❦ Learning from data - Project 2 ❦

kevinan & simjac

September 2019

Abstract

The Ising model is a model in statistical physics which describes ferromagnetism in lattices of spins. In this model, it is provable that, for two dimensions and higher, there is a clear phase transition. This phase transition happens at a critical temperature, T_c. Advances in machine learning have made it possible to analyse the Ising model and the phase transition with neural networks. This report aims to perform such an analysis with several different kinds of networks, a single neuron network, a dense single hidden layer network, and a convolutional network. A three neuron network is also described, motivated physically, and implemented. Lastly, a Bayesian neuron is implemented and evaluated. The validation accuracy for the first three networks is around 0.95 and the predicted critical temperature is around 2.30 [t.u.]. The results from the Bayesian neuron are consistent with the other networks.

1 Introduction

The applications of machine learning algorithms and neural networks to physics is of great interest. One such application is the famous Ising model. Below follows a background for both neural networks and the Ising model.

1.1 Background of Ising model

One of the most famous models in physics is the Ising model, which describes atomic spin configurations. The model can be used to determine phase transitions between a low temperature ordered phase (paramagnetic) and a high temperature unordered phase (ferromagnetic) [1]. The model was first proposed in 1920 by Wilhelm Lenz. In 1924, it was proved by Ernst Ising, who the model is named after, that there does not exist a phase transition in one dimension. It was later proved, that for any dimensions greater then one, there exists a phase transition. In 1944, Lars Onsager found that, in the two dimensional case, the phase transition happens at the critical temperature

$$T_c = \frac{2}{\log(1+\sqrt{2})}[\text{t.u.}],$$

where the temperature unit, [t.u.], is such that the Boltzmann constant k_B and J from the upcoming equation, (1), are set to 1.

In two dimension, with no external magnetic field, the Ising model can be described by the Hamiltonian

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

where $s = (-1, 1)$ $[\frac{1}{2}\hbar]$. From this, the energy e of a lattice of spins can be calculated by just summing over the lattice. The magnetization, m, of the lattice can also be calculated and is the average, $\langle s_i \rangle$, of the spins. The connection between the phase shift of the lattice and these two quantities has long been studied and one promising way of investigating is by machine learning algorithms.

1.2 Background of neural networks

In the last decade, great advances in computer power and algorithms have opened the door for the usage of new advanced computation methods. One such method is a artificial neural network. Neural networks started to develop in the 1950's [2] and were inspired by the human brain. They are systems of neurons that for a given signal either fires or not. The usage of neural networks has increased tremendously in the last years and their have proven to be especially useful in classification problems.

Today, there exists a plethora of classes of neural networks. We will, in this report, touch upon some of the most popular and simple.

The most basic network is composed of one neuron, which takes a number of inputs, passes them through an activation function, and then gives an output. A generalization of this is to create a fully connected hidden layer with many neurons that all take input, passes it through a activation, and then passes it to a output layer. A *convolutional neural network* (CNN) is a network whose purpose is to use spacial structures in the input by using convolutional operations. CNN's are for example widely used in image recognition. These are example of well established and useful neural networks, an example of a progressing class of neural networks is a *Bayesian neural network* (BNN). A Bayesian network takes into account that not only the set of weights that minimizes the cost function, can be used in the network, but rather there exist a *probability density function* (pdf) over the weights. From this weight distribution the network would also give a pdf for the output of the network. The strength in a Bayesian network is that it can tell you how sure it is of its prediction. The weakness is that it also takes a lot more computer power when the parameter space gets large.

1.3 Purpose

This study intends to investigate if neural networks can be used to describe phase transitions and predict the critical temperature of a lattice. Several different neural networks are studied, one with a single neuron (SN), one with a single hidden dense layer (DNN), and one convolutional neural network (CNN). These networks are implemented, discussed, and compared to each other. The study then aims to understand the inner workings of a simple neural network with one dense layer composed of three neurons using sigmoid

activation. Lastly, a Bayesian neural network (BNN) is implemented and evaluated for a network with a single neuron.

2 Method

The methods used in this study are presented in the following section. Firstly, how the data was generated. Then, how the single neuron, fully connected, and convolutional networks were implemented. Then, also how a network simple three neuron network were used to confirm the physical intuition about the Ising model. Lastly, the details of the Bayesian neuron are presented.

2.1 Generation of lattices

The generation of spin lattices was done with a Monte Carlo simulation. First, a lattices with size 10×10 is initialized with random spins configurations. This lattice is then stepped forward 500 times, which simulate thermalization. Each step consist of 100 of randomly proposed changes in the lattice. For each proposed change, the difference in energy, E, is calculated using 1 and the proposed lattice is accepted with the probability

$$p = \min(1, \mathrm{e}^{\frac{\Delta E}{k_\mathrm{b} T}}).$$

This was then done 1200 times for different temperature.

2.2 Single neuron classifier with energy and magnetization as input

The foundation of all networks can be described by three things; the architecture, activation rule, and the learning rule. For the SN, the architecture consisted of mapping the input data, energy x_e and magnetization x_m of the lattices, to an output probability y of the lattice being in the unordered phase. The activation function used was the *sigmoid function*

$$Z = \frac{1}{1 + e^{-a}}, \tag{2}$$

where the *activation* a was given by

$$a = w_0 + w_1 x_e + w_2 x_m.$$

To update the weights, we used the sum of all cross entropy and an l_2-regularizer as cost function:

$$C(w) = \sum_{i=1}^{n} \Big[t_i \log\big(y(x_i; w)\big) + (1 - t_i) \log\big(1 - y(x_i; w)\big) \Big] + \frac{1}{2} \alpha \sum_{i=1}^{l} w_i^2, \tag{3}$$

where t_i is the target (in this case the true phase 0 or 1), n is the number of training data, and l is the length of the weight vector. With this cost function, the learning rule becomes fairly simple:

$$\begin{aligned} \nabla_w C &= x(\vec{t} - \vec{y}) + \alpha \vec{w}, \\ \mathrm{d}w &= -\eta \nabla_w C(w). \end{aligned} \tag{4}$$

When training the SN, the lattices were split into 70 % training data and 30 % validation data. In the learning rule (4), two hyperparameters are introduced: the step size η was set to 10^{-2} and the regularization parameter α was set to 1.

2.3 Fully connected hidden layer with spin lattice as input

The architecture for the DNN consisted of an input layer with the flattened spin lattice (of size 100), one hidden layer with 50 fully connected neurons, and an output layer with one neuron which outputted the probability of the lattice being in the unordered phase. The activation function *rectified linear unit* (RELU) that was used in the hidden layer is

$$Z = \max(0, a), \tag{5}$$

and a *sigmoid* function was used in the output layer.

The learning rule that updated the weights of this network is specified by the Adam optimization algorithm on an *binary cross-entropy* cost function with an l_2-regularizer. The regularizer parameter, α, was set to 10^{-2}.

The DNN was implemented in python by using TensorFlow Keras. In training this network, the data was also split into 70 % training and 30 % validation data. The training was done for 50 epochs with batch size equal to the training size and $\alpha = 10^{-4}$.

The DNN was trained predict the phase, given the configuration of the spin lattice. To calculate the critical temperature, the validation lattices was passed through the network and the probability of being in the unordered phase was plotted against the temperature. A sigmoid function was fitted to this plot and the critical temperature was extracted as the temperature where the sigmoid function was precisely 0.5, which is the point that indicate where the network has no idea if the lattice describes an ordered or unordered phase.

2.4 Convolutional neural network with spin lattice as input

For the CNN, the input was, as for the DNN, the full spin lattice. The architecture and activation rules of the CNN are displayed in Table 1. The cost function used was, again, (3) with optimization Adam. The CNN is a space invariant network that take advantage of spatial structures in the lattices. Since we know that the energy of the lattice is related to what spins the nearest neighbours has, it is motivated to have as a first layer a convolutional layer with window size of 3×3.

Table 1: Specifications for the CNN. The optimization was performed using optimizer Adam, loss categorical crossentropy, and metric accuracy.

	input	convolutional layer	max pooling layer	convolutional layer	max pooling layer	dense layer	output
size	10×10	$3 \times 3 \times 10$	2×2	$3 \times 3 \times 10$	2×2	10	2
activation		RELU		RELU		RELU	sigmoid
regularization		10^{-2}		10^{-2}		10^{-2}	

As for the DNN, the CNN was implemented in python by using TensorFlow Keras. While the training and test data were chosen randomly for the DNN, for the CNN, the training data consisted of all lattices with temperature under 1.3 [t.u.] and over 4.7 [t.u.], the test data consisting of the remaining data. The training was done for 200 epochs with batch size 100. The critical temperature was extracted in the same way as for the DNN, by sending all test data through the network and then fitting a sigmoid function.

2.5 Understanding the neural network

Following the first section of the paper "Machine learning phases of matter" [3], we also implemented two fully connected single hidden layer networks with sigmoid activation, one with three neurons in its hidden layer and one with ten neurons in its hidden layer. We did this to show that the neurons generally fall into one of three categories, detecting polarized up configurations, detecting polarized down configurations, and detecting if a configuration is polarized or unpolarized. With the ten neuron network, we wanted to show that if there are more than three neurons in such a network, there is some redundancy.

The model described above can be expressed as

$$Wx + b = \frac{1}{1+\epsilon} \begin{bmatrix} m(x) - \epsilon \\ -m(x) - \epsilon \\ m(x) + \epsilon \end{bmatrix}, \tag{6}$$

which depends on only one parameter, $\epsilon \in [0, 1]$. The first vector entry, $m(x) - \epsilon$, activates when the lattice is polarized up ($m(x) > \epsilon$). The second entry, $-m(x) - \epsilon$, activates when the lattice is polarized down ($m(x) < -\epsilon$). The third entry, $m(x) + \epsilon$, activates when the lattice is polarized up or unpolarized ($m(x) < -\epsilon$). Morally, what the network does is to check whether the lattice is polarized up, down, or unpolarized.

The role of ϵ in this model is to say how much large the lattice magnetization has to be in order for it to be considered polarized.

2.6 Bayesian neuron

The last part of this study was to present a BNN made of a single neuron which takes energy and magnetization of a lattice and gives a pdf of how sure the network is that the lattice is in the ordered or unordered phase.

As always when doing a Bayesian analysis, Bayes theorem was used:

$$p(w|D, I) \propto p(D|w, I)p(w|I). \tag{7}$$

The likelihood was chosen as

$$\mathrm{e}^{-C},$$

where C is as in (3) with $\alpha = 0$. A Gaussian prior with standard deviation 1 was set on the weights. The sampling of this 3-dimensional posterior was made by a Markov chain Monte Carlo sampling with 10 walkers, warm period 100, and 10 000 samples per walker. The prediction of the BNN was then made by calculating one output for every set of weights from the 100 000 sampled weights. The distribution of these 100 000 predicted values was then used as the pdf for the BNN's prediction.

3 Results & discussion

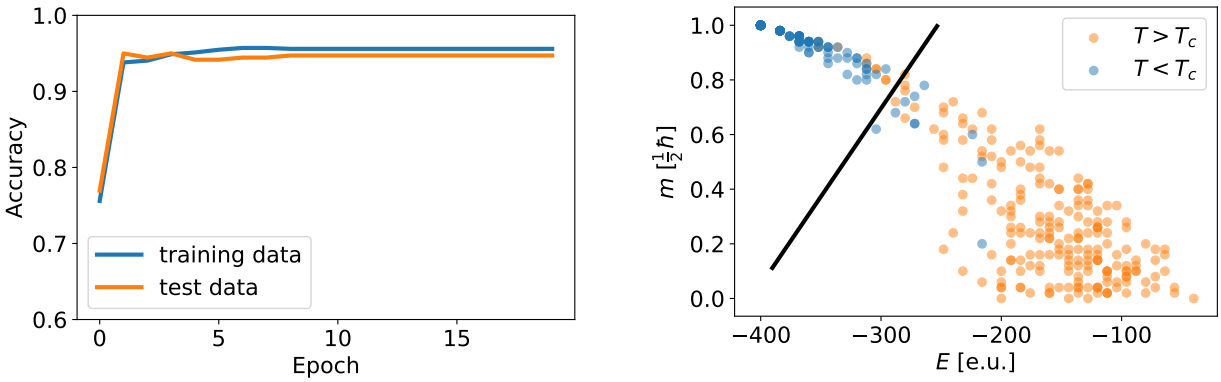
3.1 Describing the phase with neural networks

The summarized results for the first three neural network approaches to describe the Ising problem are presented in Table 2. There, the different accuracies on the training and test data are shown along with the prediction of the critical temperature. The accuracy on the test data is very similar for all three networks: around 0.95. On the training data, the accuracy for the CNN is higher than the on test data accuracy, but for the SN and DNN, the training data accuracy is approximately the same as for the test data. This could indicate that the CNN is overfitting.

Table 2: Accuracies for different networks presented for both their training and testing data. Compare the predicted T_c to the true value 2.2692 [t.u.]

	accuracy on training data	accuracy on test data	predicted T_c [t.u.]
single neuron	0.9560	0.9472	
DNN	0.9681	0.9522	2.3095
CNN	1.0000	0.9422	2.2818

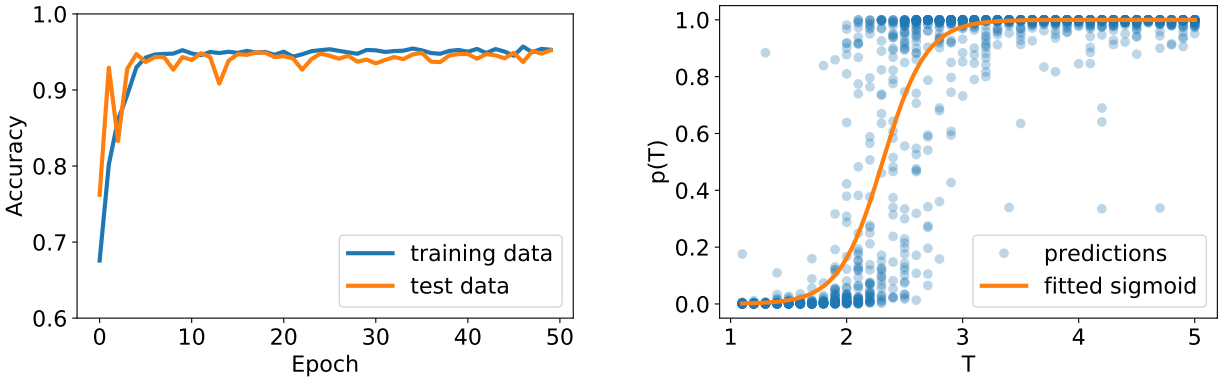
To further study the SN's performance, Figure 1 is presented, which shows how the training and test accuracy changes through the epochs, along with the prediction boundary for the test data. From Figure 1a, the accuracy seems to converge at around 0.95 already after 2 epochs, which indicates that the network converges fast. This isn't strange since one epoch trains the network with 840 data points. In Figure 1b, the the lattices in ordered phase and unordered phase are to a good approximation separated by the decision boundary. It is also clear that some of the lattices gets wrongly classified by the network, that is they are at the wrong side of the decision boundary. It is easy to see that this points seems to be outliers, they are in a areas where most of the data is in the other phase. The overlap could appear due to the method of calculating the energy or magnetisation, or if the connection to the phase is more complicated. Most likely is that this point doesn't fully represent the connection of magnetization, energy, and phase.



(a) (b)

Figure 1: In (a) the evolution over the epochs of the training and validation accuracy is shown. In (b) energy and magnetization data points of all the lattices in the test set is presented with a color representing which phase the lattice is in. In (b) the decision bounder from the single neuron is also presented.

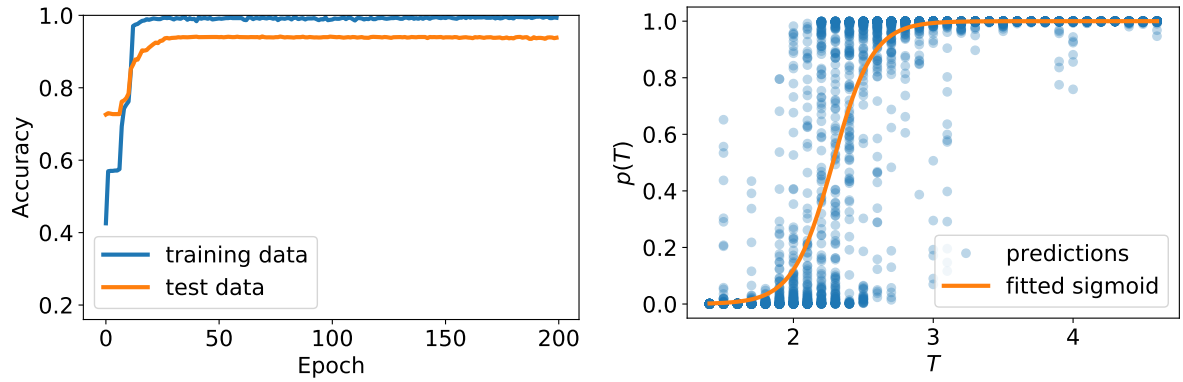
To further study the performance and the prediction of the critical temperature of the DNN, Figure 2 is presented. Figure 2a displays how the accuracy on the training and test data evolves through the epochs. Both the training and validation accuracy converges at about 0.95 fairly fast. In Figure 2b, a sigmoid function has benn fitted to the DNN's output from the test data. The DNN is fairly certain of the classification for low and high temperature, with more variations close to the critical temperature. From the fit, the critical temperature T_c (the sigmoid is equal to 0.5) 2.3095 [t.u.] was extracted. The correct value is 2.269 [t.u.]. The spread of the predictions for lattices with temperature close to T_c indicates a rather large uncertainty in the critical temperature.



(a) (b)

Figure 2: In (a) the evolution over the epochs of the training and validation accuracy, by the network with one fully connected hidden layer, is shown. (b) shows the predicted probabilities of being in the high temperature phase on the test data by the network. In (b) a sigmoid function fitted to the prediction is also presented.

The performance of the CNN is presented in Figure 3. The accuracy of this network, seen in Figure 3a, is rather low for about 10 epochs but then finds a set of weights which increases the training and test accuracy to approximately 1 and 0.94 respectively. The CNN then seems to have converged, with some overfitting present since the test accuracy is lower than the training accuracy. We tried to reduce the overfitting by increasing the regularizers but the network was very sensitive to these changes and the performance got worse. The predicted probabilities of being in the unordered phase from the test data is, together with a fitted sigmoid function, presented in Figure 3b. The prediction seems to be stable for high and low temperatures but has a rather large variance for the predictions close to T_c. From the sigmoid function, $T_c = 2.2818$ [t.u.] was extracted, but as for the DNN, the uncertainty of the sigmoid fit is rather large.



(a) (b)

Figure 3: In (a) the evolution over the epochs of the training and validation accuracy, by the convolutional neural network, is shown. (b) shows the predicted probabilities of being in the high temperature phase on the test data by the network. In (b) a sigmoid function fitted to the prediction is also presented.

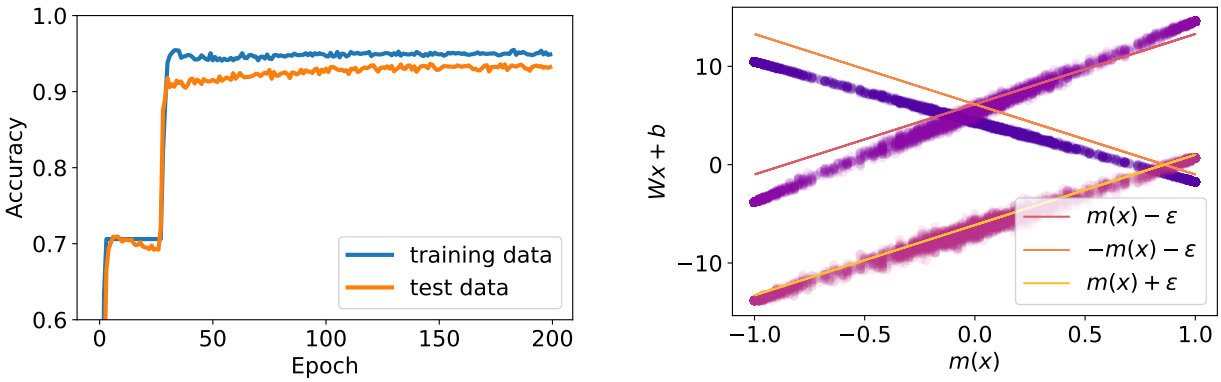
It is interesting to compare the CNN and DNN since they have the same input. From Table 2, we can see that they are very comparable in predicting the phases of the training data. We expected the CNN to perform better since it considers the lattice's spatial structure but we stress that the difference between the networks is minimal. By comparing Figure 2b with Figure 3b, we see that the spread of the predictions is a bit greater for the CNN. This is probably due to the CNN being trained only on low and high temperature data, so when predicting phases near T_c, it is extrapolating. The DNN trains on a random subset of the data set, so it does not need to extrapolate near T_c. Had the CNN been trained on a random subset as well, we believe it would have outperformed the DNN.

3.2 Understanding the neural network

Figure 4b shows the intermediate outputs for the three neuron hidden layer along with (6) for the optimized $\epsilon = 0.86$. It is clear that (6) is a good description of the network.

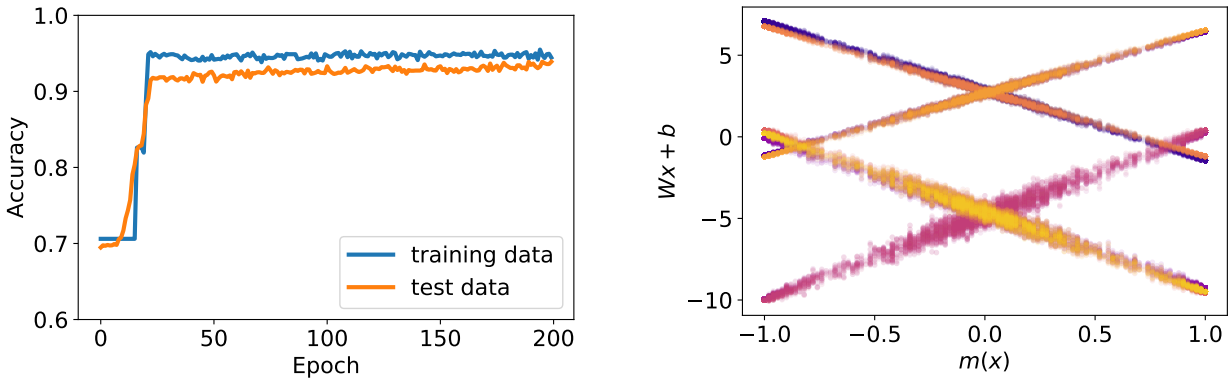
Figure 5b shows the intermediate outputs for the ten neuron hidden layer. It is clear that, even when adding more neurons, the three basic functions described in section 2.5 are the most effective states for the neurons to be in. The reason that we actually see four different patterns, instead of just the three from (6), is that we can also have a "$-m(x) + \epsilon$"-term, checking if the lattice is either polarized down or unpolarized. Any combination of terms which has at least three different of these four will function in the same way: measuring whether the lattice is polarized up, down, or unpolarized.

Comparing the model performances presented in Figure 4a and Figure 5a, we see that, while more neurons seem to allow the network to converge faster, they are effectively equal after 200 epochs. The accuracy of the three neuron network was 0.9511, while the accuracy was 0.9456 for the ten neuron network.



(a) (b)

Figure 4: Plot of intermediate output of 3 neuron hidden layer. To the intermediate output, the model (6) has been fitted. Accuracy was 0.9593 on training data and 0.9511 on test data.



(a) (b)

Figure 5: Plot of intermediate output of 10 neuron hidden layer. We can clearly see that all 10 outputs fall into four distinct categories. Accuracy on was 0.9595 on training data, and 0.9456 on test data.

3.3 Bayesian neuron

The performance of the Bayesian neuron will be studied by a number of figures. First Figure 6 is presented, which shows a corner plot describing the pdf of the three weights. There is a rather large variance on the weights, the standard deviation on w_1 and w_2 is approximately 0.5. This means that a rather large shift in weights could be equally likely. From Figure 6, we also see that w_1 and w_2 are correlated, which means that decision boundary would have approximately the same slope for all set of samples weights.

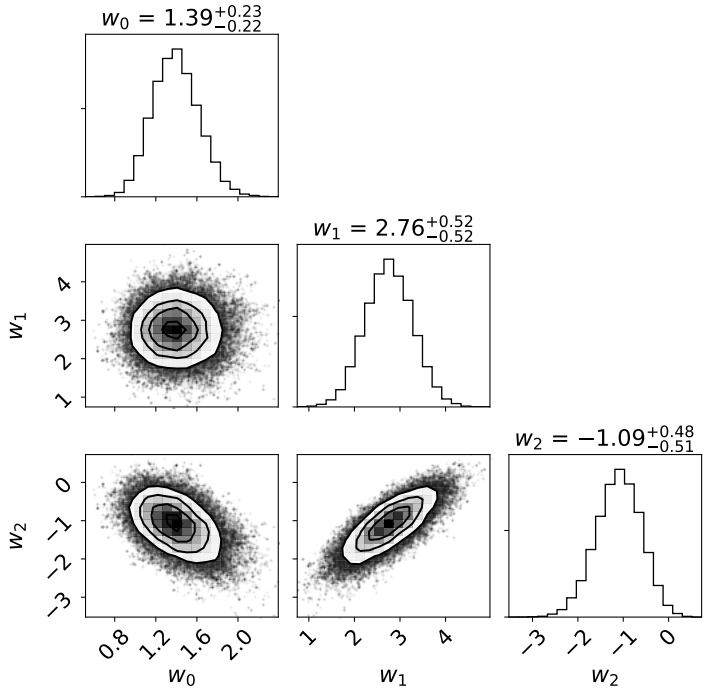


Figure 6: Caption

From figure 6, we see how the probabilities of the weights are distributed. In Figure 7, we see the effect of these distributions for three lattices. The figure shows the pdf:s of the prediction for one lattice with high temperature, one with low temperature, and one with temperature close to T_c. For the high temperature case, the BNN is extremely certain that it is an unordered phase, with approximately 100 % the prediction between 0.995 and 1. For the low temperature case, the BNN is also fairly certain of a ordered phase. For the intermediate case, there is greater range of prediction. The BNN is extrapolating in this region, so it is reasonable to both have a greater uncertainty of the prediction and also in the

probability of prediction.

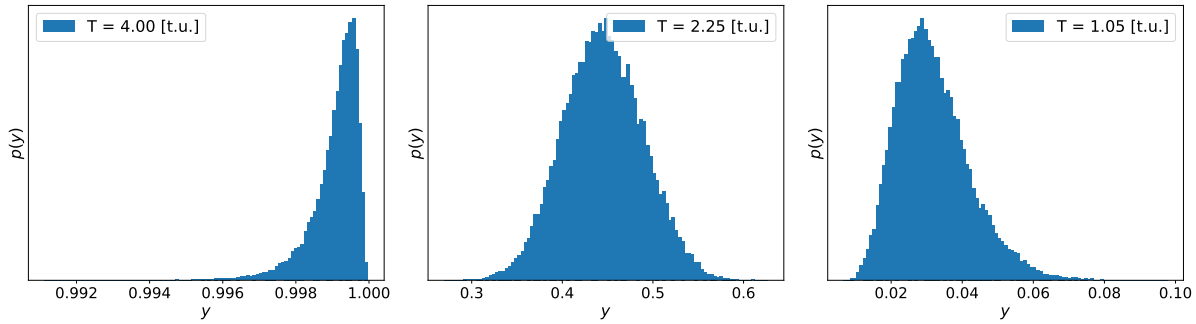


Figure 7: pdf of the prediction, y, of the Bayesian neuron for three cases: low temperature, high temperature, and medium temperature.

At last, Figure 8 is presented, which shows the mean and standard deviation of the pdf of the predictions for all possible values of energy and magnetization. The figure also shows all training data and their true phase. We also see that it exists a rather large band where the neuron isn't sure of the phase and the mean of the prediction pdf is between 0.25 and 0.75. This is reasonable since the data in this region have points in both phases. We observe that the standard deviation changes abruptly in the different areas. The standard deviation is low in areas where we have much data and higher in areas where there exist no data. This is very good since we ought to have greater uncertainties in our predictions where we don't have lots of observed data. This is the big advantage of BNN:s, that they can tell you how sure they are of their prediction.

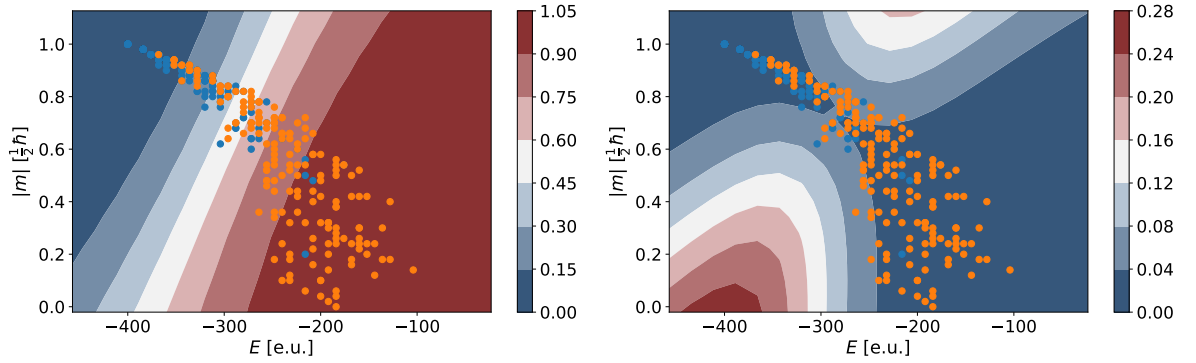


Figure 8: The figure shows to the left the mean of the Bayesian neurons prediction and to the right its standard deviation. The figure also displays all the training data with blue points being in ordered phase and orange points in unordered phase.

References

[1] *Ising model*, https://en.wikipedia.org/wiki/Ising_model.

[2] *Artificial neural network*, https://en.wikipedia.org/wiki/Artificial_neural_network#History.

[3] J. Carrasquilla and R. G. Melko, "Machine learning phases of matter," *Nature Physics*, vol. 13, pp. 431–434, 2017. DOI: https://doi.org/10.1038/nphys4035. [Online]. Available: https://www.nature.com/articles/nphys4035#supplementary-information.