine learning quantum phases of matter beyond the fermion sign problem", *Scientific reports*, vol. 7, no. 1, p. 8823, 2017.

[43] T. Ohtsuki and T. Ohtsuki, "Deep learning the quantum phase transitions in random two-dimensional electron systems", *Journal of the Physical Society of Japan*, vol. 85, no. 12, p. 123 706, 2016. DOI: 10.7566/JPSJ.85.123706. eprint: https://doi.org/10.7566/JPSJ.85.123706. [Online]. Available: https://doi.org/10.7566/JPSJ.85.123706.

[44] D. Maccariello, W. Legrand, N. Reyren, K. Garcia, K. Bouzehouane, S. Collin, V. Cros, and A. Fert, "Electrical detection of single magnetic skyrmions in metallic multilayers at room temperature", *Nature nanotechnology*, vol. 13, no. 3, p. 233, 2018.

[45] N. Nagaosa and Y. Tokura, "Topological properties and dynamics of magnetic skyrmions", *Nature nanotechnology*, vol. 8, no. 12, p. 899, 2013.

[46] K. Ashraf, B. Wu, F. N. Iandola, M. W. Moskewicz, and K. Keutzer, "Shallow networks for high-accuracy road object-detection", *CoRR*, vol. abs/1606.01561, 2016. arXiv: 1606.01561. [Online]. Available: http://arxiv.org/abs/1606.01561.

[47] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines", in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10, Haifa, Israel: Omnipress, 2010, pp. 807–814, ISBN: 978-1-60558-907-7. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104322.3104425.

[48] *7 types of neural network activation functions: How to choose?* [Online]. Available: https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *CoRR*, vol. abs/1502.03167, 2015. arXiv: 1502.03167. [Online]. Available: http://arxiv.org/abs/1502.03167.

[50] J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, Jul. 2011.

[51] T. Howley, M. G. Madden, M.-L. O'Connell, and A. G. Ryder, "The effect of principal component analysis on machine learning accuracy with high dimensional spectral data", in *Applications and Innovations in Intelligent Systems XIII*, A. Macintosh, R. Ellis, and T. Allen, Eds., London: Springer London, 2006, pp. 209–222, ISBN: 978-1-84628-224-9.

[52] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)", *arXiv preprint arXiv:1711.11279*, 2017.

[53] D. Garcia-Gasulla, F. Parés, A. Vilalta, J. Moreno, E. Ayguadé, J. Labarta, U. Cortés, and T. Suzumura, "On the behavior of convolutional nets for feature extraction", *CoRR*, vol. abs/1703.01127, 2017. arXiv: 1703.01127. [Online]. Available: http://arxiv.org/abs/1703.01127.

[54] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules", *CoRR*, vol. abs/1710.09829, 2017. arXiv: 1710.09829. [Online]. Available: http://arxiv.org/abs/1710.09829.