Alejandro J. Rigau López

May 20, 2019

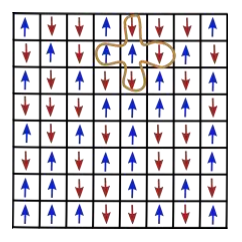
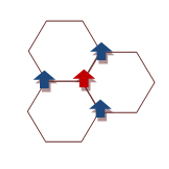
FISI4999

Rafael Ramos

Final Report: Identifying Phase Transitions using Neural Networks

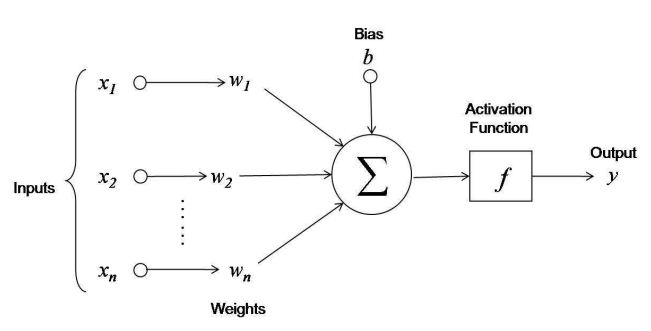
During FISI4999 this semester, I discovered many ways to apply my programming knowledge into the field of machine learning and physics. With the help of Jairo Orozco, I created a program that could accurately predict phase transitions using data generated from Monte Carlo simulations of the Ising model. Classifying and discovering phases and phase transitions is one of the most important topics in Condensed Matter Physics; however, it is not an easy job to do. Because of this, we use Machine Learning algorithms to try to predict these critical temperatures where the phase transition occurs.

To study these phase transitions, we use the Ising model. The Ising model is a simplistic model of a magnet, represented with a lattice structure where each site contains a “spin”, an arrow that points up or down. We use the Monte Carlo method to generate data that corresponds to the Ising model. The Monte Carlo method combined with machine learning algorithms has been widely used in the past, but most research was conducted using square lattice structure. In our research, we experiment using a hexagonal lattice structure, and compare it with previously studied square lattice.

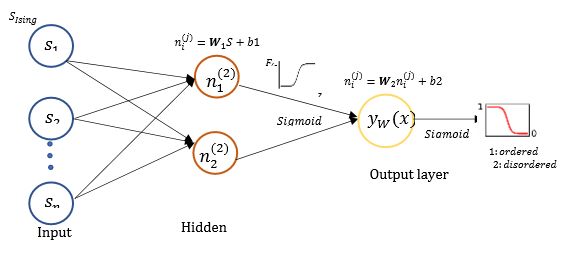


**Figure 1.** Hexagonal lattice vs. square lattice.

To predict the phase transition, I used the Artificial Neural Network. The Artificial Neural Network, or ANN for short, is a machine learning algorithm that closely resembles how the neurons in our brain work. This algorithm is “trained” to find the ideal parameters that best represent the input data. These parameters are called weights (*W*) and bias (*b*). Many artificial neurons are stacked in layers to create the network. It is important to note that during our experimentation we used two different structures. One was the complex network who had 300 neurons in the hidden layer, and the other was the simple network who only had two neurons in the hidden layer. By having only two neurons, it made it easier to analyze network parameters and identify patters, while the more complex network was used to accurately predict the critical temperature of the system.

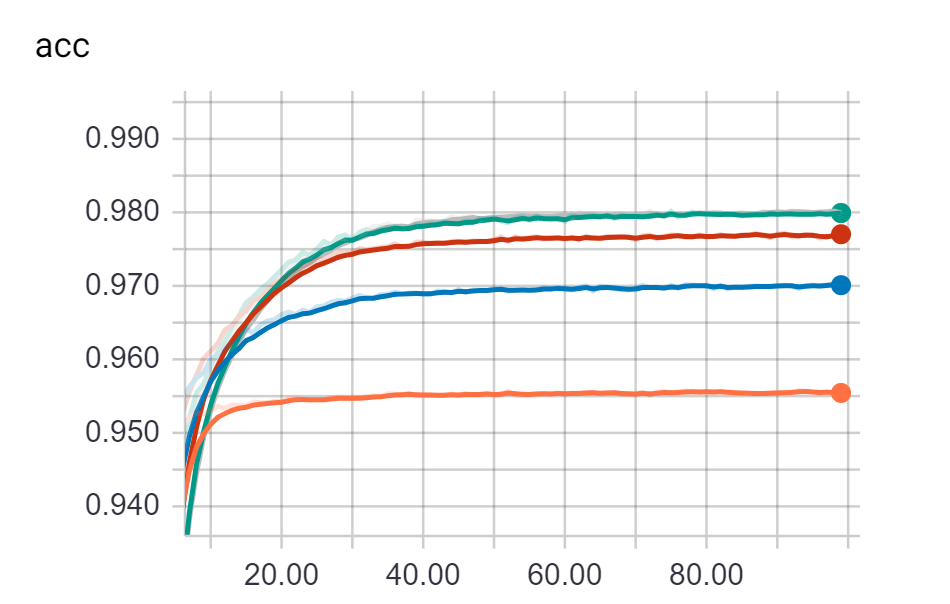


**Figure 2.** Simple neuron representation.

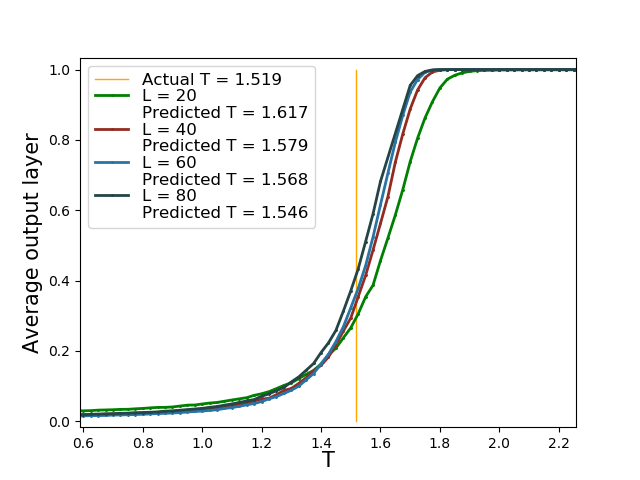


**Figure 3.** My Artificial Neural Network structure.

Using a python program created by Jairo, we would generate long sequences of “spin” configurations of a system. The spin was represented with a one for an up direction or a zero for a down direction. This was done with lattice sizes of 20, 40, 60, and 80, but most experimentation was done with lattice size 20 due to its reduced complexity. These configurations would then be the input of the network and, since and order parameter distinguishes two different phases, the output of our network could be considered our order parameter. Based on the error of this output, the algorithm would determine how much should the network parameters be changed. These calculations were done using the Cross-Entropy loss function and the Adam optimizer. L2 loss was also applied to avoid overfitting to the data with a 𝜆 = 0.001. A total of 196200 configurations were used in the training set, 65400 in the validation set and 65400 in the testing set. The training set is the big bulk of data used to optimize our network parameters and fit our data accurately, the validation set was used to test the network during the training phase and the testing set was used to test the network after the training phase was over. 147000 of these configurations are below critical temperature and 180000 are above it. Configurations below the critical temperature were labeled 0 and the ones above the critical temperature were labeled 1. Using a 1080Ti GPU, we trained the 300-neuron network on 100 epochs, resulting in above 95% accuracy with every lattice structure. Accuracy would increase with the size of lattice.



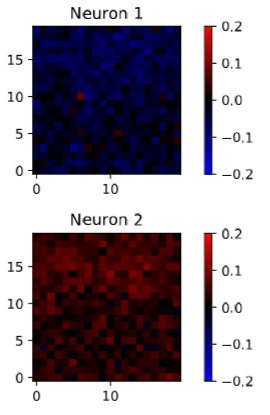
**Figure 4.** As the lattice size was increased, more accuracy was achieved. Each color represents a different lattice.



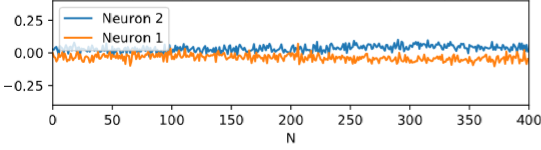
**Figure 5.** Prediction of Critical Temperature with different lattice size.

From Figure 3 and Figure 4 we can observe how the lattice size would affect the accuracy of our predictions. The actual critical temperature is 1.519 and my network predicted a 1.546 using the lattice size of L = 80 and the 300-neuron network. It is important to note that it was possible to obtain near perfect results by increasing the number of neurons in the hidden layer. We later decided to keep it at 300 neurons because we felt that a larger network was too complicated to analyze.

Using our smaller network and a lattice size of L = 20, we were able to analyze all network parameters, letting us observe some curious behaviors. It was clear that the parameters from one neuron would favor positive numbers and the other neuron would favor negative numbers.

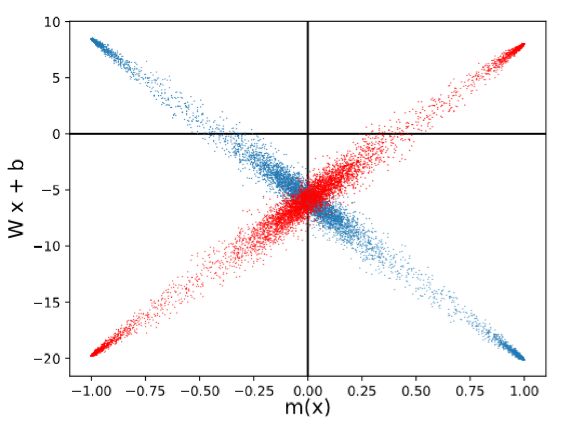
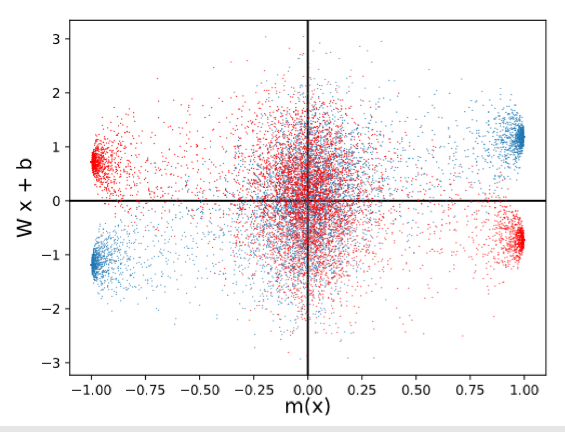


**Figure 6.** Weight values in each neuron. Red represents positive values and blue represents negative values.



**Figure 7.** Weight value vs neurons.

This information gave us hints that the order parameter is encoded in the network. We also plotted the argument 𝑊 𝑥 + 𝑏 of the hidden layer as a function of magnetization 𝑚(𝑥) of the configurations 𝑥.

  
**Figure 8.** Hidden layer arguments as a function of magnetization 𝑚(𝑥) before and after training.

The figure shows that the magnetization is encoded inside the hidden layer of the network. This is corroborated with Carrasquilla and Melko’s research when they used three neurons in the hidden layer. These results also indicate that a small 2-neuron network can encode the magnetization, order parameter, and can also predict the critical temperature of the system. It is important to note that this network had no prior knowledge of the critical temperature, magnetization or order parameters. The only data that the network has is the data generated by the Monte Carlo simulations.

To create our neural network, I used a python programming library developed by Google called TensorFlow. TensorFlow is widely used by researchers to simplify the development, prototyping and deployment of these machine learning algorithms. Most of these graphs and visualizations were all directly programmed by me with the help of python plotting libraries. All my work contained more than 775 lines of code and many versions of these models. A thorough optimization and documentation process of the code was also conducted in order to publish my code with our findings. My code can be found here: <https://github.com/AlejandroRigauLopez/My-Ising>

During the semester, I managed to learn a lot about neural networks, how they are built and the math behind them. With the help of Jairo, I was able to take my programming knowledge and use it to do some useful physics research. I went from zero knowledge, to possibly publishing a research paper soon. Together we were able to demonstrate how machine learning can be a powerful tool in phase transition detection and in physics in general.