Lecture 8: Stochastic Methods

Andreas Wichert

Department of Computer Science and Engineering

Técnico Lisboa

Hopfield Model

- The Hopfield model represents a model of the associative memory. In a Hopfield model n units units are connected to each other by the weights w_{ij}, the dynamic of the network are represented by

$$x_i = sgn(net_i) = sgn\left(\sum_{j=1}^{n} w_{ij} \cdot x_j\right)$$

$$net_i = \sum_{j=1}^{n} w_{ij} \cdot x_j$$

$$sgn(net) = \begin{cases} 1 & \text{if} \quad net \geq 0 \\ -1 & \text{if} \quad net < 0 \end{cases} .$$

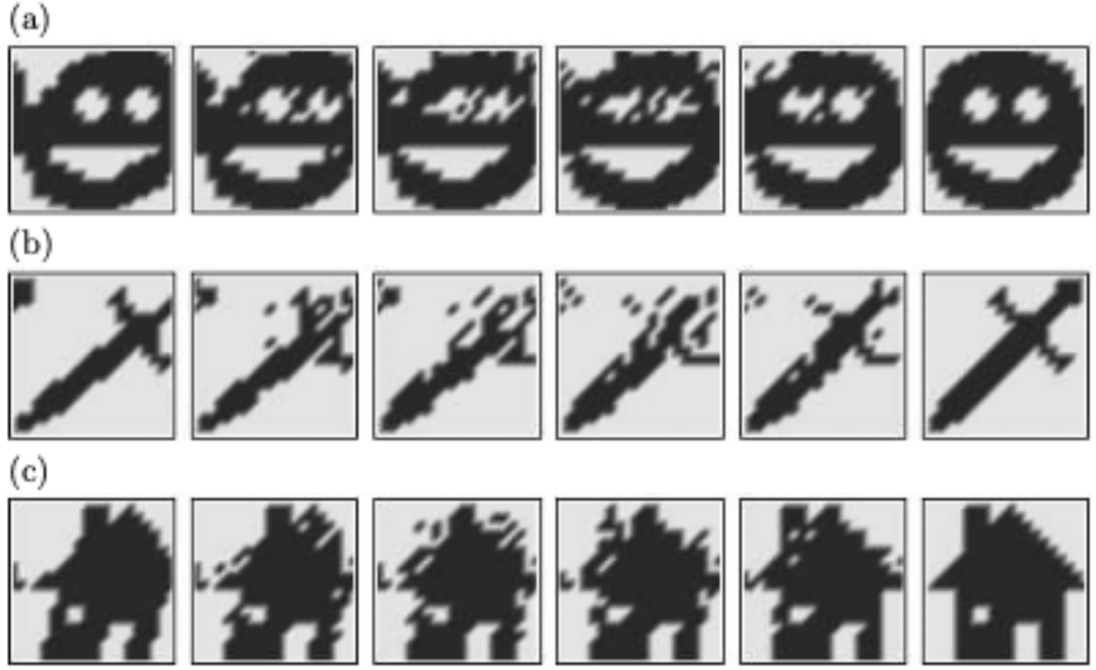
- Usually the units are updated asynchronously, updated them one at the time. For example at each time step, a random unit i is selected and updated
- A set of *N* binary patterns of dimension *n*

$$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \qquad\qquad x_{k,i} \in \{1, -1\}$$

is stored using the Hebbian learning rule with $w_{ij} = w_{ji}$ for all i,j and $w_{ii} = 0$ for all i

$$w_{ij} = \frac{1}{n} \cdot \sum_{k=1}^{N} x_{k,i} \cdot x_{k,j}$$

- Patterns that the network uses for training (called retrieval states) become attractors of the system.
- After the training we start with some configuration x_{query} and the network will converge after several steps using the update rule to an attractor



Hopfield Model

- Convergence is generally assured. The attractors are the stored patterns, however sometimes the network will converge to spurious patterns (different from the training patterns)

- The storage capacity of the model is

$$L = 0.138 \cdot n$$

Energy function

- One important property of the model is the idea of the energy function

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot x_i \cdot x_j .$$

  - The energy function describes the configuration of the system by a hilly "surface" In statistics and physics the energy function is called Hamiltonian.
    - The Hamiltonian always decrease (or remains constant) as the system evolves to its dynamical rule.
  - Attractors are the local minima of the Hamiltonian (energy function).
    - The dynamics of the system is similar to the motion of a particle on the hilly energy "surface" under the influence of gravity.

Ising Model

- We can describe a magnetic material by a set of atomic magnets arranged on a regular lattice that represents the crystal structure of the material
- We call the atomic magnets spins. A simple model of spin 1/2 atoms is the Ising model.
- The spins represented by a variable s_i can point only into two different distinct directions that is oriented up $s_i = 1$ and down $s_i = -1$.
- The Ising model is related to the Hopfield associative memory in which a unit can fire $x_i = 1$ or be inactive $x_i = -1$.

Ising Model

- The magnetic field consist of the internal field produced by the spins and the external field h_{ext} that influences the spins.
- The coefficient w_{ij} measures the strength of the influence of spin. The influence is symmetric $w_{ij} = w_{ji}$ resulting in magnetic field

$$h_i = \sum_{j=1}^{n} w_{ij} \cdot s_j + h^{ext}.$$

At low temperatures the spin is updated by

$$s_i = sgn(h_i) = sgn\left(\sum_{j=1}^{n} w_{ij} \cdot s_j + h^{ext}.\right).$$

Ising Model - Hamiltonian

- The updates are asynchronously or in random order. The energy function (Hamiltonian) is

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot s_i \cdot s_j - h^{ext} \cdot \sum_{i=1}^{n} s_i .$$

Spin Glass

- A spin glass is an Ising model without external field the model with the energy function (Hamiltonian)

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot s_i \cdot s_j .$$

- Only nearest neighbor interactions similar to the Ising model.
- The spin glass is named from the n analogy between the magnetic disorder in a spin glass and the positional disorder of a conventional window glass.
- At low temperature the model is the same as Hopfield model.

Finite Temperature Dynamics

- If the temperature is not very low thermal fluctuations tend to flip the spins randomly up and down.
- The thermal fluctuations become important at high temperature and decrease at low temperature and vanish at the absolute zero temperature.
- At high temperature the thermal fluctuations dominate the dynamics of the spin.

- The thermal fluctuations can be described by the Glauber dynamics that result in a stochastic rule

$$s_i = \begin{cases} +1 & \text{with probability } \; g(h_i) \\ -1 & \text{with probability } \; 1 - g(h_i). \end{cases}$$

- The “Glauber” choice is represented by the sigmoid function

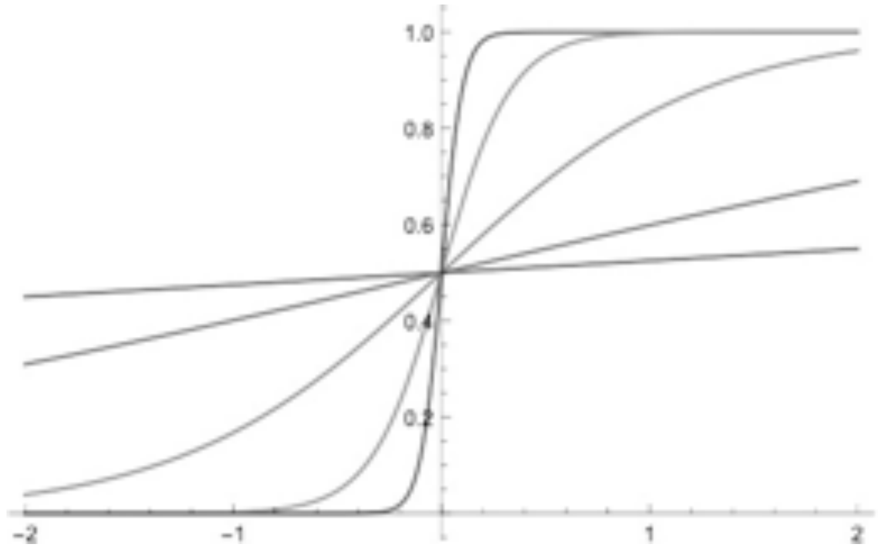
$$g(h) = \frac{1}{1 + e^{(-2\cdot\beta\cdot h)}} \qquad\qquad \beta = \frac{1}{k \cdot T}$$

- where T is the absolute temperature and k the Boltzmann’s constant

- The probability of spin s_i being *1* or *−1*

$$p(\pm s_i) = g(\pm h_i) = \frac{1}{1 + e^{(\mp 2 \cdot \beta \cdot h_i)}}$$

$$\beta = \frac{1}{k \cdot T}$$



- The temperature controls the steepness near h = 0

Boltzmann-Gibbs Distribution

- H_α denote the energy of the system when it is in state α. A fundamental result from the physics tells us that in thermal equilibrium each of the possible states α occurs with probability

$$p_\alpha = \frac{1}{Z} \cdot e^{\left(-\frac{H_\alpha}{k \cdot T}\right)}$$

- The partition function that is the sum over all states, with *Z* because the German name Zustadsumme.

$$Z = \sum_\alpha e^{\left(-\frac{H_\alpha}{k \cdot T}\right)} \qquad 1 = \sum_\alpha p_\alpha$$

- For a Hopfield network with probabilistic units and a bias bi with the resulting energy function

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot x_i \cdot x_j - \sum_{i=1}^{n} b_i \cdot x_i$$

- The partition function is over all possible stats, all *2^n* combinations of *$x_i = \pm 1$*

$$Z = \sum_{x_1=\pm 1} \sum_{x_2=\pm 1} \cdots \sum_{x_n=\pm 1} \exp\left(\frac{\beta}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot x_i \cdot x_j + \beta \cdot \sum_{i=1}^{n} b_i \cdot x_i \right)$$

- where *$\beta = 1/T$* since the temperature of a stochastic network is not related to physical temperature and we set *$k = 1$*

- The average activation of unit i is

$$\langle x_i \rangle = \frac{1}{Z} \cdot \sum_{x_1=\pm 1}^{n} \cdots \sum_{x_n=\pm 1}^{n} x_i \cdot \exp\left(\frac{\beta}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot x_i \cdot x_j + \beta \cdot \sum_{i=1}^{n} b_i \cdot x_i \right)$$

- Since the thermal average of the measured value of *A* is

$$\langle A \rangle = \sum_{\alpha} A_\alpha \cdot p_\alpha$$

- We have to compute two times sum over all combinations

- Knowing *Z* as a function of b_i we can simplify to

$$\langle x_i \rangle = \frac{1}{\beta \cdot Z} \cdot \frac{\partial Z}{\partial b_i} = T \cdot \frac{\partial}{\partial b_i} \cdot \log Z.$$

Stochastic Dynamics

- Transition probability of flipping from s_i to $-s_i$ are represented as

$$W(s_i \to -s_i) = \frac{1}{1 + e^{(\beta \cdot \Delta H_i)}}$$

$$\Delta H_i = H(s_1, s_2, \cdots, -s_i, \cdots s_n) - H(s_1, s_2, \cdots, s_i, \cdots s_n) = 2 \cdot h_i \cdot s_i$$

- In general the transition probabilities for the states *α* and *α′* are
$$W(\alpha \to \alpha').$$
- It is important to seek a condition on *W(α → α′)* to guarantee equilibrium
- A sufficient condition for a thermal equilibrium is that the average number of transitions from *α* to *α′* and from *α′* to *α* be equal
$$p_\alpha \cdot W(\alpha \to \alpha') = p'_\alpha \cdot W(\alpha' \to \alpha)$$
$$\frac{W(\alpha \to \alpha')}{W(\alpha' \to \alpha)} = \frac{p'_\alpha}{p_\alpha} = e^{(-\beta \cdot \Delta H)}$$

- This is called the principle of detailed balance

$$W(\alpha \to \alpha') = \frac{1}{1+e^{(\beta \cdot \Delta H)}}, \quad W(\alpha' \to \alpha) = \frac{1}{1+e^{(-\beta \cdot \Delta H)}}$$

with Boltzmann-Gibbs distribution

$$p_\alpha = \frac{1}{Z} \cdot e^{(-\beta H_\alpha)}, \quad p'_\alpha = \frac{1}{Z} \cdot e^{\left(-\beta H'_\alpha\right)}.$$

Gibs Sampling

- The stochastic dynamics of the Ising model can be described by Gibs sampling.
- Suppose that the system is in a state s and we have chosen an arbitrary coordinate i.
- We can then ignore the actual state of the spin si and ask for the conditional probability that this spin points upwards given all other spins.

$$h_i = \sum_{j=1, i \neq j}^{n} w_{ij} \cdot s_j + h^{ext}$$

with

$$p\left(s_i = 1 | \{s_j\}_{j \neq i}\right) = \frac{1}{1 + e^{(-2 \cdot \beta \cdot h_i)}}, \qquad \beta = \frac{1}{k \cdot T}.$$

A single Gibbs sampling step now proceeds as follows:

- Randomly pick a coordinate i.
- Calculate the conditional probability $p = p\left(s_i = 1 | \{s_j\}_{j \neq i}\right)$.
- Draw a real number $\xi \in [0, 1]$ from the uniform distribution.
- If ξ is at most equal to p, set the spin at position i to $+1$, otherwise set it to -1.

- The algorithm then starts with a randomly chosen state and subsequently applies a large number of Gibbs sampling steps.
- After some time, called the burn-in time, the states after each step then form the sample we are looking for represented by he equilibrium distribution.

Markov chain Monte Carlo algorithm

- Gibbs sampling is a simple and widely applicable Markov chain Monte Carlo algorithm.
- It generates a Markov chain with the Gibbs distribution as the equilibrium distribution since during the evolution none of the conditional distributions be anywhere zero

- Consider the distribution $p(\mathbf{x}) = p(x_1, x_2, \cdots, x_n)$

  from which we want to sample.
- We have chosen some initial state for the Markov chain.
- Each step of the Gibbs sampling procedure involves replacing the value of one of the variables by a value drawn from the distribution of that variable conditioned on the values of the remaining variables.

Gibbs Algorithm:

- Initialise

$$x_1, x_2, \cdots, x_n$$

- For $t = 1, 2, \cdots, T$:
  - sample $x_1^{(t+1)}$ and replace $x_1^{(t)}$

$$x_1^{(t+1)} \sim p(x_1 | x_2^{(t)}, x_3^{(t)}, \cdots, x_M^{(t)}),$$

  - sample $x_2^{(t+1)}$ and replace $x_2^{(t)}$

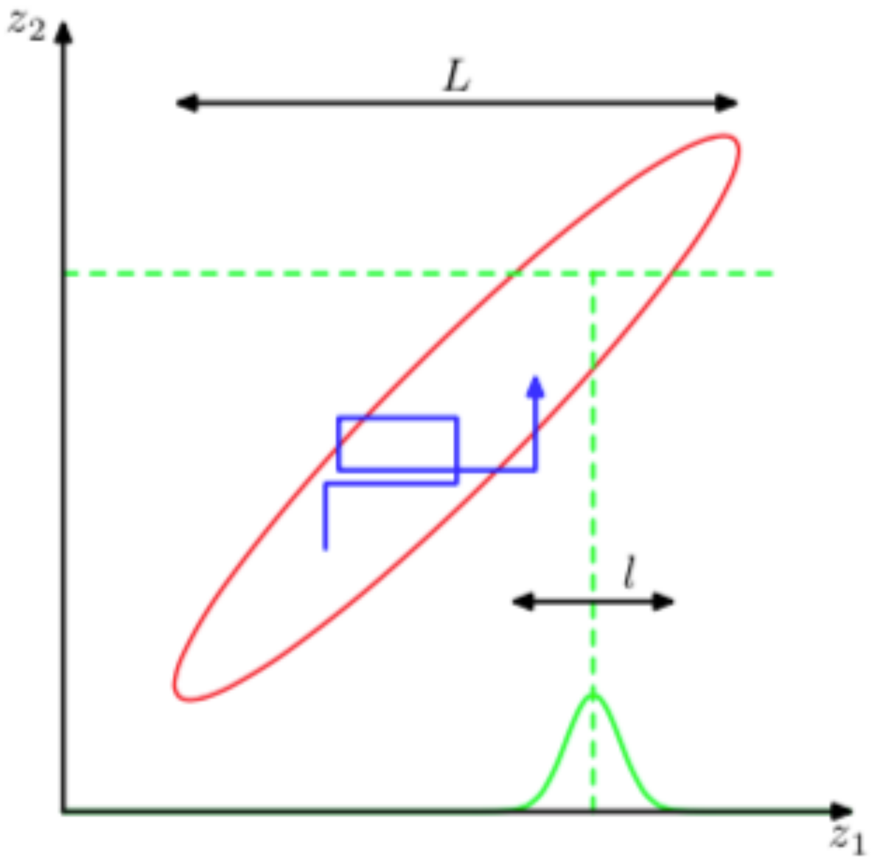
$$x_2^{(t+1)} \sim p(x_2 | x_1^{(t+1)}, x_3^{(t)}, \cdots, x_M^{(t)}),$$

  - ...

$$\cdots\cdots\cdots,$$

  - sample $x_M^{(t+1)}$ and replace $x_M^{(t)}$

$$x_M^{(t+1)} \sim p(x_M j | x_1^{(t+1)}, x_2^{(t+1)}, x_3^{(t+1)}, \cdots, x_{M-1}^{(t-1)},)$$


z_2
L
l
z_1

Metropolis Algorithm

- Metropolis Algorithm is a modified Monte Carlo method, it is referred to as a Markov chain Monte Carlo.
- Metropolis proposed the following stochastic matrix that is composed on transition probabilities that describe the transition of states *W(α → α')* with

$$T_{ij} \geq 0, \quad T_{ij} = T_{ji}, \quad \sum_i T_{ij} = 1$$

and is defined as

$$P_{ij} = \begin{cases} \begin{cases} T_{ij} & if \quad p_i/p_j \geq 1 \\ T_{ij} \cdot p_i/p_j & if \quad p_i/p_j < 1 \end{cases} & i \neq j \\ T_{jj} & i = j \end{cases}$$

- The detailed balance holds for all i and j

A sufficient condition for a thermal equilibrium is that the average number of transitions from i to j and from j to be equal with

$$P_{ij} \cdot p_j = (T_{ij} \cdot p_i/p_j) \cdot P_j = (T_{ji} \cdot p_j/p_i) \cdot P_i = P_{ij} \cdot p_i.$$

- With $\beta = \dfrac{1}{k \cdot T}$

and Boltzmann-Gibbs distribution

$$p_i = \frac{1}{Z} \cdot e^{(-\beta H_i)}, \quad p_j = \frac{1}{Z} \cdot e^{(-\beta H_j)}$$

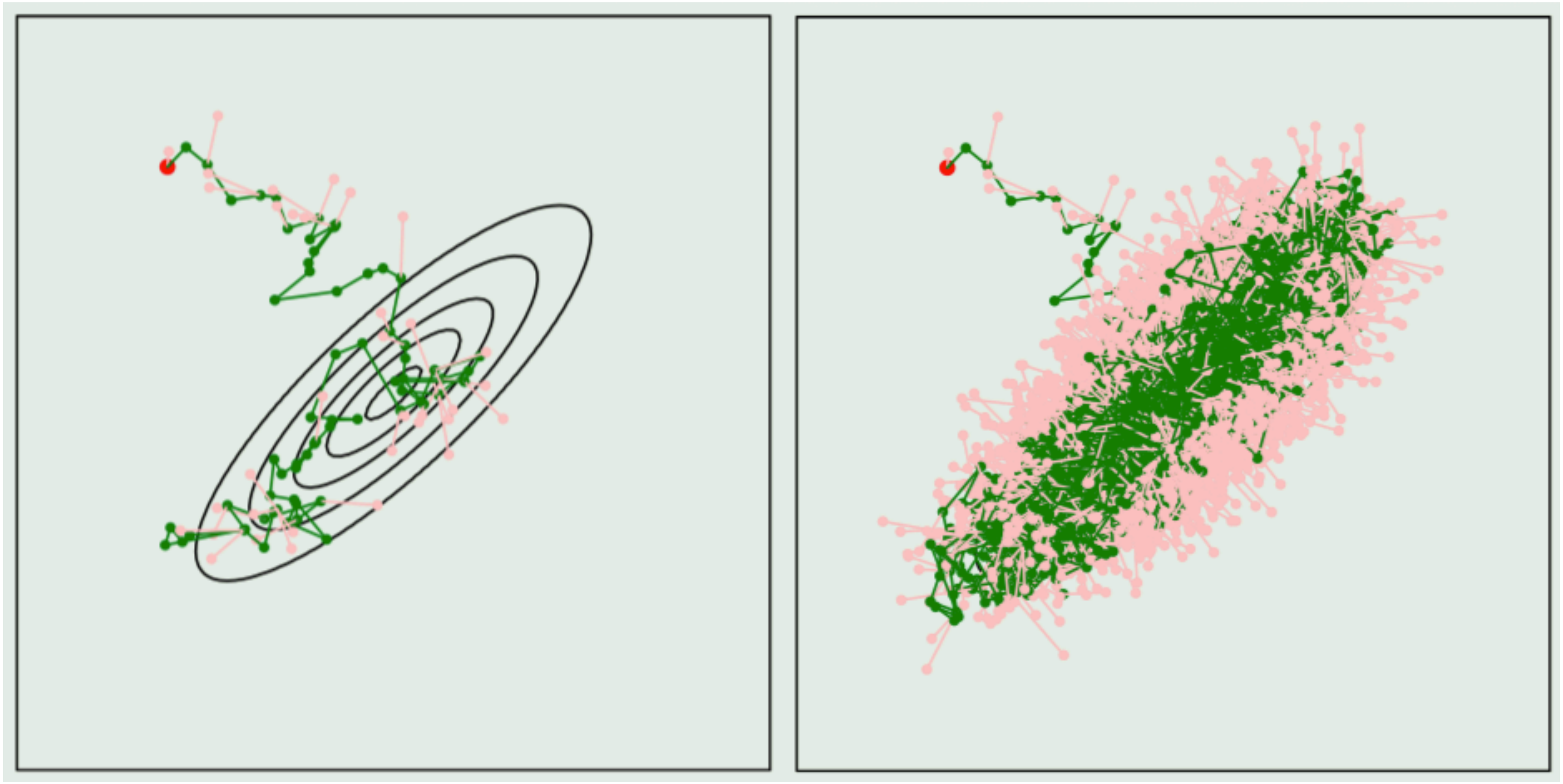
we get

$$P_{ij} = \begin{cases} \begin{cases} T_{ij} & if \quad e^{(-\beta \cdot \Delta H)} \geq 1 \\ T_{ij} \cdot e^{(-\beta \cdot \Delta H)} & if \quad e^{(-\beta \cdot \Delta H)} < 1 \end{cases} & i \neq j \\ T_{jj} & i = j \end{cases}$$

with

$$\Delta H = H_i - H_j.$$

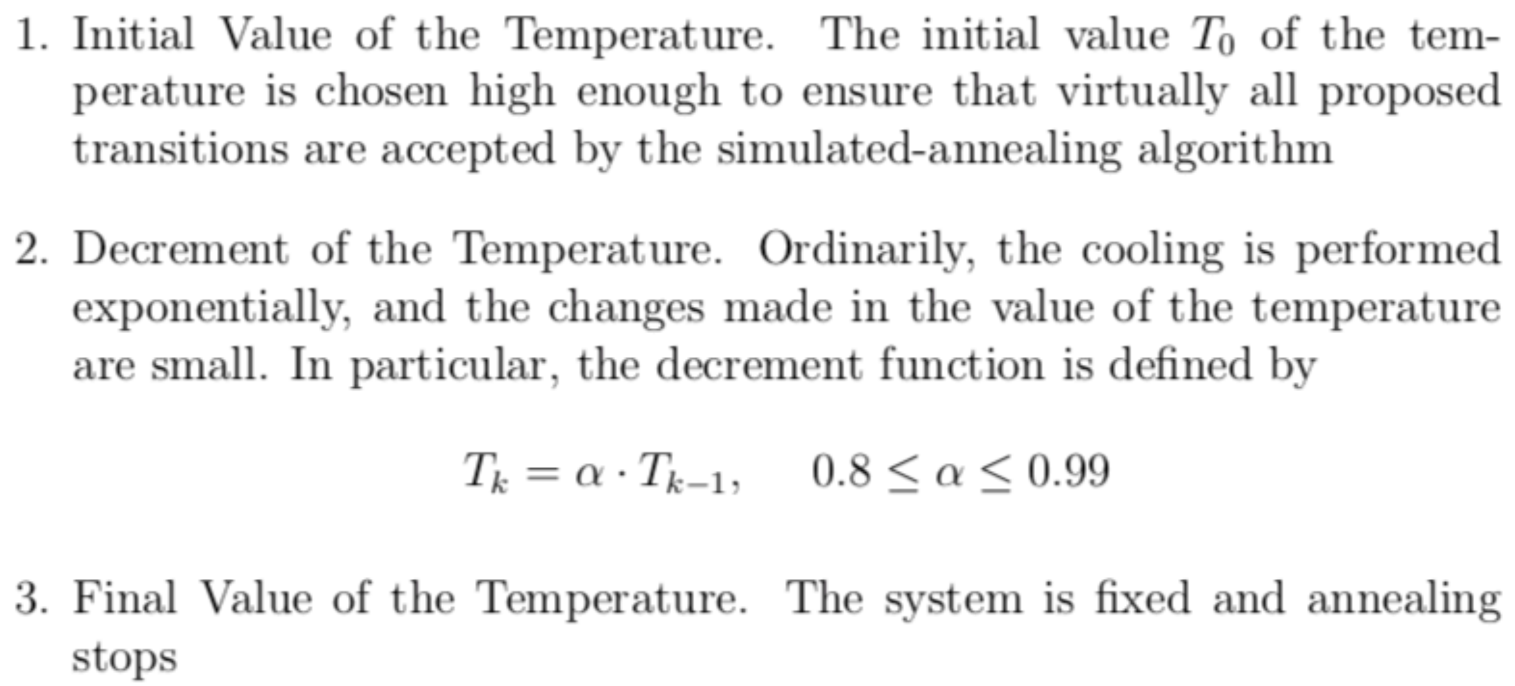
- The Metropolis algorithm states to accept a transition if the new configuration has a lower energy.
- Otherwise the algorithm says to accept the change that increase the energy but only with the probability $e^{(-\beta \cdot \Delta H)}$.



Simulated Annealing

- Rather, the preferred method for improved computational efficiency is to operate the stochastic system at a high temperature where convergence to equilibrium is fast, and then maintain the system at equilibrium as the temperature is slowly lowered
- The Metropolis algorithm is the basis for the simulated-annealing process, in the course of which the temperature is decreased slowly
- The simulated-annealing process will converge to a configuration of minimal energy provided that the temperature is decreased no faster than logarithmically

Simulated Annealing - Schedule



1. Initial Value of the Temperature. The initial value T_0 of the temperature is chosen high enough to ensure that virtually all proposed transitions are accepted by the simulated-annealing algorithm

2. Decrement of the Temperature. Ordinarily, the cooling is performed exponentially, and the changes made in the value of the temperature are small. In particular, the decrement function is defined by

$$T_k = \alpha \cdot T_{k-1}, \quad 0.8 \leq \alpha \leq 0.99$$

3. Final Value of the Temperature. The system is fixed and annealing stops

Combinatorial Optimization

Simulated annealing is particularly well suited for solving combinatorial-optimization problems.

The objective of combinatorial optimization is to minimize the cost function of a finite, discrete system characterized by a large number of possible solutions.

Simulated annealing uses the Metropolis algorithm to generate a sequence of solutions by invoking an analogy between a physical many-particle system and a combinatorial-optimization problem.

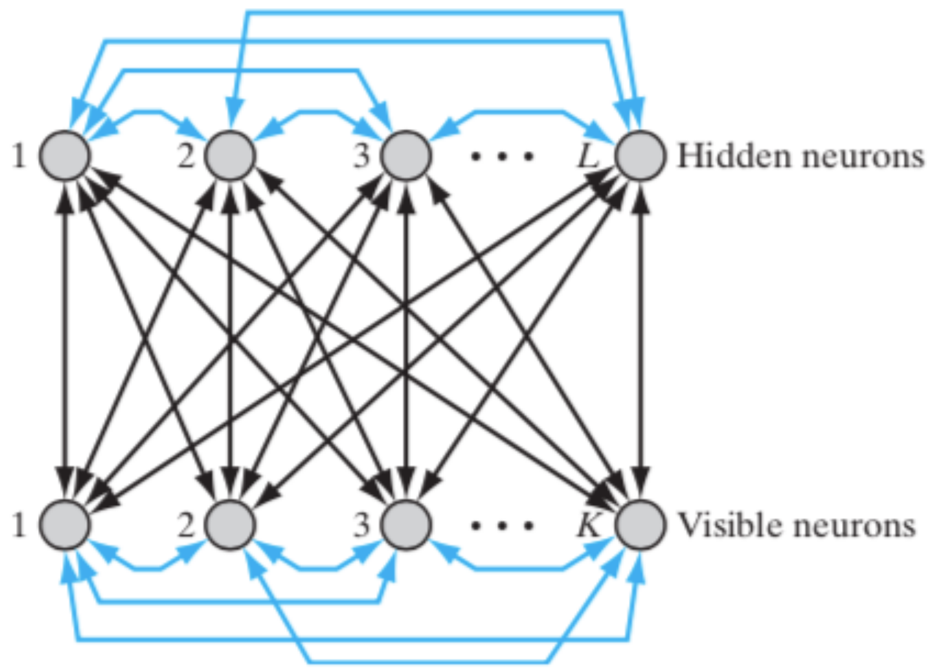
Statistical physics	Combinatorial optimization
Sample	Problem instance
State (configuration)	Configuration
Energy	Cost function
Temperature	Control parameter
Ground-state energy	Minimal cost
Ground-state configuration	Optimal configuration

Boltzmann Machine

- The Boltzmann machine is a stochastic Hopfield network with hidden units.
- It is a stochastic binary machine whose composition consists of stochastic neurons in state *1* or state *0* (or *1* and −*1*).
- Boltzmann machine is the uses symmetric synaptic connections between its neurons. The synaptic connection form neuron *i* to *j* are

- The stochastic neurons of the Boltzmann machine are divided into visible and hidden neurons.
- Visible neurons provide an interface between the network and the environment in which it operates.
- When a partial information-bearing vector is clamped onto a subset of the visible neurons, the network performs completion on the remaining visible neurons.
- By so doing, the network can perform pattern completion.

Boltzman Machine



- Assuming the network is composed of n units. Usually the units are updated asynchronously, updated them one at the time.
  - For example at each time step, a random unit i is selected and updated with

$$net_i = \sum_{j=1}^{n} w_{ij} \cdot x_j + b_i = \sum_{j=1, j \neq i}^{n} w_{ij} \cdot x_j + b_i$$

- with *b*$_i$ being the bias. Unit i then turns on with a probability given by the sigmoid (logistic) function

$$p(x_i = 1) = \frac{1}{1 + e^{(-\beta \cdot net_i)}}, \quad \beta = \frac{1}{T}.$$

Stochastic Dynamics

- The stochastic dynamics of Boltzmann machine can be described by Gibs sampling.
- Suppose that the system is in a state x and we have chosen an arbitrary coordinate i.
- We can then ignore the actual state of the unit xi and ask for the conditional probability

$$p\left(x_i = 1 \middle| \{x_j\}_{j \neq i}\right) = \frac{1}{1 + e^{(-\beta \cdot net_i)}}.$$

A single Gibbs sampling step now proceeds as follows:

- Randomly pick a coordinate i.
- Calculate the conditional probability $p = p\,(x_i = 1|\{x_j\}_{j\neq i})$.
- Draw a real number $\xi \in [0, 1]$ from the uniform distribution.
- If ξ is at most equal to p, set the spin at position i to $+1$, otherwise set it to -1.

Provided that the stochastic simulation is performed long enough, the network will reach thermal equilibrium at temperature T. The time taken to reach thermal equilibrium can be much too long. To overcome this difficulty, simulated annealing is used.

Learning

- During the training phase of the network there are two phases to the operation of the Boltzmann machine:
  - (1) Positive phase. In this phase, the network operates in its clamped con- dition under the direct influence of the training sample. The visible neurons are all clamped onto specific states determined by the environ- ment.

  - (2) Negative phase. In this second phase, the network is allowed to run freely, and therefore with no environmental input.The states of the units are determined randomly. The probability of finding it in any particular global state depends on the energy function

A set of N binary patterns of dimension v

$$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$$

with (k identifies the training pattern and i the dimension)

$$x_{k,i} \in \{1, 0\}$$

A learning is preformed over the whole training set of N vectors. Such learning is called the batch learning. It is often more efficient to learn over a sub sample of the training set, in this case the learning is called mini-batch usually with 10 to 100 training patterns.

- Initialize the weights and the bias with random numbers. With small random values chosen from a from a zero-mean Gaussian with a standard deviation of about 0.01.

- For $\tau = 1, 2, \cdots, \tau^{end}$:
  - Initialise:

    $$pp_{ij} = 0, \quad pp_{i0} = 0, \quad pn_{ij} = 0, \quad pn_{i0} = 0$$

  - For $k = 1, 2, \cdots, N$:
    * **Positive phase:** At starting state the visible units are in state representing a training pattern $\mathbf{x}_k$ and the hidden units are chosen random (0 or 1). Preform Gibbs sampling until a thermal equilibrium is reached. Store the statistics for the visible phase according the Hebbian learning rule:

      $$pp_{ij}^{new} = \begin{cases} pp_{ij}^{old} + 1 & if \quad x_i \cdot x_j = 1 \\ pp_{ij}^{old} & \text{otherwise.} \end{cases} \tag{14.57}$$

      and for the bias

      $$pp_{i0}^{new} = \begin{cases} pp_{i0}^{old} + 1 & if \quad x_i = 1 \\ pp_{i0}^{old} & \text{otherwise.} \end{cases} \tag{14.58}$$

* **Negative phase:** At starting state the visible units are chosen random and the hidden units are chosen random (0 or 1). Preform Gibbs sampling until a thermal equilibrium is reached. Store the statistics for the negative phase according the anti Hebbian learning rule:

$$pn_{ij}^{new} = \begin{cases} pn_{ij}^{old} + 1 & if \quad x_i \cdot x_j = 1 \\ pn_{ij}^{old} & \text{otherwise.} \end{cases} \tag{14.59}$$

and for the bias

$$pn_{i0}^{new} = \begin{cases} pn_{i0}^{old} + 1 & if \quad x_i = 1 \\ pn_{i0}^{old} & \text{otherwise.} \end{cases} \tag{14.60}$$

Negative phase can be identified with unlearning and it was suggested that such reverse learning appears in REM sleep (Crick1983).

– Normalize

$$\overline{pp}_{ij}^{new} = \frac{pp_{ij}^{old}}{N}, \quad \overline{pp}_{i0}^{new} = \frac{pp_{i0}^{old}}{N},$$

$$\overline{pn}_{ij}^{new} = \frac{pn_{ij}^{old}}{N}, \quad \overline{pn}_{i0}^{new} = \frac{pn_{i0}^{old}}{N}$$

Adapt the weights and the bias

$$\Delta w_{ij} = \eta \cdot (\overline{pp}_{ij} - \overline{pn}_{ij})$$

and the bias

$$\Delta b_i = \eta \cdot (\overline{pp}_{i0} - \overline{pn}_{i0})$$

Proof of Learning in Boltzmann Machine

Energy of a Boltzmann machine is

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \cdot x_i \cdot x_j - \sum_{i=1}^{n} b_i \cdot x_i \tag{14.63}$$

or simplified assuming bias is equal to w_{i0} and $x_0 = 1$

$$H = -\frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{j=0}^{n} w_{ij} \cdot x_i \cdot x_j \tag{14.64}$$

is the same as for a Hopfield network with bias b_i. Usually the energy function has many local minima. If no hidden units are present, then the learning is related to of the for a Hopfield network

$$w_{ij} = \frac{1}{n} \cdot \sum_{k=1}^{N} x_{k,i} \cdot x_{k,j} \tag{14.65}$$

we divide by n and not N, since its storage capacity is lower than N, since N can be maximally $L = 0.138 \cdot n$.

A state of the Boltzmann machine is described by the random variable X, we preform a partition into visible sates V and hidden states H. Assuming the training sample is statistically independent with possible values $\mathbf{v}_k$, the overall probability is

$$p(V) = \prod_{k=1}^{N} p(\mathbf{v}_k) \tag{14.66}$$

Maximizing the likelihood (ML) is dependent on $\mathbf{w}$ is

$$\mathbf{w}_{ML} = \arg\max_{\mathbf{w}} p(V|\mathbf{w})) = \arg\max_{\mathbf{w}} \log(p(V) \tag{14.67}$$

since log is monotonic increasing function

$$\mathbf{w}_{ML} = \arg\max_{\mathbf{w}} \log\left(\prod_{k=1}^{N} p(\mathbf{v}_k)\right) = \arg\max_{\mathbf{w}} \sum_{k=1}^{N} (\log p(\mathbf{v}_k)) \tag{14.68}$$

and we can define a $L(\mathbf{w})$

$$L(\mathbf{w}) = \sum_{k=1}^{N} (\log p(\mathbf{v}_k)) \tag{14.69}$$

By the law of total probability

$$p(\mathbf{v}_k) = \sum_{\mathbf{h}} p(\mathbf{v}_k, \mathbf{h}) = \frac{1}{Z} \cdot \sum_{\mathbf{h}} e^{\left(-\frac{H(\mathbf{v}_k, \mathbf{h})}{T}\right)} \tag{14.70}$$

and

$$Z = \sum_{\mathbf{x}} e^{\left(-\frac{H(\mathbf{x})}{T}\right)} \tag{14.71}$$

and we get

$$L(\mathbf{w}) = \sum_{k=1}^{N} \left(\log \left(\frac{1}{Z} \cdot \sum_{\mathbf{h}} e^{\left(-\frac{H(\mathbf{v}_k, \mathbf{h})}{T}\right)} \right) \right) \tag{14.72}$$

$$L(\mathbf{w}) = \sum_{k=1}^{N} \left(\log \left(\sum_{\mathbf{h}} e^{\left(-\frac{H(\mathbf{v}_k, \mathbf{h})}{T}\right)} \right) - \log \left(\sum_{\mathbf{x}} e^{\left(-\frac{H(\mathbf{x})}{T}\right)} \right) \right) \tag{14.73}$$

differentiation $L(\mathbf{w})$

$$\frac{\partial L(\mathbf{w})}{\partial w_{ij}} = \sum_{k=1}^{N} \left(\frac{\partial}{\partial w_{ij}} \log \left(\sum_{\mathbf{h}} e^{\left(-\frac{H\left(\mathbf{v}_k,\mathbf{h}\right)}{T}\right)} \right) - \frac{\partial}{\partial w_{ij}} \log \left(\sum_{\mathbf{x}} e^{\left(-\frac{H(\mathbf{x})}{T}\right)} \right) \right)$$

$$= \frac{1}{T} \cdot \sum_{k=1}^{N} \left(\frac{\left(\sum_{\mathbf{h}} e^{\left(-\frac{H\left(\mathbf{v}_k,\mathbf{h}\right)}{T}\right)} \cdot \frac{\partial H(\mathbf{v}_k,\mathbf{h})}{\partial w_{ij}} \right)}{\sum_{\mathbf{h}} e^{\left(-\frac{H\left(\mathbf{v}_k,\mathbf{h}\right)}{T}\right)}} - \frac{\left(\sum_{\mathbf{x}} e^{\left(-\frac{H(\mathbf{x})}{T}\right)} \cdot \frac{\partial H(\mathbf{x})}{\partial w_{ij}} \right)}{\sum_{\mathbf{x}} e^{\left(-\frac{H(\mathbf{x})}{T}\right)}} \right) \tag{14.75}$$

and we can simplify to

$$\frac{\partial L(\mathbf{w})}{\partial w_{ij}} = \frac{1}{T} \cdot \sum_{k=1}^{N} \left(\sum_{\mathbf{h}} p(\mathbf{v}_k, \mathbf{h}) \cdot x_i \cdot x_j - \sum_{\mathbf{x}} p(\mathbf{x}) \cdot x_i \cdot x_j \right). \tag{14.76}$$

We introduce two definitions

(1)

$$\langle x_i x_j \rangle_{data} = \sum_{k=1}^{N} \sum_{\mathbf{h}} p(\mathbf{v}_k, \mathbf{h}) \cdot x_i \cdot x_j$$

correlation when the network is clamped

(2)

$$\langle x_i x_j \rangle_{model} = \sum_{k=1}^{N} \sum_{\mathbf{x}} p(\mathbf{x}) \cdot x_i \cdot x_j$$

and

$$\frac{\partial L(\mathbf{w})}{\partial w_{ij}} = \frac{1}{T} \cdot (\langle x_i x_j \rangle_{data} - \langle x_i x_j \rangle_{model})$$

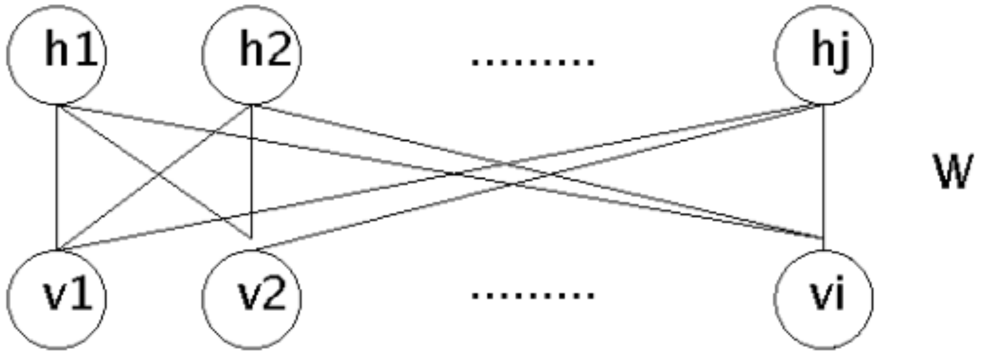
and the gradient ascent rule is called the Boltzmann learning rule:

$$\Delta w_{ji} = \epsilon \cdot \frac{\partial L(\mathbf{w})}{\partial w_{ij}} = \eta \cdot (\langle x_i x_j \rangle_{data} - \langle x_i x_j \rangle_{model})$$

with $\eta = \frac{\epsilon}{T}$

Harmonium - Restricted Boltzmann Machine

- A re- stricted Boltzmann machine (RBM) has only connections between visible and hidden units to make inference and learning easier.
- It was initially invented under the name Harmonium by Paul Smolensky in 1986



- Energy of a restricted Boltzmann machine can rewritten

$$H(\mathbf{v},\mathbf{h}) = -\sum_{i=1}^{H}\sum_{j=1}^{V} w_{ij}\cdot h_i \cdot v_j - \sum_{i=1}^{n} b_i \cdot h_i - \sum_{j=1}^{n} c_j \cdot v_j.$$

since there are no connections between units hi · hj or vi · vj

- Due to the constraints an RBM it only takes one step to reach thermal equilibrium when the visible units are clamped and the the hidden units are chosen random (0 or 1)

$$\langle v_i h_j \rangle_{data}.$$

- During the negative phase the units are chosen random random (0 or 1) one preforms Gibbs sampling until a thermal equilibrium is reached

$$\langle v_i h_j \rangle_{model}$$

and one preforms the update rule

$$\Delta w_{ij} = \eta \cdot (\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model})$$

Contrast Divergence

- To avoid the long computing time of $\langle v_i h_j \rangle$ model contrastive divergence (CD) based learning was proposed. It does not directly use Gibbs sampling
- Starting at random chosen units. Instead the training starts with a clamped vector $\mathbf{v}^0$ on the visible units and the hidden layer values $\mathbf{h}^0$ are computed with

$$p(h_i = 1|v) = \frac{1}{1 + e^{-(\sum_{j=1}^{n} w_{ij} v_j + c_i)}},$$

that define

$$\langle v_i h_j \rangle^0_{data}.$$

Then a reconstruction of the visible layer $\mathbf{v}^1$ is computed with

$$p(v_i = 1|h) = \frac{1}{1 + e^{-(\sum_{j=1}^{n} w_{ij} h_j + b_i)}}, \tag{14.83}$$

with the clamped reconstructed vector $\mathbf{v}^1$ on the visible units and the hidden layer values $\mathbf{h}^1$ are computed using the Equation 14.81 again.

$$\langle v_i h_j \rangle^1_{recon}. \tag{14.84}$$

One alternate between updating all the hidden units in parallel and updating all the visible units in parallel repeating τ steps resulting in

$$\Delta w_{ij} = \epsilon \cdot (\langle v_i h_j \rangle^0_{data} - \langle v_i h_j \rangle^\tau_{recon}) \tag{14.85}$$

One alternate between updating all the hidden units in parallel and updating all the visible units in parallel repeating τ steps resulting in

$$\Delta w_{ij} = \epsilon \cdot (\langle v_i h_j \rangle^0_{data} - \langle v_i h_j \rangle^{\tau}_{recon}) \tag{14.85}$$

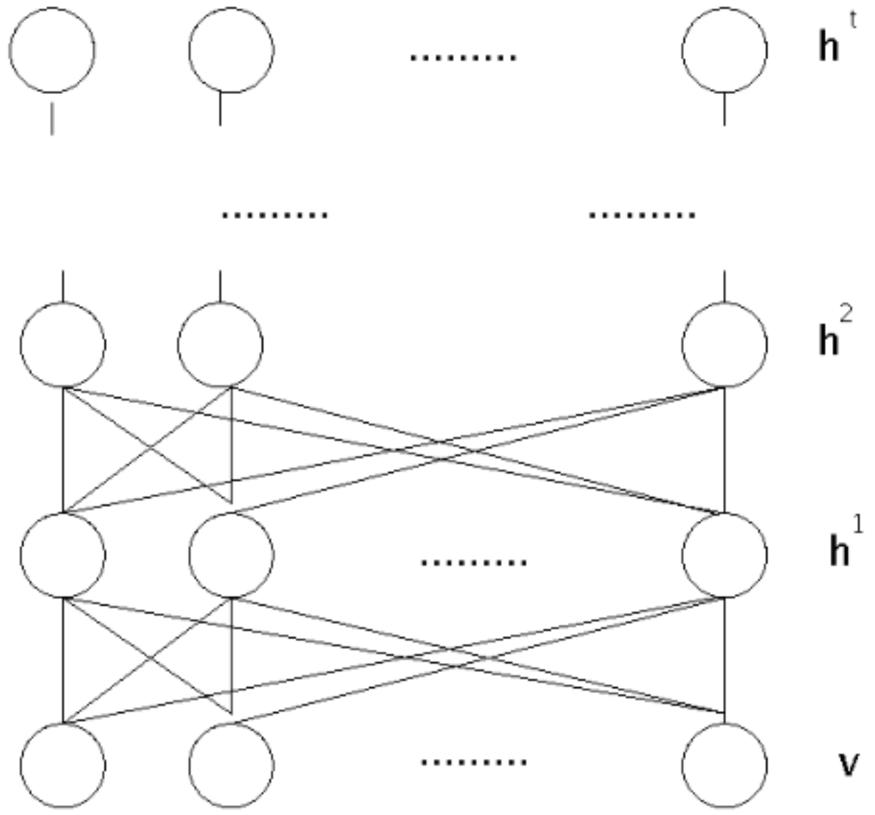
It was shown (19) that if the process of forward sampling and reconstruction were repeated τ times, then (10) would converge to the true gradient as $\tau \to \infty$ although in practice the updates are computed using just a single-step reconstruction with $\tau = 1$. The single-step reconstruction wit $\tau = 1$ is not following the gradient of the log likelihood, sometimes CD may take many iterations with $1 \ll \tau$ to converge.

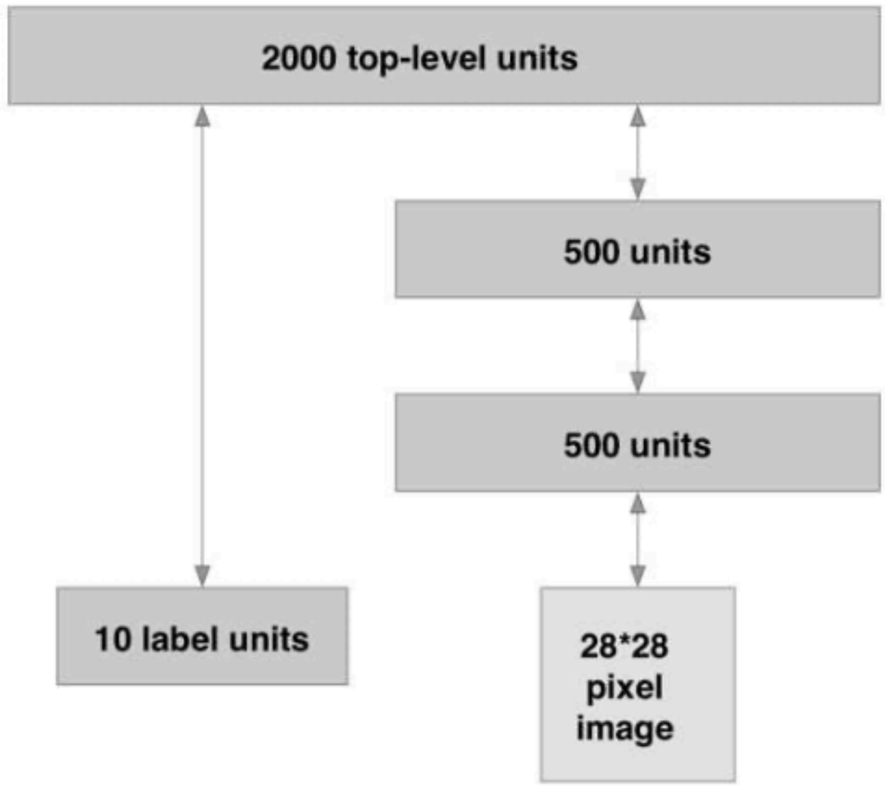
Deep Learning with Deep Belief Nets

- A deep belief network (DBN) is a generative graphical model, or alter-natively a class of deep neural network, composed of multiple layers of latent variables (hidden units), with connections between the layers but not between units within each layer.
- It can be viewed as a composition of restricted Boltzmann machines Deep Belief Networks (Hinton, 2006(Capture higher-level representations of input features.
- Around 2012, DBNs get renamed as Deep Neural Networks (DNNs)

- A restricted Boltzmann machine (RBM) is trained directly on the input data, thereby making it possible for the stochastic neurons in the hidden layer of the RBM to capture the important features that characterize the input data.
- Then treat the activations of the trained features as if they were input features and learn features of features in a second hidden layer. The activations of the trained features are then used as input data to train a second RBM that is trained.
- The process of learning features of features is continued until a number t of hidden layers, until t RBMs have been trained

A deep belief network





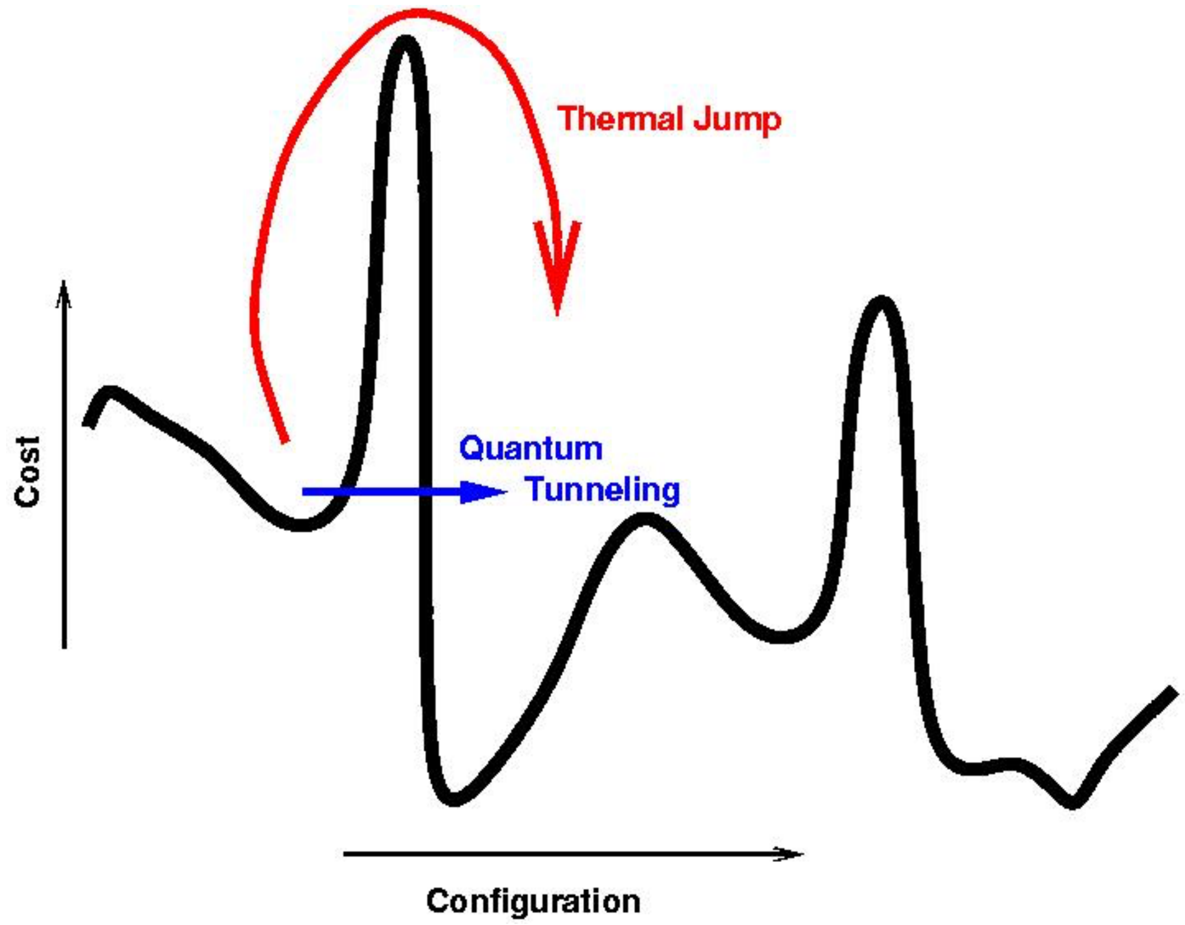
- The model learned to to generate combinations of labels and images.
- To perform recognition we start with a random state of the label units and clamp the input image.
- Then we do up-pass from the image followed by a few iterations of the top-level layers.

- The deep belief network learns to disregard irrelevant features while simultaneously learning relevant features.
- Deep belief nets provide a great deal of freedom.
- An efficient sampling method as proposed by future quantum annealers could extend this a creativity to yet unknown frontiers.

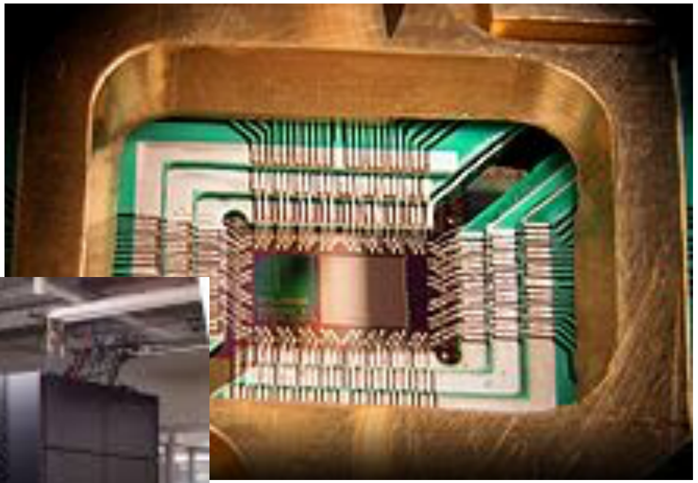
Quantum Annealing

- In quantum annealing, the quantum fluctuation parameter replaces a local minimum state with a randomly selected neighboring state in some fixed radius.
- The neighborhood extends over the whole search space at the beginning, and then, it is slowly reduced until the neighborhood shrinks to those few states that differ minimally from the current states.
- In a quantum system, the quantum fluctuation can be performed directly by an adiabatic process rather than needing to be simulated.
- These processes are based on quantum tunneling.

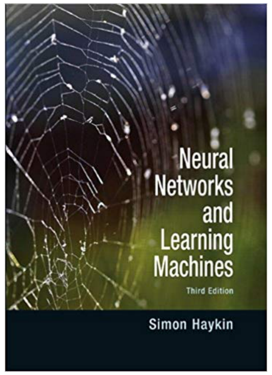
- Quantum tunneling is based on the Heisenberg uncertainty principle, as shown, and the wave-particle duality of matter represented by the wave propagation.
- Quantum annealing can speed up some machine learning tasks
- It is an alternative to the simulated annealing that is used in the learning and optimization tasks.
- Adiabatic quantum computers based like D-Wave are based on quantum annealing


Thermal Jump
Quantum
Tunneling
Cost
Configuration

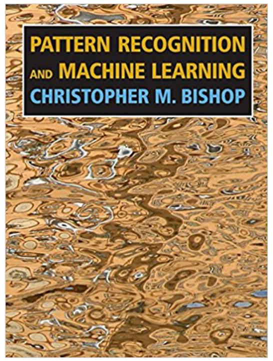
D-Wave



Literature

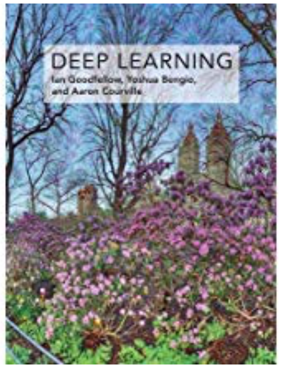


- Simon O. Haykin, Neural Networks and Learning Machine, (3rd Edition), Pearson 2008
  - Chapter 11



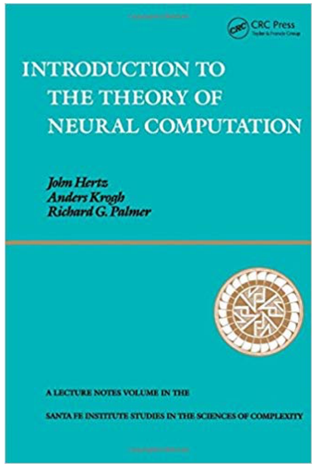
- Christopher M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer 2006
  - Chapter 11

Literature



- Deep Learning, I. Goodfellow, Y. Bengio, A. Courville MIT Press 2016
  - Chapter 17, 20

Literature (Additional)


CRC Press
INTRODUCTION TO THE THEORY OF NEURAL COMPUTATION
John Hertz
Anders Krogh
Richard G. Palmer
A LECTURE NOTES VOLUME IN THE SANTA FE INSTITUTE STUDIES IN THE SCIENCES OF COMPLEXITY

- *Introduction To The Theory Of Neural Computation (Santa Fe Institute Series Book 1), John A. Hertz, Anders S. Krogh, Richard G. Palmer, Addison-Wesley Pub. Co, Redwood City, CA; 1 edition (January 1, 1991)*
  - *Chapter 7*