Simulated Annealing

input : $(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N) \in \mathbb{R}^d \times \{-1, +1\}; T_{start}, T_{stop} \in \mathbb{R}$

output: $\mathbf{w}$

begin

  Randomly initialize $\mathbf{w}$

  $T \leftarrow T_{start}$

  repeat

    $\widehat{\mathbf{w}} \leftarrow N(\mathbf{w})$ //neighbors of $\mathbf{w}$, e.g. by adding Gaussion noise $(\mathcal{N}(0, \sigma))$

    if $E(\widehat{\mathbf{w}}) < E(\mathbf{w})$ **then** $\mathbf{w} \leftarrow \widehat{\mathbf{w}}$

    else if $\exp\left(-\frac{E(\widehat{\mathbf{w}}) - E(\mathbf{w})}{T}\right) > rand[0, 1)$ **then**

      $\mathbf{w} \leftarrow \widehat{\mathbf{w}}$

    decrease (T)

  until $T < T_{stop}$

  return $\mathbf{w}$

end

Continuous Hopfield Network

Let us consider our previously defined Hopfield network (identical architecture and learning rule), however with following activity rule

$$S_i = \tanh\left(\frac{1}{T} \cdot \sum_j w_{ij} S_j\right)$$

Start with a large (temperature) value of T and decrease it by some magnitude whenever a unit is updated (deterministic simulated annealing).

This type of Hopfield network can approximate the probability distribution

$$P(\mathbf{x}|\mathbf{W}) = \frac{1}{Z(\mathbf{W})} \exp[-E(\mathbf{x})] = \frac{1}{Z(\mathbf{W})} \exp\left[\frac{1}{2}\mathbf{x}^T\mathbf{W}\mathbf{x}\right]$$

Continuous Hopfield Network

$$Z(\mathbf{W}) = \sum_{\mathbf{x}'} \exp(-E(\mathbf{x}')) \qquad \text{(sum over all possible states)}$$

is the partition function and ensures $P(\mathbf{x}|\mathbf{W})$ is a probability distribution.

Idea: construct a stochastic Hopfield network that implements the probability distribution $P(\mathbf{x}|\mathbf{W})$.

- Learn a model that is capable of generating patterns from that unknown distribution.
- Quantify (classify) by means of probabilities seen and unseen patterns.
- If needed, we can generate more patterns (generative model).

Boltzmann Machines

Given patterns $\{\mathbf{x}^{(n)}\}_1^N$, we want to learn the weights such that the generative model

$$P(\mathbf{x}|\mathbf{W}) = \frac{1}{Z(\mathbf{W})} \exp\left(\frac{1}{2}\mathbf{x}^T\mathbf{W}\mathbf{x}\right)$$

is well matched to those patterns. The states are updated according to the stochastic rule:

- set $x_n = +1$ with probability $\frac{1}{1+\exp\left(-2\sum_j w_{ij}x_j\right)}$
- else set $x_n = -1$.

Posterior probability of the weights given the data (Bayes' theorem)

$$P(\mathbf{W}|\{\mathbf{x}^{(n)}\}_1^N) = \frac{\left[\prod_{n=1}^N P(\mathbf{x}^{(n)}|\mathbf{W})\right] P(\mathbf{W})}{P(\{\mathbf{x}^{(n)}\}_1^N)}$$

Boltzmann Machines

Apply maximum likelihood method on the first term in numerator:

$$\ln\left[\prod_{n=1}^{N} P(\mathbf{x}^{(n)}|\mathbf{W})\right] = \sum_{n=1}^{N}\left[\frac{1}{2}\mathbf{x}^{(n)^T}\mathbf{W}\mathbf{x}^{(n)} - \ln Z(\mathbf{W})\right]$$

Taking derivative of the log likelihood gives: note that $\mathbf{W}$ is symmetric $(w_{ij} = w_{ji})$ that is $\frac{\partial}{\partial w_{ij}}\frac{1}{2}\mathbf{x}^{(n)^T}\mathbf{W}\mathbf{x}^{(n)} = x_i^{(n)}x_j^{(n)}$ and

$$\begin{aligned}
\frac{\partial}{\partial w_{ij}}\ln Z(\mathbf{W}) &= \frac{1}{Z(\mathbf{w})}\sum_{\mathbf{x}}\frac{\partial}{\partial w_{ij}}\exp\left(\frac{1}{2}\mathbf{x}^{(n)^T}\mathbf{W}\mathbf{x}^{(n)}\right) \\
&= \frac{1}{Z(\mathbf{W})}\sum_{\mathbf{x}}\exp\left(\frac{1}{2}\mathbf{x}^{(n)^T}\mathbf{W}\mathbf{x}^{(n)}\right)x_i x_j \\
&= \sum_{\mathbf{x}} x_i x_j\, P(\mathbf{x}|\mathbf{W}) = \langle x_i x_j\rangle_{P(\mathbf{x}|\mathbf{W})}
\end{aligned}$$

Boltzmann Machines (cont.)

$$\begin{aligned}\frac{\partial}{\partial w_{ij}} \ln P(\{\mathbf{x}^{(n)}\}_1^N|\mathbf{W}) &= \sum_{n=1}^{N}\left[x_i^{(n)}x_j^{(n)} - \langle x_i x_j\rangle_{P(\mathbf{x}|\mathbf{W})}\right] \\ &= N\left[\langle x_i x_j\rangle_{\text{Data}} - \langle x_i x_j\rangle_{P(\mathbf{x}|\mathbf{W})}\right]\end{aligned}$$

Empirical correlation between x_i and x_j

$$\langle x_i x_j\rangle_{\text{Data}} \equiv \frac{1}{N}\sum_{n=1}^{N}\left[x_i^{(n)}x_j^{(n)}\right]$$

Correlation between x_i and x_j under the current model

$$\langle x_i x_j\rangle_{P(\mathbf{x}|\mathbf{W})} \equiv \sum_{\mathbf{x}} x_i x_j P(\mathbf{x}|\mathbf{W})$$

Interpretation of Boltzmann Machines Learning

Illustrative description (MacKay's book, pp. 523):

- **Awake state:** measure correlation between x_i and x_j in the real world, and *increase* the weights in proportion to the measured correlations.
- **Sleep state:** dream about the world using the generative model $P(\mathbf{x}|\mathbf{W})$ and measure the correlation between x_i and x_j in the model world. Use these correlations to determine a proportional *decrease* in the weights.

If correlations in dream world and real world are matching, then the two terms balanced and weights do not change.

Boltzmann Machines with Hidden Units

To model higher order correlations hidden units are required.

- $\mathbf{x}$: states of visible units,
- $\mathbf{h}$: states of hidden units,
- generic state of a unit (either visible or hidden) by y_i, with $\mathbf{y} \equiv (\mathbf{x}, \mathbf{h})$,
- state of network when visible units are clamped in state $\mathbf{x}^{(n)}$ is $\mathbf{y}^{(n)} \equiv (\mathbf{x}^{(n)}, \mathbf{h})$.

Probability of $\mathbf{W}$ given a single pattern $\mathbf{x}^{(n)}$ is

$$P(\mathbf{x}^{(n)}|\mathbf{W}) = \sum_{\mathbf{h}} P(\mathbf{x}^{(n)}, \mathbf{h}|\mathbf{W}) = \sum_{\mathbf{h}} \frac{1}{Z(\mathbf{W})} \exp\left(\frac{1}{2}\mathbf{y}^{(n)^T}\mathbf{W}\mathbf{y}^{(n)}\right)$$

where

$$Z(\mathbf{W}) = \sum_{\mathbf{x},\mathbf{h}} \exp\left(\frac{1}{2}\mathbf{y}^T\mathbf{W}\mathbf{y}\right)$$

Boltzmann Machines with Hidden Units (cont.)

Applying the maximum likelihood method as before one obtains

$$\frac{\partial}{\partial w_{ij}} \ln P(\{\mathbf{x}^{(n)}\}_1^N | \mathbf{W}) = \sum_n \left[\underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}^{(n)}, \mathbf{W})}}_{\text{clamped to } \mathbf{x}^{(n)}} - \underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}, \mathbf{W})}}_{\text{free}} \right]$$

Term $\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}^{(n)}, \mathbf{W})}$ is the correlation between y_i and y_j when Boltzmann machine is simulated with visible variables clamped to $\mathbf{x}^{(n)}$ and hidden variables freely sampling from their conditional distribution.

Term $\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}, \mathbf{W})}$ is the correlation between y_i and y_j when the Boltzmann machine generates samples from its model distribution.

Boltzmann Machines with Input-Hidden-Output

The so far considered Boltzmann machine is a powerful stochastic Hopfield network with no ability to perform classification. Let us introduce visible input *and* output units:

- $\mathbf{x} \equiv (\mathbf{x}_i, \mathbf{x}_o)$

Note that pattern $\mathbf{x}^{(n)}$ consists of an input and output part, that is, $\mathbf{x}^{(n)} \equiv \left(\mathbf{x}_i^{(n)}, \mathbf{x}_o^{(n)}\right)$.

$$\sum_n \left[\underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}^{(n)}, \mathbf{W})}}_{\text{clamped to } (\mathbf{x}_i^{(n)}, \mathbf{x}_o^{(n)})} - \underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}, \mathbf{W})}}_{\text{clamped to } \mathbf{x}_i^{(n)}} \right]$$

Boltzmann Machines Updates Weights

Combine gradient descent and simulated annealing to update weights

$$\Delta w_{ij} = \frac{\eta}{T}\left[\underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x}^{(n)},\mathbf{W})}}_{\text{clamped to } (\mathbf{x}_i^{(n)}, \mathbf{x}_o^{(n)})} - \underbrace{\langle y_i y_j \rangle_{P(\mathbf{h}|\mathbf{x},\mathbf{W})}}_{\text{clamped to } \mathbf{x}_i^{(n)}}\right]$$

High computational complexity:

- present each pattern several times
- anneal several times

Mean-field version of Boltzmann learning:

- calculate approximations of the correlations ($[y_i y_j]$) entering the gradient

Deterministic Boltzmann Learning

input : $\{\mathbf{x}^{(n)}\}_1^N; \eta, T_{start}, T_{stop} \in \mathbb{R}$

output: $\mathbf{W}$

begin

  $T \leftarrow T_{start}$

  repeat

    randomly select pattern from sample $\{\mathbf{x}^{(n)}\}_1^N$

    randomize states

    anneal network with input and output clamped

    at final, low T, calculate $[y_i y_j]_{\mathbf{x}_i, \mathbf{x}_o}$ clamped

    randomize states

    anneal network with input clamped but output free

    at final, low T, calculate $[y_i y_j]_{\mathbf{x}_i}$ clamped

    $w_{ij} \leftarrow w_{ij} + \eta/T \left[[y_i y_j]_{\mathbf{x}_i, \mathbf{x}_o} \text{ clamped}] - [y_i y_j]_{\mathbf{x}_i} \text{ clamped} \right]$

  until $T < T_{stop}$

  return w

end