Classification of the ising model - Documentation

Luca Schinnerl

March 7, 2021

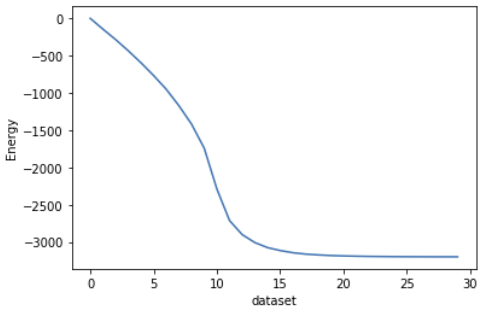
1 Goal of this project

First of all a brief description of what the goal of this project is will be provided. 30 different datasets have been proved. Each dataset corresponds to a Monte Carlo Simulation of the ising model at different temperatures. The labels of the datasets describe where the temperature of the simulation lays in respect to the critical temperature. Some are above and some are below the critical temperature. The analysis of which type of network structure shows the best performance (efficiency and accuracy) is provided. For this, the dataset has been dimensional reduced using principal component analysis. The performance with this data has then been compared to the non-reduced data. Finally, different machine learning methods have been applied. These include simple structures made out of dense layers, convolutional architectures and random forest classifiers.

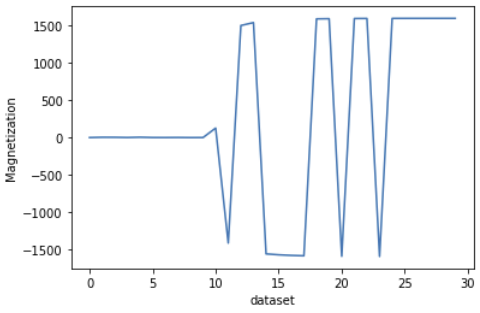
2 Generating appropriate datasets

First of all, the datasets provided were examined. How do the different phases look like? The average energies (figure 1a) and magnetisation (figure 1b) for each temperature can be plotted.

It can clearly be seen that the critical temperature is reached at dataset 10. This means all datasets with labels from 11 to 29 can be classified as being below the critical temperature, and all datasets with labels 0 to 9 can be classified as being above the critical temperature. Dataset 10 can not be classified in a binary classification as it describes exactly the critical temperature. The training and test data have been created with the function "train_test_split"



(a) Average energy of the spin configurations



(b) Average energy of the spin configurations

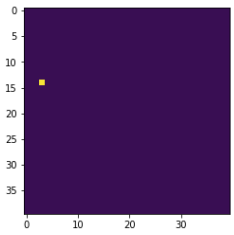
Figure 1: Plot of the average magnetisation and energy of the spin configurations at different temperatures

3 Principal Component Analysis

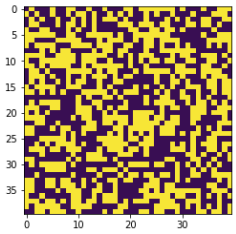
The X_train and X_test datasets have been dimensionally reduced to mere 20 components. For this, the datasets had to be flattened first (transforming the 2D structure into a single vector). This means these reduced datasets cannot be efficiently used within a convolutional architecture. The final data is a set of $29 * 1001$ spin configurations with 20 components each (instead of $40 * 40$ components each). This is a massive reduction in size and should significantly decrease computational complexity.

4 Classification with mostly Dense Layers

Before starting to build a model, looking at the complexity of this classification problem can be very useful. Two examples of the spin configurations with different labels have been plotted. It can



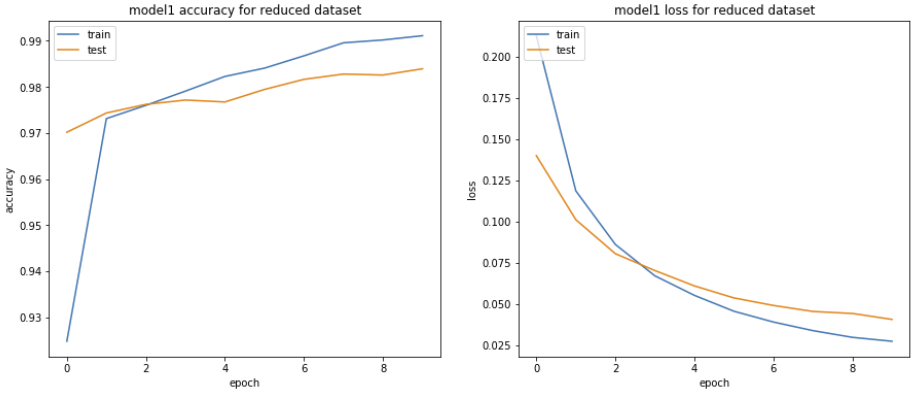
(a) Spin configuration with label 0



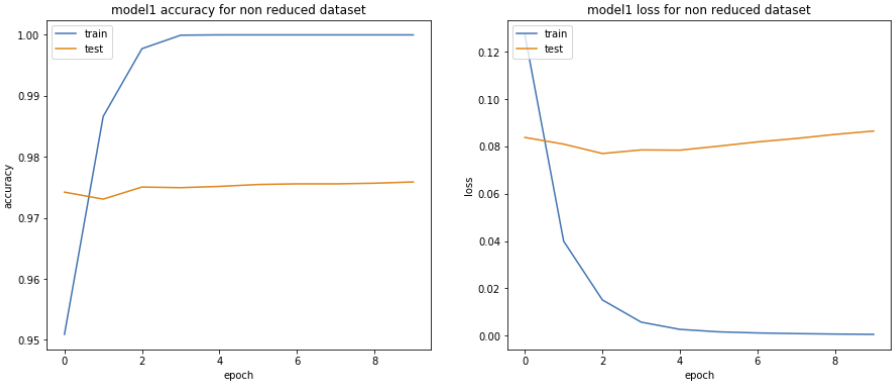
(b) Spin configuration with label 1

Figure 2: Two examples of spin configuration from the two different phases

immediately be seen that this is a relatively easy classification problem, as first of all, it is binary and second of all, a distinction can very easily be made between the two phases intuitively. In addition, with a small subset of the spin configurations, the phases still look extremely different. This is why a simple ANN structure was chosen, which is much simpler than a network that would be used for the MNIST data for example. For the first model, an architecture with a single hidden layer with 100 nodes was chosen. Its performance can be seen in figure 3. First, the choice of the Adam operator was successfully as it was able to find the minimum of the cost function very well, as training accuracy reached 1 after about 3 epochs on the non reduced dataset. However, we can see that for the non-reduced dataset overfitting occurs, as while the training accuracy increases, the test accuracy does not change. To prevent overfitting, measures such as a dropout layer, batch normalisation, and a simpler network (with only 10 hidden neurons) were implemented. The best preforming combination in the end was a model with a single hidden layer with 100 neurons and a dropout layer with a rate of 0.2, trained and tested on the reduced dataset. A validation accuracy of 0.9844 was reached while the training took 9.4 seconds (this could be reduced to 5 seconds as 10 epochs are not needed to find the minimum in the loss function).



(a) Performance on the reduced dataset



(b) Performance on the non reduced dataset

Figure 3: Performance of model 1

5 Classification with a convolutional architecture

As states before, a convolutional architecture can only be utilised efficiently when working with the 2D non reduced dataset. To start off, an architecture with a single convolutional layer was tested. Later a max pooling layer was added, which improved performance significantly. The best performing dense network from the last section was implemented in this convolutional architecture. It was immediately noticeable that training took more than 15 times as long, while validation accuracy was a bit higher than with only dense layers (0.9955), as expected. From figure 4, it can be seen that the model is not very consistent; and overfitting occurs as well. To combat this, just as in the last section, model complexity has been reduced and dropout layers have been introduced. The best combination in the end was a model with a single Conv2D layer, a dense layer with 10 neurons and a dropout layer with a rate of 0.2. Overfitting has mostly been reduced and the validation accuracy even exceeded the test accuracy. The validation accuracy is 0.996, while the training took about 130 seconds. Adding a second convolutional layer or changing some of the hyperparameter did not improve the performance.

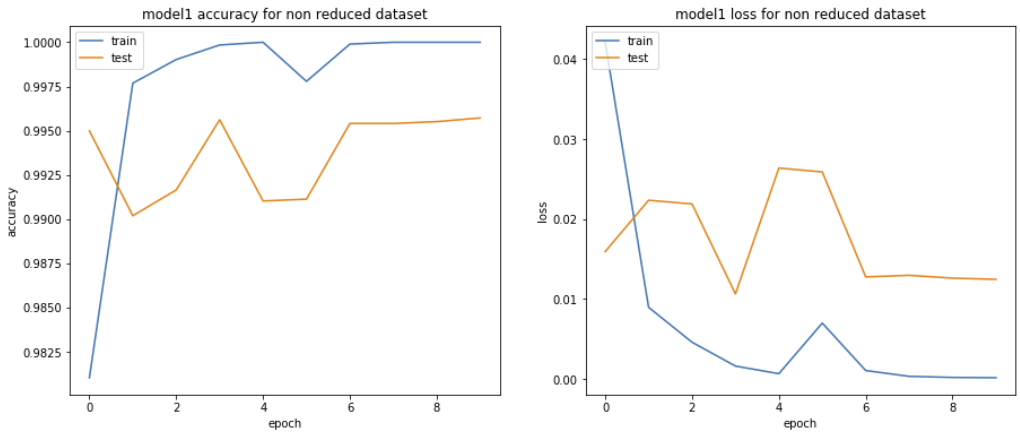


Figure 4: Performance of a model with one Conv2D, max pooling and dense layer with 10 hidden nodes

6 Random Forest classifier

To start off, a random forest with 100 estimators was implemented. The amount of estimators were then reduced step by step to see if accuracy changes. Accuracy did not deviate until reaching 20 estimators. The overall performance is extremely good, while it was best for the reduced dataset. Training only took 1.5 seconds while achieving a score of 0.986.
One more way to decrease computational cost is to introduce a maximal depth. Max depth of 10 on the non reduced data improves time a bit, but also decreases the score slightly. The score and time of the reduced dataset did not change at all, which means that the depth the Random Forest reaches is less than 11 in this case. One final way to improve performance is to preprocess the data and make them Standard Scaler. This again did not improve the performance at all. All in all, the best performance with a Random Forest was the default one on the reduced dataset with 20 estimators.

7 Comparison of the best approaches

Table 1 is a summary of the findings. It includes the best modles from each approach, as well as information on wheter or not the best performance has been reached with the reduced dataset. All models had similar accuracy of 0.990 ± 0.06. The convolutional architecture achieved the best accuracy while the Random Forest on the reduced data achieved the best overall performance. The problem with the convolutional network is that we are not able to utilise PCA here, making this classification a lot more complex than necessary. However, as this is computationally most complex, the performance is also the best. The assumption from the beginning, that this classification might be rather simple, has been reassured by the fact that best performance was always found when working with the reduced data (when possible).

	Time	Accuracy	Reduced
Dense	9.4	0.984	Yes
Convolutional	130	0.996	No
Random Forest	1.5	0.986	Yes

Table 1: Performance of the best model from each approach