

Convolutional Neural Network applied to the 2D Ising Model

Mahlet Molla

November 2023

1 Introduction

When modeling complex dynamics that include many factors that cannot be easily predicted, machine learning is a tool that can help uncover patterns and efficiently achieve accurate results. Within the context of this project, I will be using machine learning to predict the paramagnetic to ferromagnetic phase transition of a material by utilizing the 2D Ising Model. The 2D Ising Model is a well-known mathematical model for describing a ferromagnetic system and I will utilize a convolutional neural network to train and validate on a square lattice and test the model on a triangular lattice system.

2 Ising Model

A ferromagnetic system is defined by its spontaneous magnetization at a critical temperature above which it is paramagnetic. These two regimes are distinct and manifest in differences in physical properties like susceptibility χ , magnetization M , or heat capacity C . For this project, I will only be considering average magnetization to quantify the differences between the low-temperature ordered state where the spins are aligned and a high-temperature disordered state where the spins are not aligned and point at random.

$$\chi(T) = \frac{dm}{dh}|_{h=0} \propto |T_c - T|^{-\gamma} \quad (1)$$

$$m(T) = \langle \frac{|M|}{V} \rangle \propto (T_c - T)^\beta \quad (2)$$

$$C(T) \propto \log|T - T_c| \propto |T - T_c|^{-\alpha} \quad (3)$$

In order to simulate a magnetic system that is undergoing a phase transition, I will be utilizing 2D Ising models that are governed by the following simplified Hamiltonian where J is the coupling constant that determines the strength of the coupling between spins on a lattice while σ_i and σ_j represent the spins at

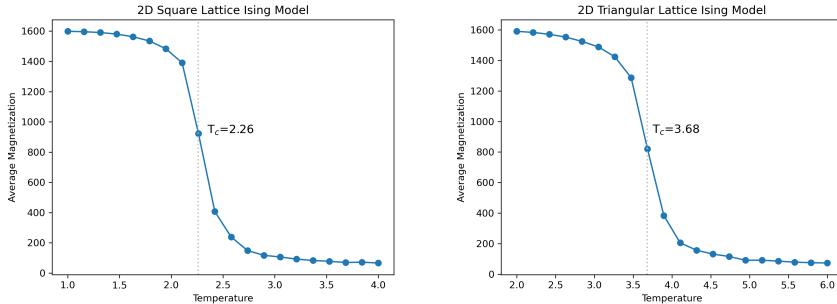


Figure 1: Simulated average magnetization vs temperature

their respective site i and site j . For this project, I chose an $N \times N$ square and triangular lattice.

$$H = -J \sum_{i,j} \sigma_i \sigma_j \quad (4)$$

Figure 1 shows the average magnetization plots of the square and triangular lattice Ising systems where $N=40$ and the critical point clearly occurs at the inflection point of the magnetization curve. For larger system sizes, the average magnetization has a larger spread at higher temperatures which makes it more difficult when determining T_c . Therefore, for finite-size scaling, it is best to determine the critical point from the Binder cumulant U since all the curves of $U(T)$ at different system sizes intersect at T_c .

$$U = 1 - \frac{\langle M^4 \rangle}{3\langle M^2 \rangle^2} \quad (5)$$

3 Monte Carlo Markov Chain (MCMC)

In order to simulate the Ising system to obtain configurations representative of ordered and disordered states, I utilized the Metropolis algorithm and applied a Markov process. Within the context of this model, the algorithm is as follows:

1. Begin by initializing the lattice where the value of each spin σ is randomly chosen to be spin up (+1) or spin down (-1) and on step k , pick a random site on the lattice to consider flipping its spin from σ_i to $-\sigma_i$
2. In order to determine whether it is energetically favorable to flip the spin and move to the next step of the Markov chain, I calculate the change in energy, ΔE that would result from flipping σ_i relative to the thermal

fluctuations operating on the system where:

$$\Delta E = E(-\sigma_i) - E(\sigma_i) \quad (6)$$

$$E(\{\sigma\}) = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j \quad (7)$$

- (a) If $\Delta E \leq 0$, flip the spin
 - (b) If $\Delta E > 0$, then the flipping of the spin is accepted with the probability $\exp(-\beta\Delta E)$, otherwise, the spin-flip is rejected.
3. Do this for each Monte Carlo step and the range of temperatures we're interested in.
 4. Calculate the magnetization for each final configuration by taking the sum of the values of the spins and calculating the average magnetization to determine T_c .

This algorithm is useful for describing the equilibrium configuration of systems that have a probability proportional to the Boltzmann factor. It is assumed that the newest configuration should have a lower energy than the current configuration and if that is not the case, the new configuration will be considered as long as the temperature is high enough. The MCMC methodology allows for an efficient survey of all possible configurations that are also consistent with equilibrium statistical mechanics.

When applying this algorithm to the 2D Ising model, the one thing that differs between the two lattice types examined in this project is the number of neighbors which in turn affects the value of the critical temperature as seen in Figure 1. A site on a square lattice has vertical and horizontal neighbors whereas a triangular lattice also includes diagonal neighbors. The additional neighbors of the triangular lattice increase the complexity of the system which leads to frustration in satisfying the pairwise interactions which means that a triangular lattice transitions at a higher temperature ($T_{c,exact} = 3.641$) than a square lattice ($T_{c,exact} = 2.269$).

4 Deep Neural Networks

Neural networks are a type of machine learning model that can be utilized in applications ranging from object recognition to data mining. It is structured similarly to a neural synapse that receives an input and fires a signal depending on the satisfaction of a threshold. Similarly, an artificial neural net receives an input, processes it via weights. If the weighted sum of the inputs surpasses the specified threshold given by the activation function, it then produces an output signal.

Any neural network can be described by Figure 2 which has an input, hidden, and output layer. The input layer contains a set of features whereas the output is in the form of a probability distribution of the predictions given by the model.

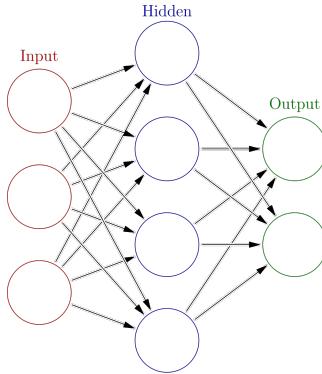


Figure 2: Simplified diagram of a neural network

The amount of as well as the complexity of the hidden layers are some of the defining factors that can differentiate a simple neural network from a deep neural network (DNN). With the deeper architecture of a DNN, the added complexity of the architecture of the networks allows it to learn more intricate and abstract features from the input.

Although the hidden layer of a neural network may sometimes appear to be a black box that can be modified for any problem, all of its functionalities can be described by a weight, a summation with a bias, and an activation function. Once an input is fed into the network, it is given a weight so that the model can prioritize features that contribute more towards training. The weighted inputs are then added together with a bias and are then passed through an activation function that decides whether the input should be activated or not based on the computed sum. The activation function introduces non-linearity which allows the network to model complex relationships present in real-world data. The particular activation function used in a DNN is dependent on the problem but some common activation functions are a Sigmoid function, a Rectified Linear Unit (ReLU), or a Hyperbolic Tangent. For the purposes of this project, I have utilized ReLU as the activation function since it is commonly used in DNN.

4.1 Convolutional Neural Network

Convolutional Neural Networks (CNN) are a subset of neural networks that are designed to process visual data since images can contain a large amount of information that can be hard to learn from. These types of networks can be described by Figure 3 which illustrates that the hidden layers of this model are a convolutional layer, a pooling layer, and a fully-connected layer.

A convolutional layer uses filters that perform convolution operations on the input matrix to extract a specific feature. The filters are moved to each position of the input matrix and an element-wise multiplication is performed and then summed between the input and the filter. The resulting matrix is called a

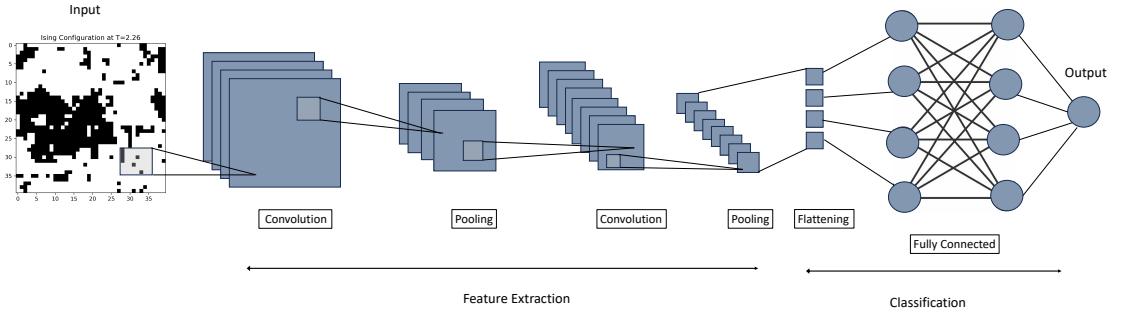


Figure 3: Convolutional Neural Network

feature map and it is then passed down to the pooling layer. The pooling layer's function is to compress the spatial size of the input while retaining the important information gathered by the previous layer. One way in which the pooling layer accomplishes this is using max pooling where the maximum element in a specified region of the feature map is selected. An alternative way to pool the sample is to use average pooling where the average element is selected. The feature extraction portion of the CNN is concluded with the application of the activation function. The feature map is then flattened for the fully connected layer that performs classification and connects each neuron so that the network can learn high-level representations.

Within the context of this project, the CNN takes the square lattice Ising configurations across 20 temperatures that are labeled as ordered or disordered depending on the value of the critical temperature. Of these configurations, 80% are used to train the model and the rest of the configurations are unseen samples that validate the model. The trained and validated model is then tested using the configurations of the triangular lattice.

5 Results

5.1 MCMC Scaling Study

As mentioned in Section 2, the analytical solution to the Ising model is for an infinite system which means the configurations found using MCMC deal with finite-size effects. The finite-size effects can be remedied by increasing the lattice size though that also increases the computational cost of the system. In Figure 4, I have performed a scaling study by using the absolute mean configuration across all the temperatures for $N = 4, 10, 20$, and 40 . The mean configuration is expected to have a similar relation to temperature as the average magnetization. This means that at the ordered state, the spins are aligned and the disordered

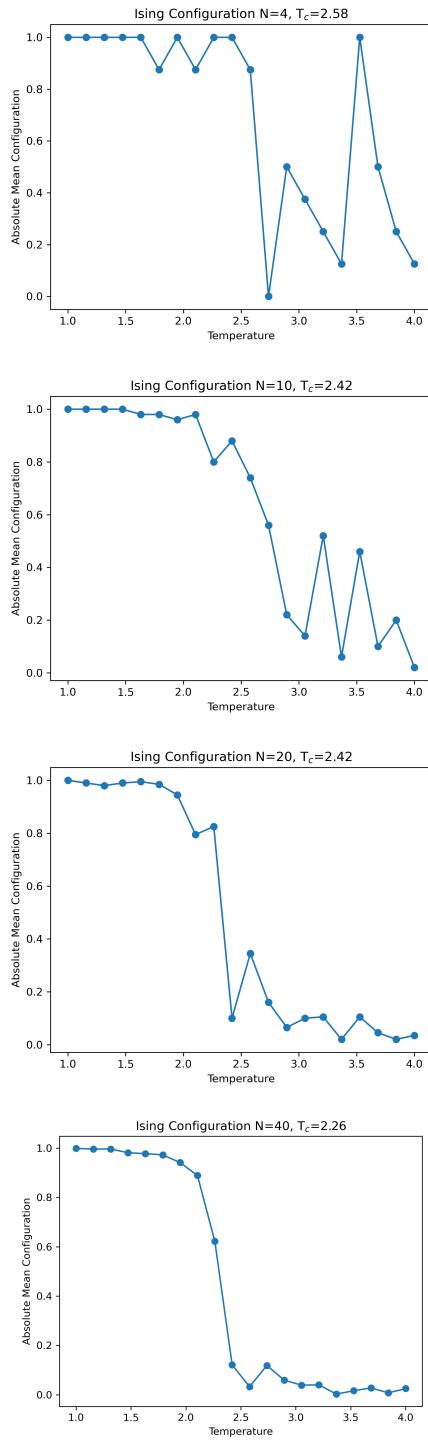


Figure 4: Finite-size scaling of MCMC
6

Learning Rate	Adam		Stochastic Gradient Descent	
	Average Validation Loss	Test Loss	Average Validation Loss	Test Loss
5.00E-01	0.55	0.703	0.696	0.674
5.00E-02	0.55	0.677	0.006	0.134
5.00E-03	0.004	0.231	0.037	0.071
5.00E-04	0.0038	0.125	0.2712	0.344

Table 1: Comparison of Learning rate of Adam and SGD Optimization and Cross-Entropic Loss

regime has a random alignment of the spins meaning the values of the different spins cancel out. This relationship is most clear at the $N=40$ and as the lattice size decreases, there is less of a distinction between the ferromagnetic and paramagnetic states due to increasing finite-size effects.

5.2 CNN Accuracy and Loss

The machine learning model used in this project outputs a binary value to indicate the predicted phase of a specified configuration. The accuracy of the model is then calculated to be a percentage of the correct number of predictions relative to the total number of predictions. Another way that I evaluated the performance is by calculating the loss of the samples going through training, validation, and testing. When using the analogy between neurons and artificial neural networks, the activation function has a threshold that decides whether to produce an output signal. This threshold is partly defined by the loss where accurate predictions are essentially assigned high probabilities and incorrect predictions have low probabilities. The loss used in this project is cross-entropy loss, H which is commonly used in neural networks that deal with classification, and its value increases as the predicted probability varies from the correct label. It is defined as follows where $p(x)$ is the correct probability distribution of the classes and $q(x)$ is the predicted probability distribution of the CNN.

$$H(p, q) = - \sum_{x \in \text{classes}} p(x) \log q(x) \quad (8)$$

Rate	Adam		Stochastic Gradient Descent	
	Average Validation Accuracy (%)	Test Accuracy (%)	Average Validation Accuracy (%)	Test Accuracy (%)
5.00E-01	55	60	69.6	60
5.00E-02	55	60	99.77	96.85
5.00E-03	99.8325	96.32	99.78	96.94
5.00E-04	99.85	97.34	89.1	81.65

Table 2: Comparison of Learning rate of Adam and SGD Optimization and accuracy of prediction

As a neural network is learning the features of the various classes, I utilize an optimization algorithm that is trying to find the minimum of the cross-entropy loss function. Additionally, the loss is very dependent on the learning rate which is essentially the step size at each iteration of the optimization scheme, and it

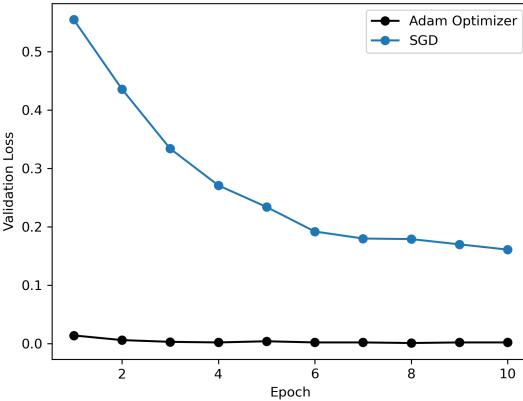


Figure 5: Adam Optimization vs SGD Optimization across epochs

is able to be tuned to find the true probability distribution. In Table 1 as well as Table 2, I compare the performance of the CNN with varying learning rates and optimizers. The ideal learning rate is a trade between convergence speed and the ability to find the global minima of the loss function. The Adams optimizer seems to be more accurate and has a lower loss value as the learning rate decreases whereas with the Stochastic Gradient Descent (SGD) optimization, the ideal learning rate is around 5×10^{-2} and 5×10^{-3} . For SGD, if the learning rate is too high, it most likely leads to overshooting the minimum, and too low of a learning rate results in getting stuck in a local minima 5×10^{-4} . Figure 5 is a plot of the loss function of the two optimizers at each epoch with a learning rate of 5×10^{-4} . An epoch is defined as the number of complete passes through the data set. From Figure 5, it is clear that the loss function is constant across all epochs with the Adams optimization whereas the SGD algorithm is getting closer to the true probability distribution though it plateaus at a minima that is higher than the one found by the Adams optimization.

6 Conclusion

Simulating various configurations of the 2D Ising model using MCMC and utilizing these configurations as inputs in a CNN to predict the phases of a particular Ising configuration sheds light on the significant interplay between machine learning and statistical physics. Additionally, this project has illustrated how although finite-size effects can play a role in getting physically accurate configurations, a lattice size of $N=40$ is sufficient to train a CNN. Additionally, when training the inputs of the neural network, the loss function which quantifies how closely the output predicts the correct phase, is greatly dependent on the learning rate and the learning rate has varying behavior depending on the

optimization algorithm tool used. This project’s cross-disciplinary exploration shows the power of neural networks in modeling physical systems which has led to a deeper understanding in both domains.