
Boltzmann Machine

The Boltzmann machine (named in honour of the 19th-century physicist and inventor of statistical mechanics by its inventors) has similarities to and differences from the Hopfield net.

Invented by Terry Sejnowski

- Convinctor of the Boltzmann machine
- Director of Computational Neurobiology Lab at Salk Institute for Biological Studies
- Professor at UCSD
- Founder and editor of the journal Neural Computation

We've already "met" the other inventor, Geoffrey Hinton.

Structure of Boltzmann Machine

- The stochastic neurons of a Boltzmann machine are in two groups: visible and hidden
- visible neurons provide an interface between the net and its environment
- during the training phase, the visible neurons are clamped; the hidden neurons always operate freely, they are used to explain underlying constraints in the environmental input vectors.

Figure 11.4: Architectural graph of Boltzmann machine. $K$ is the number of visible neurons and $L$ is the number of hidden neurons.
Boltzmann Machine Learning

The goal of Boltzmann learning is to produce a NN that categorizes input patterns according to a Boltzmann distribution. Two assumptions are made:

- Each environmental vector persists long enough for the network to reach thermal equilibrium;
- There is no structure in the sequence in which environmental vectors are clamped to the visible units of the network.

Energy Minimization

- The Boltzmann machine works by picking a hidden unit at random - say unit $j$, and flipping the state of neuron $j$ from $s_j$ to $-s_j$ at temperature $T$ (during the annealing cycle) with probability

$$\text{Prob}(s_j \rightarrow -s_j) = \frac{1}{1 + \exp(-\Delta E_j / T)}$$

where $\Delta E_j$ is the energy change resulting from such a flip. We define the energy function of the Boltzmann machine as:

$$E = -0.5 \sum_{i} \sum_{j \neq i} w_{ji} s_i s_j$$

- This summation runs over both visible and hidden units. The condition $j \neq i$ implies no self-feedback. $w_{ji}$ is the weight from unit $i$ to unit $j$. An external threshold $\theta_j$ applied to unit $j$ is provided as usual (a weight of $-\theta_j$ from a unit with a fixed output of 1).
- If the flipping procedure is applied repeatedly to the units, the net will reach thermal equilibrium. A thermal equilibrium, the units will change state, but the probability of finding the network in any particular state remains constant and follows the Boltzmann distribution.
- To find a stable configuration that is suited to the problem at hand, Boltzmann learning proceeds by first operating the net at high temperature, and gradually lowering it until it reaches thermal equilibrium at a series of temperatures, as prescribed by the simulated annealing procedure.

Summary of the Boltzmann Machine Learning Procedure

For details, particularly of derivation, see pages 598-604 of Haykin.

1. **Initialization**: set weights to random numbers in $[-1,1]$

2. **Clamping Phase**: Present the net with the mapping it is supposed to learn by clamping input and output units to patterns. For each pattern, perform simulated annealing on the hidden units at a sequence $T_0, T_1, ..., T_{final}$ of temperatures. (AHS suggest a sequence they describe as $[2@20, 2@15, 2@12, 4@10]$ – see details after step 5). At the final temperature, collect statistics to estimate the correlations

$$\rho_{ji}^+ = \langle s_j s_i \rangle^+ \quad (j \neq i)$$

Here $\langle s_j s_i \rangle^+ = \sum_{s_\beta \in \mathcal{S}} \sum_{s_\alpha \in \mathcal{S}} P(S_\beta = s_\beta | S_\alpha = s_\alpha) s_j s_i$ where $\mathcal{S}$ is the set of training examples, $s_\alpha$ represents the vector of visible neurons, and $s_\beta$ represents the vector of hidden neurons (and $s$ would be the vector of all neurons). $\mathcal{S}$ is the training sample.

4. **Updating of Weights**: update them using the learning rule

$$\Delta w_{ji} = \eta (\rho_{ji}^+ - \rho_{ji}^-)$$

where $\eta$ is a learning rate parameter, and it depends on the temperature $T$ ($\eta = \frac{1}{T} - 1$).

5. **Iterate until Convergence**: Iterate steps 2 to 4 until the learning procedure converges with no more changes taking place in the synaptic weights $w_{ji}$ for all $j, i$. 

3. **Free-Running Phase**: Repeat the calculations performed in step 2, but this time clamp only the input units. Hence, at the final temperature, estimate the correlations

$$\rho_{ji}^- = \langle s_j s_i \rangle^- \quad (j \neq i)$$

Here $\langle s_j s_i \rangle^- = \sum_{s_\beta \in \mathcal{S}} \sum_{s_\alpha \in \mathcal{S}} P(S_\beta = s_\beta | S_\alpha = s_\alpha) s_j s_i$
Details on How Annealing Procedure Fits In

- This section expands on the annealing schedule \([2@20, 2@15.2@12.4@10]\) quoted above.
- The simulated annealing starts by randomising the activations of the unclamped neurons. This is equivalent to reaching thermal equilibrium at infinite temperature.
- We use the phrase “of the unclamped neurons” a lot below, so we’ll abbreviate it to “otUN”.
- The activations \(\text{otUN}\) are maintained from training pattern to training pattern and from epoch to epoch (= run through all the training patterns).
- So for the first training pattern at temperature \(T=20\), we start with the "infinite temperature" activations in the unclamped neurons, which thus influence the new activations \(\text{otUN}\), which will in turn influence the activations \(\text{otUN}\) for the second training pattern at temperature 20, ..., last training pattern at temperature 20 (end of first epoch).
- The final activations \(\text{otUN}\) from the first epoch become the initial activations \(\text{otUN}\), etc., all the way down to the final pattern of the last epoch at \(T=10\).
- Then the final activations \(\text{otUN}\) from that epoch are used to start off the statistics collection run. Statistics are collected at a temperature of \(T=10\) for 10 epochs.

Boltzmann Machine Applications

As with Hopfield nets, the old Attrasoft site at [http://attrasoft.com/Boltzmann.html](http://attrasoft.com/Boltzmann.html) has now vanished, but the old list of link names, below, gives an indication of applications:

- Boltzmann Machine in stock market trend prediction
- Boltzmann Machine in character recognition
- Boltzmann Machine in Face recognition
- Boltzmann Machine in Internet Application
- Boltzmann Machine in Cancer Detection
- Boltzmann Machine in Loan Application
- Boltzmann Machine in Decision making

You can try the following archive site for old copies of these pages: