Discovering Phase Transitions with Unsupervised Learning

Introduction to Machine Learning (Monsoon'24)

Dhruv Pisharody* and Satwik Wats†
Student of Physics Department of Ashoka University, Haryana, India

Instructor: Professor Sandeep Juneja
(Dated: December 2, 2024)

Ferromagnetic spin models are used to model real-life magnets. As we increase the dimensionality and interactions of these models, they begin to approximate real ferromagnetic materials better, but also show a tremendous increase in complexity. To study these systems' properties then, particularly their phase transitions and critical behaviour, we cannot use traditional methods that make use of a lot of system parameters. In this report, we used the technique of unsupervised learning to characterize phase transitions in the Ising spin lattice system. This is done by first generating a large raw spin data set generated through the Monte Carlo and Metropolis-Hastings algorithms. We then reduce the dimensionality of this raw data to only the relevant few dimensions, performing Principal Component Analysis. Then, by using the k-means clustering algorithm to identify distinct phases, we characterize the phase transition in our system. This method can discover phase transitions in complex systems without needing us to correctly identify the relevant physical order parameter.

keywords: Unsupervised learning, Phase transition, Ising Model, PCA, Clustering algorithm

Contents

I. Preface

This report focuses on the theoretical and computational implementation of unsupervised learning to discover phase transitions. Phase transitions are often referred to as an abrupt change in the properties of a system as we tune a macroscopic parameter (for example, the temperature) of our system. In the field of statistical mechanics we encounter many such transitions, for example Bose-Einstein Condensation, liquid-gas transition, and ferromagnetic to paramagnetic transition.

An object exhibiting ferromagnetism consists of magnetic moments (spins), each of which tends to align in the same direction as its neighbouring spins, an alignment driven by exchange interactions between neighbouring spins. A common type of mathematical model used to represent this in statistical mechanics, are models with spins placed on a lattice. The different rules these spins can obey, like their possible orientations and the way in which neighbouring spins interact, lead to different such ferromagnetic models. For our system we use the Heisenberg Exchange Interaction which tends to align the spin in the same direction as its neighbouring spin.

When the phase transition is happening between the two phases of the system (from ordered to a disordered phase), it is possible to construct an order parameter that would be compatible with the system and explain its behavior. The order parameter generally vanishes in the disordered phase while being finite in the ordered phase. However, for our case we do not need to worry about order parameter as we are relying on our clustering algorithm to find the distinct phases of our system.

Overall, this report covers two main ideas involving unsupervised learning. First, we talk about how to find relevant data in a huge data set, and second, we talk

* dhruv.pisharody˙asp25@ashoka.edu.in
† satwik.wats˙asp25@ashoka.edu.in; Physics Department, Ashoka University.

about how to separate clusters of data using different clustering algorithm.

II. Problem Statement

Identifying phase transitions in ferromagnetic spin systems is fundamental for the field of condensed matter physics. Most of the techniques that are used in physics are based on identifying the order parameter and a lot of other details about the systems before characterizing the phase transition. However, this might not be always possible because the systems can get very complex as soon as we introduce another dimension or increase the number of interacting objects. Therefore, the question that this raises is whether we can do this without knowing everything about the system. The answer is unsupervised learning techniques from the field of machine learning.

We generate a large raw spin data set for our system, and then using principal component analysis and clustering techniques (specifically k-means clustering), we train our model over them. This approach does not require prior information about the system's order parameter and transition temperature. Clustering algorithms can separate the spin system into two phases (ordered and disordered) at every temperature. Now, the algorithm looks at the structure of the clustering and region of crossover as the temperature is varied, which will identify our critical temperature.

III. Methodology

A. Generating Ising Spin Configuration

We are working with the Ising spin system on a 2D square lattice with periodic boundary conditions (the edges of the lattice wrap around). Our model is represented by the following Hamiltonian:

$$H = -J \sum_{\langle i,j \rangle} s_i s_j \tag{1}$$

Here, J is the interaction strength, s_i is the spin at the i^{th} site having a value ± 1, and the sum is over nearest neighbour pairs of spins s_i and s_j. Now, using a Monte Carlo Simulation which uses the Metropolis-Hasting algorithm we generate our spin configuration at various temperatures T. We begin with an initial system of spins on a lattice, and a temperature T we want to simulate the system at. Choosing a random spin, we perform a trial flip of it. For the Ising model, this is a direct flip.

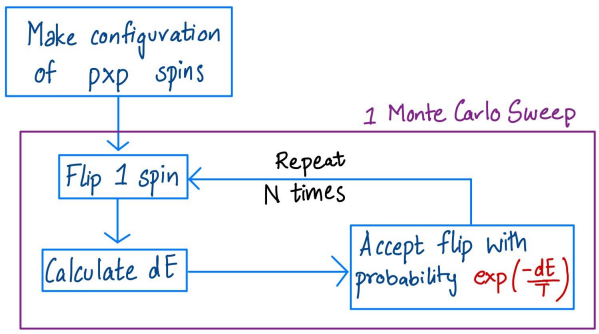

FIG. 1. Flowchart depicting the steps for one Monte Carlo sweep of the Metropolis-Hastings algorithm

We then calculate the change in energy dE this trial flip would cause, by calculating the change in energy of just the rotated spin instead of for the whole lattice. If this dE is negative, i.e. the flip lowers the energy of the lattice, it is accepted. If the dE is positive, it is accepted with a probability $\exp(-dE/T)$, i.e. with a probability drawn from the Boltzmann distribution. Computationally, we achieve this by choosing a random number between 0 and 1, and accept the flip only if it is less than $\exp(-dE/T)$. Whether the change is accepted or rejected, we move on, repeating these steps of choosing a random spin and performing a trial flip N times, where N is the number of spins on the lattice, and this is referred to as one Monte Carlo sweep. In one sweep, on average, each spin is selected and flipped.

The Metropolis-Hastings algorithm allows us to simulate our system at any temperature we want, by running a large number of Monte Carlo sweeps on it at this temperature, and waiting for it to reach equilibrium. This equilibration can be observed by plotting the energy (or magnetisation) of the system after each sweep, upon doing which we observe a stabilisation in the value it takes after about 5000 sweeps.

Here, because we are aware of the transition point we do the Monte Carlo sweeps around the critical temperature value.

B. Principal Component Analysis (PCA)

Now that we have our raw data, we want to use PCA to find the axis in it showing the maximum variance. Mathematically the matrix can be explained in the following manner:

Consider a dataset $\mathbf{X}$ with n data points, each having

d dimensions:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{x}_i \in \mathbb{R}^d.$$

The first step in PCA is to normalize the data by first subtracting the mean of each dimension, and then by dividing by the standard deviation:

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i, \quad \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})^2}$$

$$\mathbf{X}_{\text{centered}} = \frac{\mathbf{X} - \bar{\mathbf{x}}}{\sigma}.$$

However, we do not need to normalize our data. A single configuration of the Ising lattice where the system has reached equilibrium has an equal number of up and down spins, so plotting a histogram of the values of a particular spin gives us peaks at just +1 and -1, i.e. a distribution having a mean of 0 and a standard deviation of 1. Therefore, they are automatically normalized even if they we look at across multiple spin configuration. This can be thought of as following from the ergodic hypothesis that a time ensemble and a space ensemble are statistically the same.

Next, we compute the covariance matrix $\mathbf{C}$:

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^\top\mathbf{X},$$

where $\mathbf{C} \in \mathbb{R}^{d\times d}$ represents the variance and covariance of the features.

We calculate the eigenvalues and eigenvectors of the covariance matrix $\mathbf{C}$:

$$\mathbf{C}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i,$$

where λ_i are the eigenvalues, and $\boldsymbol{v}_i$ are the corresponding eigenvectors.

We sort the eigenvalues λ_i in descending order, and select the top m (two for us) eigenvalues and their corresponding eigenvectors. These eigenvectors form the principal components:

$$\mathbf{W} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m],$$

where $\mathbf{W} \in \mathbb{R}^{d\times m}$.

Project the original data onto the m-dimensional space defined by $\mathbf{W}$:

$$\mathbf{X}_{\text{var}} = \mathbf{X}\mathbf{W}.$$

C. K-means clustering algorithm

Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$, the objective is to divide the data into k clusters, each represented by a centroid $\mathbf{c}_j \in \mathbb{R}^d$ for $j = 1, 2, \ldots, k$.

The k-means algorithm minimizes the following objective function:

$$J = \sum_{j=1}^{k}\sum_{\mathbf{x}_i} \|\mathbf{x}_i - \mathbf{c}_j\|^2,$$

where $\|\mathbf{x}_i - \mathbf{c}_j\|^2$ is the squared Euclidean distance between point $\mathbf{x}_i$ and the cluster centroid $\mathbf{c}_j$.

The algorithm then proceeds iteratively as follows:

1. Select k initial centroids $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k$ randomly:

2. Assign each data point $\mathbf{x}_i$ to the nearest centroid:

$$\mathcal{C}_j = \{\mathbf{x}_i : \|\mathbf{x}_i - \mathbf{c}_j\|^2 \leq \|\mathbf{x}_i - \mathbf{c}_l\|^2, \forall l = 1, 2, \ldots, k\}$$

3. Recompute each centroid as the mean of all points assigned to that cluster:

$$\mathbf{c}_j = \frac{1}{|\mathcal{C}_j|}\sum_{\mathbf{x}_i \in \mathcal{C}_j}$$

where $|\mathcal{C}_j|$ is the number of points in cluster $\mathcal{C}_j$.

4. Repeat the assignment and update steps 2 and 3 until the centroids no longer change or the change is below a specified threshold.

IV. Raw Spin Data

Data generated through Metropolis-Hasting Algorithm:

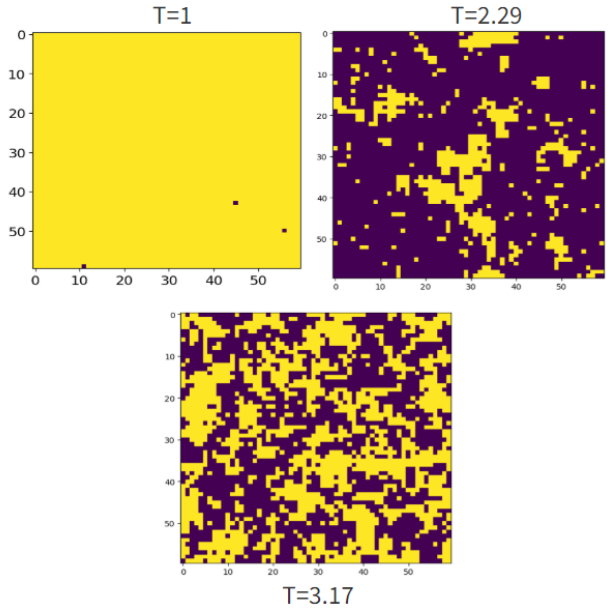

FIG. 2. Single spin configuration for a 60 x 60 lattice for three temperature values. We repeat this for 100 spin configuration and 14 temperature values.

We generate 100 uncorrelated spin configuration for each temperature value we take (total temperature values = 14) and collect them into an $M \times N$ matrix,

$$X = \begin{pmatrix} \uparrow & \downarrow & \uparrow & \dots & \uparrow & \uparrow \\ \downarrow & \uparrow & \downarrow & \dots & \uparrow & \downarrow \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \downarrow & \uparrow & \uparrow & \dots & \uparrow & \uparrow \end{pmatrix}_{M \times N} ,$$

where $M = 1400$ is the total number of samples, and N is the number of lattice sites. This matrix is the only thing we need to put in our model.

V. Model

Our system is trained on the X matrix that we input. First we apply PCA as mentioned in the above methodology section and then plot the scatter colormap of the two main components:

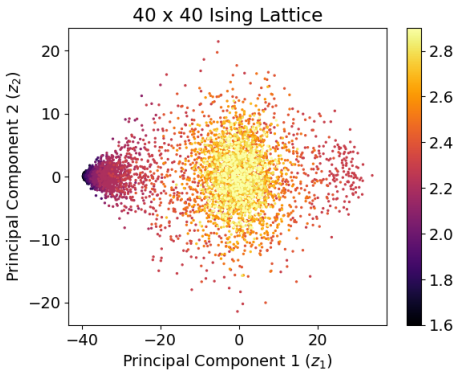

FIG. 3. The colormap of low and high temperature where the spin configuration can be seen as a function of temperature and two major principal component.

We repeat the same steps for different lattice sizes and then, instead of z_1 *vs* z_2, we plot $|z_1|$ *vs* z_2 and z_1^2 *vs* z_2. This is done because the phase transition of the Ising model is showing symmetry breaking that would mean that the negative cluster or positive cluster are the same. Therefore, we fold it across the horizontal axis. It is important to note that here we don't see a symmetric cluster because of the way we initialize our system. Our initial configuration all spins start from one direction and due to the Metropolis-Hastings algorithm it has a low probability of flipping at low temperature, so if we begin with the spins aligned in one direction then the system will be biased in that direction and that is what we see here. However, this is not an issue as we are folding it across the horizontal axis.

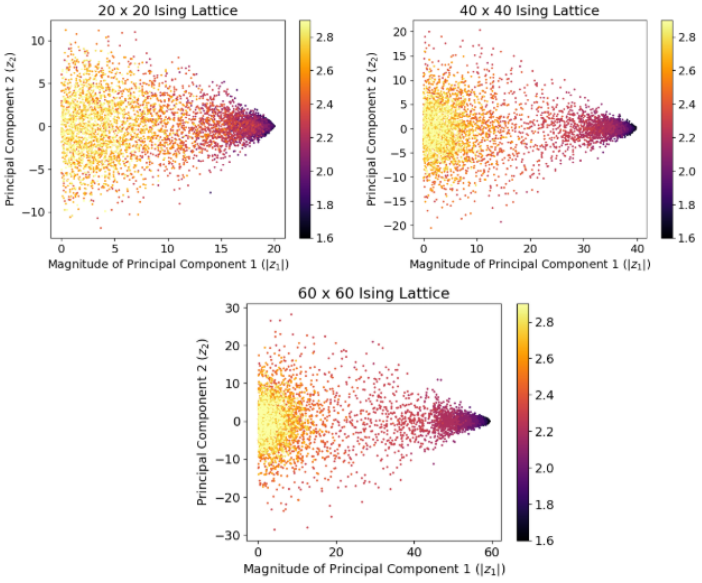

FIG. 4. Magnitude of the major Principal axis plotted for all three lattice sizes, 20×20, 40×40 and 60×60.

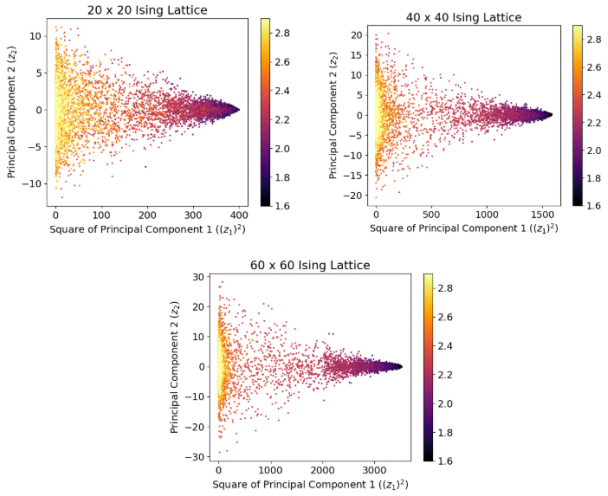

FIG. 5. Square of the major Principal axis plotted for all three lattice sizes, 20×20, 40×40 and 60×60.

It is clear from above that the z_1^2 *vs* z_2 plots have a much sharper and better distinction between the two phases and they will be better for running the k-means clustering algorithm. Therefore, we only run our clustering algorithm on them, but we do it for all three lattice sizes.

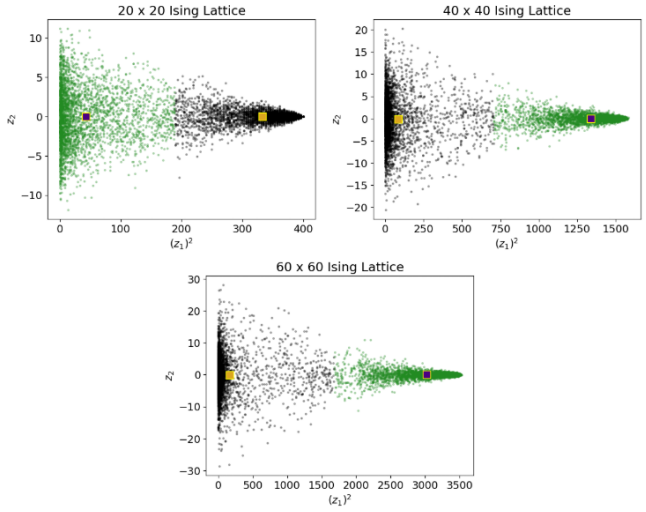

FIG. 6. After applying the clustering algorithm we see that the two clusters have been created with their centroids marked.

VI. Result

From our model above we see that the clustering algorithm has given us a decision boundary that we can use to find our critical temperature. This decision boundary implies that our system has identified that clusters of ordered and disordered phase and the point of decision boundary is the critical temperature.

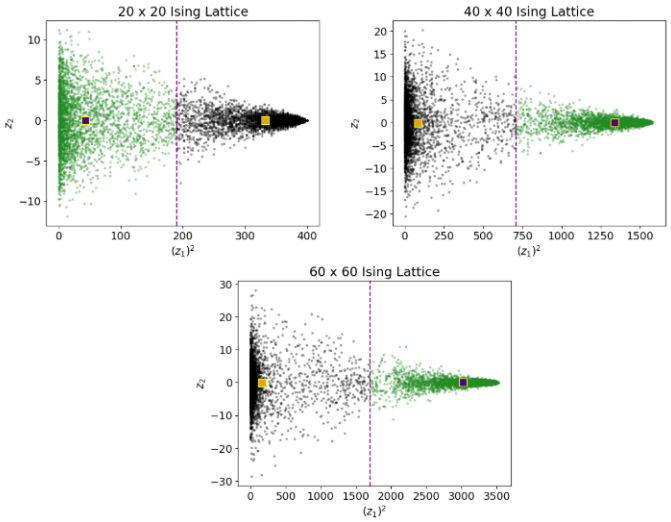

FIG. 7. We make the decision boundary between the two cluster and corresponding to that we form our critical temperature values.

We have identified the critical temperature for the Ising model as 2.36 (for the 20 x 20 lattice), 2.27 (for the 40 x 40 lattice), and 2.28 (for the 60 x 60 lattice), and the actual value of Ising transition is 2.27.

VII. Notes

The PCA method combined with clustering algorithm has provided a faster and more accurate calculation of the transition point. Even for smaller lattice sizes we see a very accurate results and of course as we increase the lattice size the results gets better. We have implemented this for a system with a single interaction term but same could be done with a more complicated system and results would follow similarly.

[1] L. Wang, Discovering phase transitions with unsupervised learning, Physical Review B **94**, 195105 (2016).