**The Ising Model**

The Ising Model is a mathematical model used in statistical mechanics to describe ferromagnetism behavior, and it's used for a great variety of systems, from spin behavior in a lattice, to neurological connections or biological growth. We will be focusing on how the model is used in physics. The Ising Model is defined on a grid (lattice) where each point (site) represents an atomic spin. The most common lattices are 1D (a line of sites), 2D (a square grid), and 3D lattices. Each site can have two possible states: Spin up (+1) or spin down (-1), and they represent the magnetic moment of atoms in a material. In addition, these spins interact with each other. This leads to the following Hamiltonian in 2D:

  
**Equation 1: Hamiltonian of the Ising Model**

Where J is the interaction strength between neighboring spins, s is a spin configuration in site (i,j), and h is the external magnetic field. This means that the energy of the system is described by the sum of interactions between neighboring spins, and the sum of interactions between an individual spin site with an external magnetic field.

The Ising Model exhibits a phase transition from a disordered to an ordered state based on an occurrence probability which changes with temperature. At high temperatures, the thermal energy dominates, and the spins are randomly oriented. This results in zero net magnetization. On the other hand, low temperatures allow for a phase transition, so spins align, and a net magnetization occurs. The boundary which determines high and low temperatures is called the critical temperature, and it depends on the dimension of the system and the interaction strength J.

**Deep Learning**

Deep learning is a subfield of machine learning, and it’s inspired by the structure and function of the human brain, particularly how neurons interact in layers. The key idea is to use hierarchical representations of data, where each layer in a neural network extracts increasingly complex features from the input. It has driven much of the recent advancements in artificial intelligence (AI) and is widely used in various applications, including image and speech recognition, natural language processing, and autonomous systems.

The layers in the neural network are composed of units called neurons. They take inputs, apply weight to them, and pass the results through an activation function to produce an output. The weight of the input determines the strength of the connections between the neurons and how they’re connected. On the other hand, the activation function introduces non-linearity and allows for learning complex patterns. Neural networks, including the ones used in deep learning, are composed of layers of neurons and are divided into three categories:

* Input layer: The first layer, which receives the raw data.
* Hidden layers: The middle layers. They do not directly interact with the input or output, but it’s where the extraction and transformation of data occurs.
* Output layer: The last layer, which produces the network’s predictions or classifications.

One of the primary uses of deep learning to the Ising Model is to train neural networks to classify spin configurations into different phases (e.g., paramagnetic vs. ferromagnetic). We teach the model by using autoencoders, which consist of an encoder and a decoder. The encoder compresses the input into a lower dimensional space, and the decoder reconstructs the input to its original shape. We use autoencoders so the model can learn to represent spin configurations in a compressed form and thus capture relevant physical features for phase transitions like order parameters. By analyzing the latent space, researchers can gain insights into the key features that the neural network considers important for distinguishing between different phases. This can help identify new order parameters or refine existing ones.

**Article: The critical temperature of the 2D-Ising model through deep learning autoencoders**

Introduction

The 2D Ising model's critical temperature has been previously determined through analytical methods (Onsager’s exact solution) and numerical simulations. The goal of the study is to apply deep learning techniques, specifically autoencoders, to estimate the critical temperature of the 2D Ising model. The motivation for using autoencoders stems from their ability to capture significant features of data, potentially identifying the phase transition point in the Ising model without explicit feature engineering.

Swendsen-Wang Algorithm

The algorithm used to replicate data that will be fed to the autoencoder is called the Swendsen-Wang algorithm. In this algorithm, we start with a lattice of spins. For each pair of neighboring spins with the same spin value, the probability of creating a bond is calculated using the following probability function:



**Equation 2: Bond Probability Function**

Where kB is the Boltzmann constant, J is the self interaction between neighboring spins, and T is the temperature of the system. We then compare this probability with a random probability (Generating a number between 0 and 1) and if our bond probability is higher, a bond is formed. We do this for all pairs to end up with a lattice matrix with clusters of spins, with each cluster being defined as all the spins which are connected via bonds. Finally, we do a global update by flipping all spin clusters with a probability of 0.5.

Depth-First Search Approach

The DFS approach is an algorithm used to understand tree like structures of data. It explores each branch of the data tree by starting at a data point, then using a recursive approach in the exploration of all neighboring spins. If we in addition mark each visited site of the branch as a cluster, we can successfully classify sets of data based on their connections. For more info, visit the following link: <https://www.youtube.com/watch?v=7fujbpJ0LB4>.

**Code**

Ising Model Code

In this work the author chose to use the Swendsen-Wang Algorithm with zero external magnetic field, and for the sake of simplicity, defined J = +-1 and kB = 1. We define a sequence of temperature values surrounding the critical temperature, and let the code determine at which value the spins become disordered.

We first define a class called **IsingModel**, we define L as the size of the 2D lattice, T as the temperature, and p as the function probability. The \_init\_ function defines an LxL grid with values of -1 or 1, representing the initial spin configuration. Then, the swendsen\_wang\_step function defines another grid filled with zeros in which the cluster labels will be added. Clusters need to be labelled in order to identify them when the new spins are defined at each temperature. This same function performs a depth first search approach to cluster and label connected spins. First, we identify if there’s already a label in the site of interest. If there isn’t, we create a stack and add a label, then look for neighbouring aligned spins. If the neighbouring spins have no label, have the same spin alignment, and have a higher bond probability than our random.random() probability, we add the same labelled as before and add them to the stack. After this is done to all the sites, the initial cluster is updated by flipping all the spins with probability 0.5 (Again, compared against our random.random() probability), and we obtain a grid with the updated spins based on the temperature probability. Finally, the magnetization function takes the mean of all these spin values, thus outputting the magnetization of the system.

After the IsingModel class, we have the **generate\_ising\_data** function, which uses the IsingModel with an iteration input to repeat the Ising Model several times and obtain several magnetization values per iteration, at a give value T, L and p. In our case, we used the probability function mention under ‘The Ising Model’ section for p, T = [0.269185, 0.769185, 1.269185, 1.769185, 2.269185, 2.769185, 3.269185, 3.769185, 4.269185], and L = 20 under 100 iterations. We defined **plot\_magnetization\_sw** to plot magnetization values for each temperature and got the following results:

A group of blue lines

Description automatically generated

A group of blue lines

Description automatically generated

A group of blue lines

Description automatically generated  
**Figure 3:** Magnetization values and spin Counts under the Ising Model for several temperature values near the critical temperature.

This shows the disordering of the spins near the critical temperature as expected!

**Autoencoder Code**

For the autoencoder, we define a similar iteration function for the magnetization called **create\_dataset**, but this time the values are appended per each temperature input as opposed to each iteration. Then, we define our **Autoencoder** function so that we have eight layers, an encoder with the input, first, second, and third layers having 625, 256, 64, and 1 neurons, a decoder that has the first, second, and third layers use 64, 256, and 625 neurons, an output layer with a number or neurons equal to the number of lattice points, an activation function of ReLu for the input, layers 1-2 in the encoder, and all hidden layers of the decoder, and a tanh activation function for the third encoder layer and the output layer. We use the dropout realization method in which 33% of the neurons are temporarily deactivated at each layer and used for testing, and the other 66% is used for training. For training, we defined the **train\_autoencoder** function, in which we used a starting learning rate of 0.001 and was reduced by 20% when learning stagnates for 30 epochs, with minimum learning rate of 0.000001. The implementation was performed using Pytorch. Finally, the **main** function implements the data from the ising model into the autoencoder, with 2000 iterations.

Our results are the following:

A screenshot of a computer

Description automatically generated

This can ultimately be used to train the autoencoder to determine critical temperatures for different physical systems, which might lead to very interesting discoveries.