Neural network - Spin Glass correspondence

F. Sorba

November 13, 2022

As it is well known [1, 2, 3, 4, 5], a deep and useful relation exists between neural networks and statistical mechanical models, in particular spin glasses (SG).

However, this correspondence does not seem to be fully exploited when it comes to convolutional neural networks (CNN), one of the most popular network models nowadays (see e.g. [6, 7] for a few related studies).

Here, we intend to investigate the relation between CNN and systems of spin glasses. This could result in the possibility to train the network by finding the ground state of the reciprocal spin model. In simple terms, we have the following scenario:

- NN: Fixed neurons state, evolving weights $\longrightarrow$ minimize loss function.
- SG: Fixed magnetisation, evolving spins state $\longrightarrow$ minimize free energy.

The idea is to identify the network's weights and neurons values (for a given training dataset) respectively with the spins and magnetisation of the spin glass model.

A few advantages can be envisioned by using this approach:

- A different training method could allow to train models where standard backpropagation fails (e.g. vanishing gradient problem).
- The temperature parameter in the spin model (which regulates how deep in the energy potential the system resides) could help to contrast overfitting.
- The relation between classical statistical mechanics and quantum mechanics [8] could be exploited to represent the neural network as a quantum spin system which can be trained on quantum computers (e.g. by quantum annealing [9]). The reduced dimensionality of the quantum system [10, 11] implies lower complexity which could result in an easier/faster to train model.

1 Overview

A general feed-forward layer of a neural network can be written as

$$s_i = f\left(\sum_j \omega_{ij}\sigma_j - \theta_i\right) , \tag{1}$$

where σ_i, $i = 0, \dots N$ is the input layer, s_i, $i = 0, \dots M$ the output layer, ω_{ij} the synapsis weights, θ_i the thresholds and $f(x)$ the activation function.

Training the layer means find the configuration $\{\omega_{ij}, \theta_i\}$ that minimize a loss function over a training dataset. For example, the mean square error loss function:

$$D = \frac{1}{Q} \sum_a^Q \sum_i^N (s_i(\sigma^a) - \xi_i^a)^2 \,, \tag{2}$$

where σ^a is the a-th training sample and ξ^a the corresponding expected result.

The Hamiltonian of the Ising model[1] is

$$H = -\sum_j (\sum_i J_{ij}\sigma_i - h_j)\sigma_j \,, \tag{3}$$

where σ_i is the spin at site i, J_{ij} the magnetisation and h_j the external magnetic field. In the standard Ising model the spin variables takes only the discrete values $\sigma_i = \pm 1$. In order to make a correspondence between the feed-forward layer and the spin glass model, we want to have continuous variables in both models. To do so, we could introduce n replica of the spin model and interpret the spin variables as an n-bit representation of a continuous variable. Or consider directly continuous models like the continuous spin Ising model [13, 14] or the XY model [15] (where the spin variables are replaced by continuous "angles"):

$$s_i \longrightarrow (\cos\theta_i, \sin\theta_i) \,, \qquad s_i s_j = \cos(\theta_i - \theta_j) \,, \tag{4}$$

and the Hamiltonian becomes

$$H_{XY} = -\sum_i \left(\sum_j J_{ij} cos(\theta_i - \theta_j) + h_i \cos\theta_i \right) \,. \tag{5}$$

In statistical models (like Ising or XY), the spin state evolves to the equilibrium (i.e. the state of minimum free energy) for a given temperature T, while in a neural network, weights evolve to minimize the loss function. Is it possible to find a correspondence between these two mechanisms?

1.1 Fully connected layer

Again let's consider a 1-layer feed-forward network defined as

$$s_i = f\left(\sum_j^M \omega_{ij}\sigma_j + \theta_i \right) \,, \qquad i = 0, \dots N \,, \tag{6}$$

For convenience, we introduce also the inactivated neuron

$$\bar{s_i} = f^{-1}(s_i) = \sum_j^M \omega_{ij}\sigma_j + \theta_i \,. \tag{7}$$

[1]Strictly speaking, in the Ising model only nearest neighbour interactions are considered. The case where each spin can interact with any other spin goes by the name of Sherrington-Kirkpatrick model [12].

Given the training set $\{\sigma_i^a, \xi_i^a\}$ (where $i = 0, \dots N$ runs over the input dimension and $a = 0, \dots Q$ over the samples in the datasets), the mean square error loss function can be rewritten as:

$$D = \frac{1}{Q} \sum_a^Q \sum_i^N \left(\bar{s_i}^a - f^{-1}(\xi_i^a) \right)^2 , \tag{8}$$

And expanding the square, we get

$$D = \frac{1}{Q} \sum_a^Q \sum_i^N \left(\bar{s_i}^a \bar{s_i}^a - 2\bar{s_i}^a f^{-1}(\xi_i^a) + f^{-1}(\xi_i^a)^2 \right) . \tag{9}$$

For simplicity, let assume $\theta_i = 0$. Then, substituting Eq. (7), the expression in brackets becomes

$$(\dots) = \sum_{jl}^M \omega_{ij}\omega_{il}\sigma_j^a\sigma_l^a + \sum_j^M \omega_{ij}\sigma_j^a(-2f^{-1}(\xi_i^a)) + f^{-1}(\xi_i^a)^2 . \tag{10}$$

By defining:

$$\begin{aligned} J_{jl} &\equiv \frac{1}{Q}\sum_a^Q \sigma_j^a\sigma_l^a , \qquad & h_j^i &\equiv \frac{1}{Q}\sum_a^Q (-2f^{-1}(\xi_i^a)\sigma_j^a) , \\ E_0^i &\equiv \frac{1}{Q}\sum_a^Q f^{-1}(\xi_i^a)^2 , \qquad & S_j^i &\equiv \omega_{ij} , \end{aligned} \tag{11}$$

the loss function becomes

$$D = \sum_i^N \left(\sum_{jl}^M J_{jl} S_j^i S_l^i + \sum_j^M h_j^i S_j^i + E_0^i \right) = \sum_i^N H_i(S_i) = H(\vec{S}) . \tag{12}$$

The loss function has been mapped to a system of N spin glass Hamiltonians. Consequently, finding the ground state of the spin glass system corresponds to find the weights configuration that minimize the loss function.

1.2 Convolutional layer

The same approach can be applied to convolutional layers. A convolutional layer can be written as

$$s_i = f\left(\sum_k^K c_k \sigma_{i-k} \right) , \tag{13}$$

where K is the dimension of the kernel c. Again, by expanding Eq. (8) we obtain

$$D = \frac{1}{Q}\sum_a^Q\sum_i^N \left(\sum_{kl}^K c_k c_l \sigma_{i-k}^a \sigma_{i-l}^a + \sum_k^K (-2f^{-1}(\xi_i^a)) c_k \sigma_{i-k} + f^{-1}(\xi_i^a)^2 \right) . \tag{14}$$

And defining

$$\begin{aligned} J^i_{kl} &\equiv \frac{1}{Q}\sum_a^Q \sigma^a_{i-k}\sigma^a_{i-l}\,, & h^i_k &\equiv \frac{1}{Q}\sum_a^Q(-2f^{-1}(\xi^a_i)\sigma^a_{i-k})\,, \\ E^i_0 &\equiv \frac{1}{Q}\sum_a^Q f^{-1}(\xi^a_i)^2\,, & S_k &\equiv c_k\,, \end{aligned} \tag{15}$$

we arrive at

$$D = \sum_i^N \left(\sum_{kl}^K J^i_{kl}S_kS_l + \sum_k^K h^i_kS_k + E^i_0\right) = \sum_i^N H_i(\vec{S})\,, \tag{16}$$

which similarly to Eq. (12) is the Hamiltonian of a spin glass system.

1.3 Multilayers network

We have seen how a single layer network can be described in terms of a spin glass system. Is it possible to extend the correspondence to multilayered networks?

Let's start by considering a 2 layers network:

$$s_i = f\left(\sum_l^L \bar{\omega}_{il}\tau_l\right)\,, \qquad \tau_l = f\left(\sum_j^M \omega_{lj}\sigma_j\right)\,, \tag{17}$$

where s_i are the N output neurons, τ_l the L hidden neurons and σ_j the M input neurons.

Consider the mean square error loss function (8) and the training set $\{\sigma^a_i, \xi^a_i\}$, $a = 0 \dots Q$. Due to the activation function in the hidden layer it is difficult to obtain a spin glass system directly. It seems more manageable to split the loss function between the 2 layers:

$$\begin{aligned} D_1 &= \frac{1}{Q}\sum_a^Q\sum_i^N \left(\bar{s}_i(\tau^a) - f^{-1}(\xi^a_i)\right)^2\,, \\ D_0 &= \frac{1}{Q}\sum_a^Q\sum_l^L \left(\bar{\tau}_l(\sigma^a) - f^{-1}(\zeta^a_l)\right)^2\,, \end{aligned} \tag{18}$$

where τ^a and ζ_a are the (unknown) expected input and output values in the trained hidden layer corresponding to the training sample a:

$$\tau^a_l = f\left(\sum_j^M \omega_{lj}\sigma^a_j\right)\,, \qquad \zeta^a_l = \sum_i^N \bar{\omega}^{-1}_{li} f^{-1}(\xi^a_i)\,. \tag{19}$$

Note that ζ^a_l is obtained by backpropagating the output sample ξ^a_i through the inverse output layer. Also, in this scenario ω_{lj} and $\bar{\omega}_{il}$ are supposed to be the weights in the already trained network.

In order to actually train the network, we then assume that it is possible to proceed in a loop:

1. Generate τ^a and ζ_a using the current network weights ω, $\bar{\omega}$

2. Minimize D_0 and D_1 obtaining new values for ω, $\bar{\omega}$
3. Repeat from point 1 until a stationary point is reached.

The above procedure requires to backpropagate the output samples at each loop step. Since matrix inversion is time consuming, it is also possible to train directly the inverse output network and only invert it once at the end of the procedure.

To recapitulate, in this scenario we will start with the network layers

$$\tau_l = \sum_i^N \bar{\omega}_{li}^{-1} f^{-1}(s_i)\,, \qquad \tau_l = f\left(\sum_j^M \omega_{lj}\sigma_j\right)\,, \tag{20}$$

and minimize the loss functions

$$\begin{aligned} D_1^{-1} &= \frac{1}{Q}\sum_a^Q \sum_l^L \left(\sum_i^N \bar{\omega}_{li}^{-1} f^{-1}(\xi_i^a) - \tau_l^a\right)^2\,, \\ D_0 &= \frac{1}{Q}\sum_a^Q \sum_l^L \left(\sum_j^M \omega_{lj}\sigma_j^a - f^{-1}(\zeta_l^a)\right)^2\,, \end{aligned} \tag{21}$$

where τ_l^a and ζ_l^a are re-calculated from Eq. (19) at every loop iteration with the updated weights configurations. Finally, the inverse output weights will be inverted[2] to recover the original feed forward architecture:

$$\bar{\omega}_{li}^{-1} \longrightarrow \bar{\omega}_{il} \Longrightarrow s_i = f\left(\sum_l^L \bar{\omega}_{il}\tau_l\right)\,. \tag{22}$$

Now, we can use the identifications (11) in the loss functions (21) to reinterpret them as 2 spin glass Hamiltonians:

$$\begin{aligned} D_1^{-1} &= \sum_l^L \bar{H}_l(\bar{S}_l) = \sum_l^L \left(\sum_{ij}^N \bar{J}_{ij}\bar{S}_i^l \bar{S}_j^l + \sum_i^N \bar{h}_i^l \bar{S}_i^l + \bar{E}_0^l\right)\,, \\ D_0 &= \sum_l^L H_l(S_l) = \sum_l^L \left(\sum_{jh}^M J_{jh} S_j^l S_h^l + \sum_j^M h_j^l S_j^l + E_0^l\right)\,, \end{aligned} \tag{23}$$

where

$$\begin{aligned} &\bar{J}_{ij} \equiv \frac{1}{Q}\sum_a^Q f^{-1}(\xi_i^a) f^{-1}(\xi_j^a)\,, \qquad \bar{h}_i^l \equiv \frac{1}{Q}\sum_a^Q (-2 f^{-1}(\xi_i^a)\tau_l^a)\,, \\ &\bar{E}_0^l \equiv \frac{1}{Q}\sum_a^Q (\tau_l^a)^2\,, \qquad \bar{S}_i^l \equiv \bar{\omega}_{li}^{-1}\,, \end{aligned} \tag{24}$$

[2]While inversion is strictly defined only for square matrices, for non-degenerate non-square matrices a left or right inverse can be defined.

and

$$J_{jh} \equiv \frac{1}{Q}\sum_a^Q \sigma_j^a \sigma_h^a\,, \qquad h_j^l \equiv \frac{1}{Q}\sum_a^Q(-2f^{-1}(\zeta_l^a)\sigma_j^a)\,,$$
$$E_0^l \equiv \frac{1}{Q}\sum_a^Q f^{-1}(\zeta_l^a)^2\,, \qquad S_j^l \equiv \omega_{lj}\,. \tag{25}$$

To train the network as a spin glass system, the procedure is analogous to the one described above:

1. Initialize the spins S, $\bar{S}$ randomly
2. Compute the magnetisations J and $\bar{J}$ from the training set $\{\sigma^a, \xi^a\}$ (Eqs. (24, 25))
3. Compute the magnetic fields h, $\bar{h}$ and ground energies E_0, $\bar{E}_0$ for the given spin configuration (Eqs. (19, 24, 25))
4. Minimize the Hamiltonians (Eqs. (23))
5. Repeat steps 3, 4 until a minimum of the system energy is reached
6. Invert the output spin layer to obtain the original network architecture.

Note that the same approach can be generalized to networks containing more than 2 layers and where convolutional layers are present (see. Eq. (16)).

2 Tests and results

In this section, we briefly present the results from the numerical tests run to verify the results derived above. The code of the tests is available on Github.

2.1 Single feed-forward layer

Let's start by considering the 1-layer case discussed in Sec. 1.1. We use as dataset the California housing dataset from Scikit-learn. In this dataset, the input sample has dimension 8 and the expected output is one-dimensional, so the network consists of 8 input neurons and 1 output neuron.

In Fig. 1, the loss (energy) of the spin glass model is compared to the loss of the neural network over 1000 training steps. The two training methods seems to achieve comparable results.

We observe however that over many iterations it is common to see the spin glass stuck in some non optimal local minimum. Since we used a very basic algorithm for the evolution of the spin model, it is expected that more refined methods will be able to overcome the issue.

2.2 2-layers feed-forward network

Also in this case, we train the model on the California housing datasatet. The network is composed of 8 input neurons, 20 hidden neurons and 1 output neuron. As before we train the system over 1000 steps updating the hidden layer configuration (as explained in Sec. 1.3) every 200 steps.

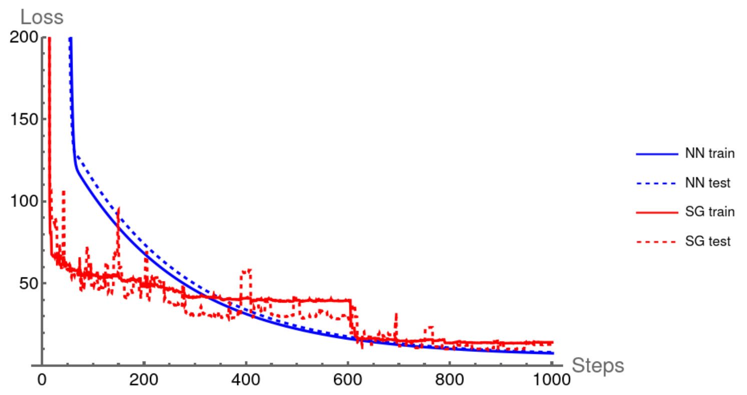

Figure 1: Comparison of loss functions obtained by training the 1-layer model as a neural network by backpropagation (blue) and by training the model as spin glass reaching equilibrium (red).

In figure 2, the global loss of the spin glass model over the training steps is depicted. In this case, it is not possible to directly compare this quantity with the loss obtained in the standard neural model. To have an indication of the result accuracy, we convert the trained spin model back to the network representation and evaluate it on the test set comparing the results with those obtained by training the network by backpropagation. The losses for a few different runs are reported in Table 1.

Spin Glass	Neural Network
1.067	2.215
1.643	12.32
3.544	3.427
3.678	10.40
2.176	5.401

Table 1: Losses obtained by evaluating the neural network over the test set when trained as spin glass and by backpropagation.

References

[1] Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Spin-glass models of neural networks. *Phys. Rev. A*, 32:1007–1018, Aug 1985.

[2] J.A. Hertz, hn Hertz, S.K. Andersen, ders Krogh, A.S. Krogh, R.G. Palmer, G. Palmer, Addison-Wesley (1942-1999)., and N.M.) Santa Fe Institute (Santa Fe. *Introduction To The Theory Of Neural Computation*. Addison-Wesley Computation and Neural Systems Series. Avalon Publishing, 1991.

[3] V. Dotsenko. *An Introduction To The Theory Of Spin Glasses And Neural Networks*. World Scientific Lecture Notes In Physics. World Scientific Publishing Company, 1995.

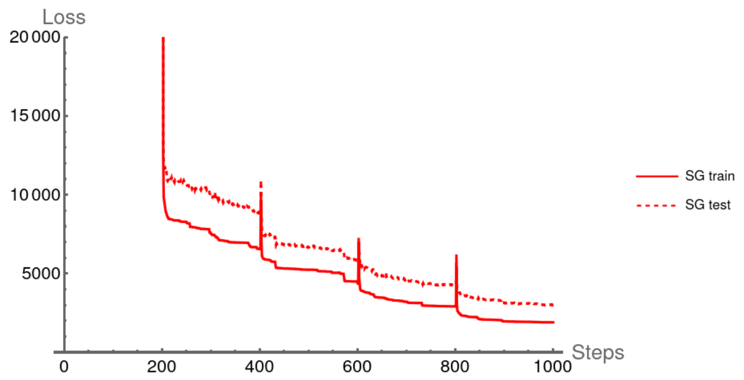

Figure 2: Loss functions obtained by training the 2-layers network as as spin glass reaching equilibrium.

[4] B. Müller, J. Reinhardt, and M.T. Strickland. *Neural Networks: An Introduction*. Physics of Neural Networks. Springer Berlin Heidelberg, 1995.

[5] H. Huang. *Statistical Mechanics of Neural Networks*. Springer Nature Singapore, 2022.

[6] Sunil Pai. Convolutional neural networks arise from ising models and restricted boltzmann machines. 2016.

[7] Pankaj Mehta and David J. Schwab. An exact mapping between the variational renormalization group and deep learning. *arXiv preprint arXiv: 1410.3831*, 2014.

[8] Michael Edward Peskin and Daniel V. Schroeder. *An Introduction to Quantum Field Theory*. Westview Press, 1995. Reading, USA: Addison-Wesley (1995) 842 p.

[9] Giuseppe E. Santoro, Roman Martonak, Erio Tosatti, and Roberto Car. Theory of quantum annealing of an ising spin glass. *Science*, 295(5564):2427–2430, mar 2002.

[10] J. Hutchinson, J. P. Keating, and F. Mezzadri. On relations between one-dimensional quantum and two-dimensional classical spin systems, 2015.

[11] Timothy H. Hsieh. From d-dimensional quantum to d+1-dimensional classical systems. 2012.

[12] Dmitry Panchenko. The sherrington-kirkpatrick model: An overview. *Journal of Statistical Physics*, 149(2):362–383, sep 2012.

[13] Henk van Beijeren and Garrett S Sylvester. Phase transitions for continuous-spin ising ferromagnets. *Journal of Functional Analysis*, 28(2):145–167, 1978.

[14] Piotr Bialas, Philippe Blanchard, Santo Fortunato, Daniel Gandolfo, and Helmut Satz. Percolation and magnetization in the continuous spin ising model. *Nuclear Physics B*, 583(1-2):368–378, sep 2000.

[15] Eytan Barouch and Barry M McCoy. Statistical mechanics of the x y model. ii. spin-correlation functions. *Physical Review A*, 3:786–804, 1971.