Neural Networks
Lecture Notes
Boltzmann machines

Marcel van Gerven

1 Learning Goals

After studying the lecture material you should:

1. be able to explain in words what is meant by simulated annealing
2. know what the update rule for a Boltzmann machine looks like
3. know how to compute the entries of a state transition matrix
4. know what is meant by the stable distribution of a Boltzmann machine
5. know how to write the probability of being in a state $\mathbf{x}$ using the Boltzmann distribution
6. know what is meant by an energy-based model
7. understand the learning rules for fully observed and partially observed BMs

2 Notes

2.1 Boltzmann machines

A *Boltzmann machine* (BM) [AHS85] is a Hopfield network whose states are stochastically updated according to:

$$P(x_i = 1) = \sigma\left(\frac{1}{T}(\mathbf{w}_i^T\mathbf{x} + \theta_i)\right)$$

where T is the temperature and $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function. BMs use sigmoid functions to define stochastic network dynamics. The higher the temperature, the more stochastically the states switch. Stochastic updates at $T > 0$ are known as *Glauber dynamics*.

We can use Boltzmann machines to find better minima using *simulated annealing*: Start the BM at high temperature T and gradually decrease it while updating the model? For $T \to \infty$ updates to a higher energy are accepted half the time. For $T \to 0$ only updates which reduce the value of E are accepted with probability 1 (Hopfield network). Varying the temperature from large values all the way down to zero corresponds to the heating and cooling process of annealing. This gives a higher probability of ending up in a global minimum.

We can compute the probability of moving from one state to another state. Consider the flip-flop with two units, a weight of 1 between them and thresholds of 0.5. The probability of transitioning from state 00 to state 01 at a temperature of $T = 1$ (default assumption for BMs) can

be computed as:

$$\begin{aligned} p_{00\to01} &= P(\text{update 2nd unit})P(x_2^t = 1 \mid x_1^{t-1} = 0, x_2^{t-1} = 0) \\ &= 0.5 \cdot \frac{1}{1+\exp(-(\sum_j w_{ij}x_j + \theta_i)/T)} \\ &= 0.5 \cdot \frac{1}{1+\exp(-(-1\cdot 0 + 0.5)/1)} \\ &= 0.5 \cdot \frac{1}{1+\exp(-0.5)} = 0.31 \end{aligned}$$

All of these transitions can be represented in a *state transition matrix* $\mathbf{P}$ such that p_{ij} denotes the probability of moving from state i (e.g. 00) to state j (e.g. 01).

Given the state transition matrix and a vector $\mathbf{v}_0$, listing for all states (four in case of the flip-flop) its initial probability of finding the network in that state, we can compute the probability distribution over network states after one transition as $\mathbf{v} = \mathbf{v}_0$. In general we have a Markov chain such that $\mathbf{v}_t = \mathbf{v}_{t-1}\mathbf{P}$. If we run the BM long enough then it will eventually reach *thermal equilibrium*. This means that the probability distribution v over states satisfies: $\mathbf{v} = \mathbf{v}\mathbf{P}$.

2.2 Relation between transition probabilities and energy levels

Note that the energy function of a BM is the same as for a Hopfield network. Transition probabilities between two states α and β of the Boltzmann machine are related to the energy levels of these states via:

$$p_{\alpha\to\beta} = \frac{1}{1+\exp((E_\beta - E_\alpha)/T)} \tag{1}$$

where E_α and E_β are the energies of the corresponding states. Physical systems obeying this relationship have the *Boltzmann distribution* as their equilibrium distribution, which is defined as

$$p(\mathbf{x}) = \frac{1}{Z}\exp(-E(\mathbf{x})/T)$$

with $Z = \sum_{\mathbf{x}} \exp(-E(\mathbf{x})/T)$. This also holds for the Boltzmann machine and this is where the Boltzmann machine gets its name from. We will move on to show that Equation (1) holds for the BM, which implies the Boltzmann distribution as its equilibrium distribution.

First note that if a transition occurs from state α to state β then some unit k must have changed its state from x_k to x'_k. The energy of the BM in different states is given by:

$$\begin{aligned} E_\alpha &= -\frac{1}{2}\sum_{i\neq k}\sum_j w_{ij}x_i x_j - \sum_{i\neq k}\theta_i x_i - \sum_j w_{kj}x_j x_k - \theta_k x_k \\ E_\beta &= -\frac{1}{2}\sum_{i}\sum_j w_{ij}x_i x_j - \sum_{i}\theta_i x_i - \sum_j w_{kj}x_j x'_k - \theta_k x'_k \end{aligned}$$

The contribution of unit k in the last two terms is the only difference between both energy levels. This energy difference is therefore given by:

$$E_\beta - E_\alpha = -\sum_j w_{kj}x_j(x'_k - x_k) - \theta_k(x'_k - x_k)$$

Assume that $x_k = 0$ (state α) and $x'_k = 1$ (state β) such that

$$E_\beta - E_\alpha = -\sum_j w_{kj}x_j - \theta_k$$

The probability that x_k assumes state 1 (i.e. we transition from state α to state β) is:

$$\begin{aligned} p_{\alpha\to\beta} &= \frac{1}{1+\exp\left(-\left(\sum_j w_{kj}x_j+\theta_k\right)/T\right)} \\ &= \frac{1}{1+\exp((E_\beta-E_\alpha)/T)} \end{aligned}$$

Conversely assume that $x_k = 1$ (state α) and $x'_k = 0$ (state β) such that

$$E_\beta - E_\alpha = \sum_j w_{kj}x_j + \theta_k$$

The probability that x_k assumes state 0 (i.e. we transition from state α to state β) is:

$$\begin{aligned} p_{\alpha\to\beta} &= 1 - \frac{1}{1+\exp\left(-\left(\sum_j w_{kj}x_j+\theta_k\right)/T\right)} \\ &= 1 - \frac{1}{1+\exp(-(E_\beta-E_\alpha)/T)} \end{aligned}$$

This proves that transition probabilities are determined by differences in energy levels.

2.3 Learning in Boltzmann machines

We now move on to describe learning in Boltzmann machines as these form the basis for models that can learn complex features in an unsupervised manner.

Energy-based models

Boltzmann machines can be used to learn latent representations in an unsupervised manner. Successes in this area spawned renewed interest in neural networks. We cannot learn stable memories as in the Hopfield network since there are no stable states for temperature $T > 0$. We can however train a BM to reproduce a stable probability distribution over network states. We derive a BM learning algorithm by starting from the notion of *energy-based models*.

In an energy-based model (EBM), the joint distribution over x is defined by:

$$P(\mathbf{x}) = \frac{1}{Z}\left(\exp(-E(\mathbf{x}))\right) \tag{2}$$

where Z is the *partition function* $Z = \sum_{\mathbf{x}} \exp\left(-E(\mathbf{x})\right)$ and $E(\mathbf{x})$ depends on parameters $\boldsymbol{\theta}$.

Boltzmann machines are a particular kind of EBM with

$$E(\mathbf{x}) = -\frac{1}{2}\mathbf{x}^T\mathbf{W}\mathbf{x} - \mathbf{x}^T\mathbf{b}\,. \tag{3}$$

We choose T=1 without loss of generality and now use $\mathbf{b}$ for the biases. Parameters $\boldsymbol{\theta}$ are given by weights $\mathbf{W}$ and biases $\mathbf{b}$. This is nothing else than a specification of the Boltzmann distribution.

Learning in fully observed Boltzmann machines

Consider a training set $D = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right\}$ for which we want to maximize the (log) probability of the observed training data under the model:

$$L(\boldsymbol{\theta}) = \log P(D \mid \boldsymbol{\theta}) = \log \prod_{n=1}^{N} P\left(\mathbf{x}^{(n)}\right) = \sum_{n=1}^{N} \log P\left(\mathbf{x}^{(n)}\right)\,. \tag{4}$$

Learning takes place via gradient ascent:

$$\Delta\theta \propto \sum_{n=1}^{N} \frac{\partial \log P\left(\mathbf{x}^{(n)}\right)}{\partial\theta} \tag{5}$$

which shows that this is written in terms of a sum over individual data points. We can also write the parameter update for multiple data points as:

$$\Delta\theta \propto \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \frac{\partial \log P(\mathbf{x})}{\partial \theta} \tag{6}$$

where

$$\hat{P}(\mathbf{x}) = \begin{cases} 1/N & \text{if } \mathbf{x} \in D \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

is the *empirical distribution*.

If we can compute the partial derivative for one data point then we know how to update the parameters. This partial derivative is given by:

$$\frac{\partial \log P\left(\mathbf{x}^{(n)}\right)}{\partial \theta} = -\frac{\partial E\left(\mathbf{x}^{(n)}\right)}{\partial \theta} + \sum_{\mathbf{x}} P(\mathbf{x}) \frac{\partial E(\mathbf{x})}{\partial \theta} \,. \tag{8}$$

The first term on the right hand side is the derivative of the energy for an observed vector $\mathbf{x}^{(n)}$. The second term on the right hand side is the expectation of the derivative of the energy under the *model distribution* P.

If we plug the partial derivative (8) into (6) and simplify, we obtain:

$$\Delta\theta \propto \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \left[-\frac{\partial E\left(\mathbf{x}\right)}{\partial \theta} + \sum_{\mathbf{z}} P(\mathbf{z}) \frac{\partial E(\mathbf{z})}{\partial \theta} \right] \tag{9}$$

$$= \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \left[\sum_{\mathbf{z}} P(\mathbf{z}) \frac{\partial E(\mathbf{z})}{\partial \theta} \right] - \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \frac{\partial E\left(\mathbf{x}\right)}{\partial \theta} \tag{10}$$

Since $\hat{P}$ sums to one and the term between brackets does not depend on $\mathbf{x}$, we obtain:

$$\Delta\theta \propto \sum_{\mathbf{x}} P(\mathbf{x}) \frac{\partial E(\mathbf{x})}{\partial \theta} - \sum_{\mathbf{x}} \hat{P}(\mathbf{x}) \frac{\partial E(\mathbf{x})}{\partial \theta} \tag{11}$$

By introducing notation $\mathbb{E}_P[f] = \sum_{\mathbf{x}} P(\mathbf{x}) f(\mathbf{x})$ for the *expectation* of the function f under a distribution P, we can rewrite (11) as

$$\Delta\theta \propto \mathbb{E}_P\left[\frac{\partial E}{\partial \theta}\right] - \mathbb{E}_{\hat{P}}\left[\frac{\partial E}{\partial \theta}\right] \,. \tag{12}$$

This shows that EBM learning amounts to computing the difference in the expectation of the energy gradient under the model distribution and the empirical distribution. This also makes clear that we need to be able to compute expectations as well as be able to evaluate gradients. The expectation under the empirical distribution can be computed by iterating over training examples. However, the expectation under the model distribution, written as

$$\mathbb{E}_P\left[\frac{\partial E}{\partial \theta}\right] = \sum_{\mathbf{x}} P(\mathbf{x}) \frac{\partial E(\mathbf{x})}{\partial \theta} \tag{13}$$

is impossible to compute for all 2^K different state vectors $\mathbf{x}$!

Instead we approximate the expectation by generating (high-probability) samples from the model distribution: $\mathbb{E}_P\left[\frac{\partial E}{\partial \theta}\right] \approx \frac{1}{T}\sum_{t=1}^{T} \frac{\partial E(\mathbf{x}^{(t)})}{\partial \theta}$ where $\mathbf{x}^{(t)} \sim P(\mathbf{x})$. Sampling from the model distribution is done via a procedure called *Gibbs sampling*:

- Start with a random initial state $\mathbf{x}^{(1)}$
- For $t = 2, \ldots, T$:

– Sample each variable in $\mathbf{x}^{(t)}$ from its conditional distribution:

$$x_j^{(t)} \sim P(x_j \mid \mathbf{x}_{\setminus j}) \tag{14}$$

$\Rightarrow$ when all variables are updated then you have one sample

- Discard the first part of these samples (burn in)
- Use the remaining samples to approximate $\mathbb{E}_P\left[\frac{\partial E}{\partial \theta}\right]$.

For the Boltzmann machine, the required partial derivatives are easy to compute. They are given by:

$$\frac{\partial E(\mathbf{x})}{\partial w_{ij}} = -x_i x_j \tag{15}$$

$$\frac{\partial E(\mathbf{x})}{\partial b_i} = -x_i \tag{16}$$

3 Reading material

For more background info on Boltzmann machines, consult [AHS85]. For background on energy-based models, consult [Ben09].

References

[AHS85] D Ackley, Geoffrey E Hinton, and T Sejnowski. A learning algorithm for Boltzmann machines. *Cogn. Sci.*, 9(1):147–169, 1985.

[Ben09] Y Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn. Mach. Learn.*, 2(1):1–87, 2009.