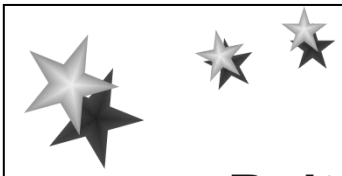


# Boltzmann Machines

## Simulated Annealing

- Key in reviving interest in neural nets
  - circa early 80s
  - A machine - wow!
- Annealing
  - A form of tempering or strengthening alloys
  - Simulated annealing means tempering a neural net
  - Significantly slower to learn and use than BP



Historically, work on Boltzmann machines was key in reviving interest in neural networks. Even the use of the word, "machine" played a part in attracting attention from the scientific community.

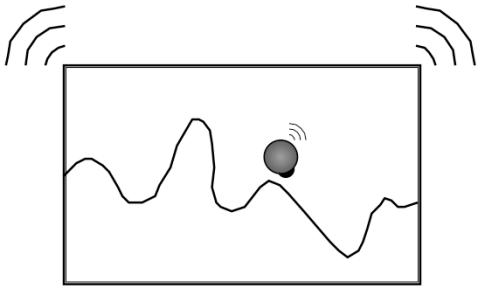
Boltzmann learning is based upon a technique known as *simulated annealing* and compared to other learning algorithms such as backpropagation, is significantly slower.

The learning technique central to Boltzmann machines is referred to as simulated annealing. In physics, annealing is a way of tempering certain alloys by heating them and letting them gradually cool. Typically, annealing causes the arrangements of molecules in the metal to settle into a low energy state, which in alloys, means forming a well ordered crystalline structure. In turn this strengthens the alloy.

At least one company (see http://www.attrasoft.com/company_overview.asp) was recently (1999) selling a Boltzmann machine for solving pattern recognition problems.

## Simulated Annealing cont.

- Introduces notion of temperature
- The hotter, the more the box is shaking





Something similar to the tempering of alloys happens in simulated annealing. Consider the problem of reaching a global energy minimum. If we take a box containing a landscape and a marble, we could start by shaking the box vigorously at first and then gradually reducing the amount of shaking. The idea is that at any one period, there should be just enough energy in the box to shake the marble out of all but the deepest "well".

## Versatile

- Very Versatile
  - Boltzmann machines can be applied to many types of problems.
  - Supervised, unsupervised, associative ...



The Boltzmann Machine is a highly recurrent and highly versatile neural network. It can be applied to practically all supervised, unsupervised, or pattern completion problems.

# Versatility cont.

- Constraint Satisfaction
  - Good for problems involving many degrees of freedom
    - (e.g. Airline crew scheduling)
- Regular supervised learning
- Pattern completion
  - viz. Hopfield Net.

## Characteristics

- Training philosophy
  - Based upon gradient descent (viz. BP)
  - Kirkpatrick et al. (1983) in reference to gradient descent:

> "*always* going downhill, try going downhill *most of the time.*"



One of the problems cited with gradient descent is that of falling into local minima. This has been identified as a deficiency of backpropagation and its derivatives. The training philosophy adopted in Boltzmann learning although based upon gradient descent, is theoretically more likely to reach a global minimum than in BP. It's able to do this because the machine also accommodates gradient ascent as well, at least for some of the time.

## Characteristics

- Temperature introduces randomness
  - "*synthetic temperature*"
- Randomness means
  - Can "jump" out of local minima
  - Gross details encountered at high temperatures.
  - Fine details at lower temperatures.



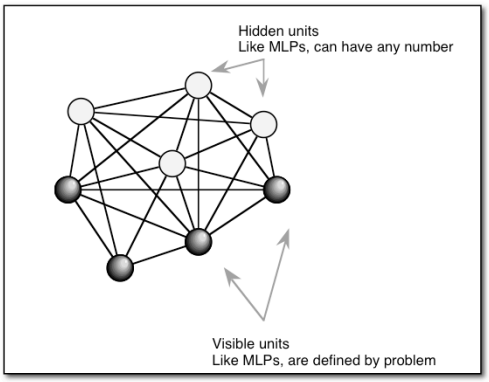
The introduction of the notion of temperature introduces a significant divergence from other learning mechanisms. The use of temperature introduces a level of randomness, such that the states that can be reached by the machine are a function of a probability distribution function. The level of probability being a function of temperature.

At high temperatures, the randomness is high and many different states are possible. This permits gross or high level details to be searched for. Local minima can be avoided because at high temperatures, the machine is able to jump out of minima.

At lower temperatures, there is less randomness and the machine is able to settle on and focus in on the details of each learned memory pattern.

# Comparison with Hopfield

- Boltzmann is a highly recurrent version of the Hopfield model
  - Like BP, hidden units are allowed






Neurons in a Boltzmann machine can be inputs, outputs or hidden units! Generally, a Boltzmann machine is similar in operation to a Hopfield net. In a Hopfield net, the number of neurons is tied to the number of dimensions in the problem space. In a Boltzmann machine this is also true for so called visible units. However, like a multi-layer perceptron, any number of hidden units may be added to increase capacity.

## Comparison with Hopfield cont

| Similarities | Differences |
|---|---|
| Binary outputs | Hidden units allowed |
| Symmetric weights | Stochastic neurons |
| Units selected for update at random | Training can be supervised or unsupervised |
| No self-connections | |



Major similarities

- Outputs must be binary (or bipolar) in nature. I.e. continuous values are not supported.
- Weights are symmetric. I.e. weight from neuron $u_1$ to $u_2$ is the same as from $u_2$ to $u_1$
- Asynchronous update of neuronal activations
- No self connections

Major differences

- A Boltzmann machine can have hidden units. Hopfield cannot.
- Perhaps the biggest difference between the Boltzmann machine and any other neural network paradigm is the use of stochastic neurons. I.e. the threshold function is stochastic not deterministic.
- Hopfield nets are typically used for pattern completion tasks and as such will undergo unsupervised learning (at least for autoassociators). Boltzmann machines can be configured to solve and generalize for supervised learning problems too.

# Usage

- Unsupervised learning
  - Units are outputs
  - Some units could be hidden
- Supervised learning
  - Mixture of output, input and hidden units
- Units can be ***clamped*** or ***free-running***

# Usage cont.

- Clamped units
  - have a fixed, unchanging I.e. clamped output
    - Input units are always clamped
    - In supervised learning, output units are clamped during training.
- Free-running units
  - output changes according to latest computed value
    - Hidden units are always free-running.

## Theory

- Ackley et al. (1986)
- Weights are symmetric

$$w_{i,j} = w_{j,i} \quad \text{for} \quad i \neq j$$



As with the Hopfield net, weights are symmetric,

$$w_{i,j} = w_{j,i} \quad \text{for} \quad i \neq j$$

## Theory cont.

- Neurons can be both input and output cells
  - A weighted sum is computed :

$$S_i = \sum_j w_{ij} u_j$$



Cells can be both input and output cells and each cell computes a weighted sum as normal:

$$S_i = \sum_j w_{ij} u_j$$

## Theory cont.

- Cell activation is ***stochastic***, not ***deterministic***
  - "Synthetic temperature"
  - Cell state :

$$u_i = 1 \text{ with probability } P_i = \frac{1}{1 + e^{-Si/T}}$$
$$u_i = 0 \text{ with probability } 1 - P_i$$

  - where $T$ is the synthetic temperature

November 2011 EECE 592 -The Boltzmann Machine

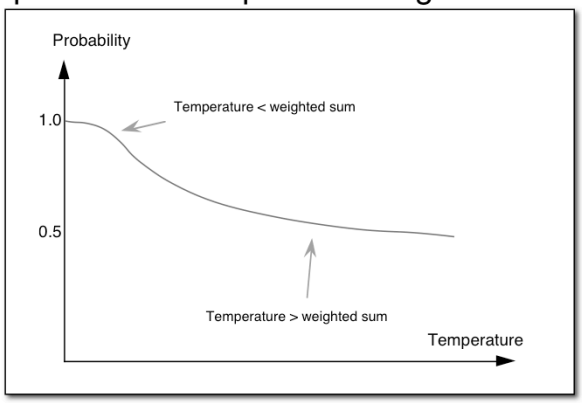
The activation of the cell in the Boltzmann machine is *stochastic*, not *deterministic* and the probability that a cell is in a given state depends on the *synthetic temperature* of the system:

$$u_i = \begin{cases} 1 \text{ with probability } P_i = \dfrac{1}{1 + e^{-S_i/T}} \\ 0 \text{ with probability } 1 - P_i \end{cases}$$

where $T$ is the synthetic temperature.

# Prob. vs. temp.

- graph shown for a positive weighted sum




- at high temperatures, probability is 0.5
  - I.e. all states are possible



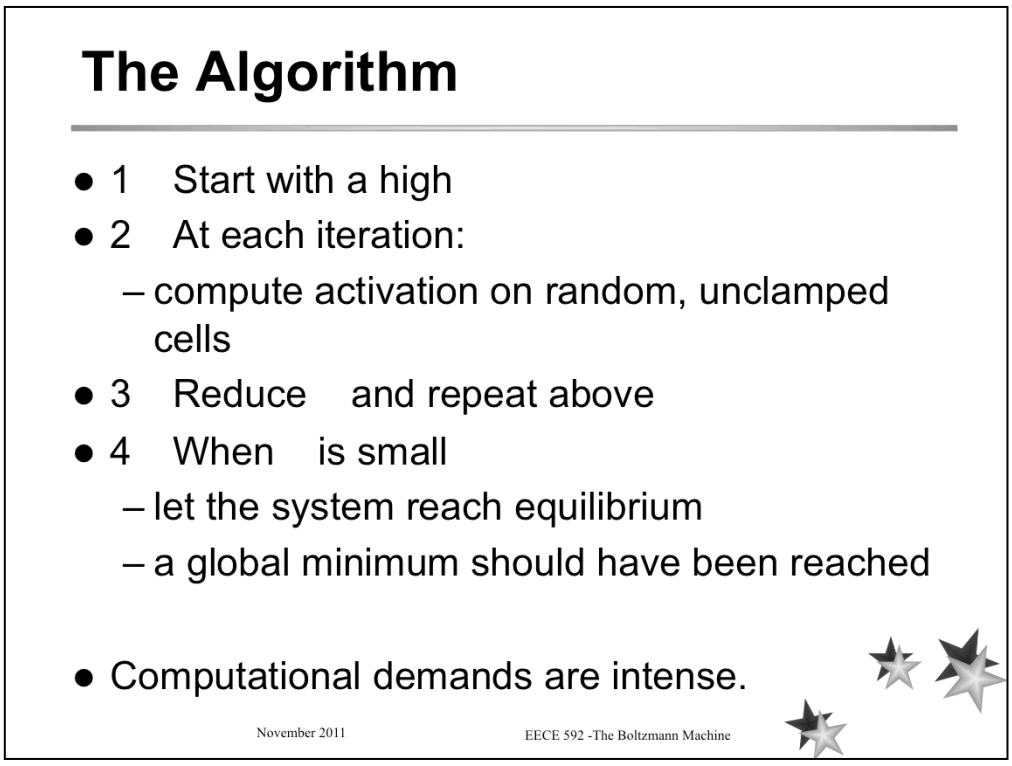
The graph shows, that for a positive weighted sum, that the activation function asymptotically approaches a value of 0.5 as temperature increases. Thus the probability of the neuron activating an output value of 1 is about 50%. This will be true for all neurons in the machine meaning that at high temperatures, any state is possible regardless of the problem being solved.

## Cells

- Clamped
  - Activations fixed
- Free-running
  - Activations updated



Input and output cells can be *clamped* or *free-running*. If they are clamped, then their activations, $u_i$ are fixed. If they are free-running, their activations change according to the above equation. Hidden cells are always free-running and input cells are typically clamped. Output neurons (when used for supervised learning) will be clamped when learning and free-running when in equilibrium. (Equilibrium is used to describe the state when the machine is free-running and allowed to reach a stable state of its own accord - I.e. based upon the current set of weights).




The general idea idea is as follows:

**1**

Start with a high temperature $T$.

**2**

At each iteration, select a cell, $u_i$, at random and if not clamped, compute a new activation. Statistics should be collected in order to build up a measure of the probability distribution.

**3**

Gradually reduce $T$ as iterations proceed. This is the essence of the annealing process and repeat the above steps.

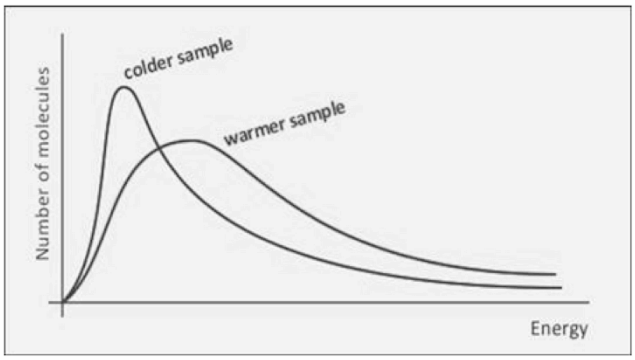
**4**

When $T$ is sufficiently small, let the system reach equilibrium, i.e. when no further activations change. At this point, theoretically, the minimum reached should have a high probability of being a global minimum.

However, computational demands are intensive and in practice this point may be difficult to reach without compromise.

# The Boltzmann Distribution

- A way of describing a gas in terms of the energy distribution of its molecules



## The Boltzmann Distribution

- Consider :
  - $\alpha$ = [1 0 0 1 1]
  - $\beta$ = [0 1 1 0 1]
- $\alpha$ and $\beta$ are two system states:



Consider two different states of the system, $\alpha$ and $\beta$. As an example, the outputs may be:

$$\alpha = [1\ 0\ 0\ 1\ 1\ ]$$
$$\beta = [\ 0\ 1\ 1\ 0\ 1]$$

Let $P(\alpha)$ be the probability at equilibrium that the system is in state $\alpha$ with energy $E_\alpha$ and likewise for $\beta$. Then it can be shown that the probability satisfies the Boltzmann distribution:

$$\frac{P(\alpha)}{P(\beta)} = e^{-(E_\alpha - E_\beta)/T}$$

## The Boltzmann Distribution cont.

- Probability of system at equilibrium:

of state $\alpha$ with energy $E_\alpha$ is $P(\alpha)$

of state $\beta$ with energy $E_\beta$ is $P(\beta)$

- Satisfies the Boltzmann distribution:

$$\frac{P(\alpha)}{P(\beta)} = e^{-(E_\alpha - E_\beta)/T}$$



The energies are calculated exactly as for the Hopfield net.

The Boltzmann distribution defines the relationship between the probabilities of being in one state versus another as a function of the difference between their energies.

The Boltzmann machine learns distributions not target values. So the goal here is for the probabilities to match the distribution of input/output cell patterns in the training example set. That is to get the the neural net to exhibit the same probability for a given state to match the probability for that same state as observed naturally by the problem itself. In BP, this is done by reducing total actual error and is based upon measuring the difference between actual and target outputs. In the Boltzmann machine, the comparison is between distributions.

## The Boltzmann Distribution cont.

- Energies calculated as for the Hopfield net.
  - Distribution of patterns ≈ Boltzmann distribution {at equilibrium}
- Kullback's asymmetric divergence G.
  - $P(a)$ - probability due to input distribution
  - $P'(a)$ - probability at equilibrium.

$$G = \sum_{\alpha} P(\alpha) \ln \frac{P(\alpha)}{P'(\alpha)}$$



To measure how closely two distributions match we can use the following approach (known as Kullback's asymmetric divergence):

Let $P(\alpha)$ be the probability of state $\alpha$ given by the training examples and $P'(\alpha)$ be the probability of state $\alpha$ when the net is free running at equilibrium. The asymmetric divergence G is :

$$G = \sum_{\alpha} P(\alpha) \ln \frac{P(\alpha)}{P'(\alpha)}$$

## Theory cont.

- Ackley et al. (1986)
  - learning derived from gradient descent.
- In BP
  - gradient descent performed on , total error.
- For Boltzmann machine
  - gradient descent performed using , asymmetric divergence



Ackley et al. suggested using *G* as an error function and effectively performing gradient descent on it as a way of computing weight changes:

## Theory cont.

- In BP

$$\Delta_p w_{i,j} \propto -\frac{\partial E^p}{\partial w_{ij}}$$

- For Boltzmann machine

$$\Delta w_{i,j} = -\rho \frac{\partial G}{\partial w_{ij}}$$

November 2011 EECE 592 -The Boltzmann Machine

Ackley et al. suggested using $G$ as an error function and effectively performing gradient descent on it as a way of computing weight changes:

Gradient descent as before is: $\Delta w_{i,j} = -\rho \frac{\partial G}{\partial w_{ij}}$

where $\rho$ is the learning rate.

Ackley et al. showed that: $\frac{\partial G}{\partial w_{ij}} = -\frac{1}{T}(p_{i,j} - p'_{i,j})$

Where $p_{i,j}$ is the probability that $u_i$=$u_j$=1 for training examples (i.e. clamped input/outputs)

And $p'_{i,j}$

is the probability that $u_i$=$u_j$=1 for the net free-running at equilibrium.

Then

$$\Delta w_{i,j} = \rho(p_{i,j} - p'_{i,j})$$

The problem is estimating $p'_{i,j}$ and $p_{i,j}$ .

# Theory cont.

- Ackley et al. showed that:

$$\frac{\partial G}{\partial w_{ij}} = -\frac{1}{T}(p_{i,j} - p'_{i,j})$$

where
$p_{i,j}$ is probability $u_i = u_j = 1$ from input distribution

and
$p'_{i,j}$ is probability $u_i = u_j = 1$ at equilibrium

- Then weight update is:

$$\Delta w_{i,j} = \rho(p'_{i,j} - p_{i,j})$$

## Theory cont.

- The problem is estimating $_{i,j}$ and $_{i,j}$
- Estimation of $_{i,j}$
  - Gather statistics from net with clamped inputs/outputs
- Estimation of $_{i,j}$
  - Gather statistics from net with unclamped outputs



Estimation of $p_{i,j}$

Each training vector is clamped onto the input and output units and the net allowed to reach equilibrium.

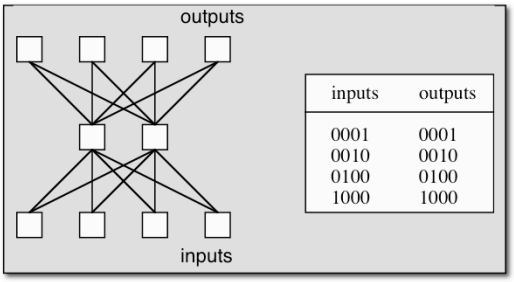
To estimate $p_{i,j}$ requires gathering statistics over time about how often pairs of units were both *on* at the same time.

Estimation of $p'_{i,j}$

The network should be completely unclamped and allowed to reach equilibrium. Again statistics are required to estimate $p'_{i,j}$ .

# Example

- The 4-2-4 Encoder
  - Famous demonstration of Boltzmann learning




| inputs | outputs |
|---|---|
| 0001 | 0001 |
| 0010 | 0010 |
| 0100 | 0100 |
| 1000 | 1000 |

  - {Interconnections between input units and interconnections between output units not shown}



The 4-2-4 Encoder

A famous demonstration of Boltzmann learning is the application of the technique to build a 4-2-4 encoder.

The diagram shows the connectivity of cells used in the Boltzmann machine. (Not shown on the diagram are interconnections between input units and interconnections between output units)

The action that the encoder is performing is one of compression. With the 4-2-4 encoder, the network is forced to seek a representation which can encode the position of the only *on* input using just two units.

Ackley et al. successfully trained the Boltzmann machine to perform encoding and found that as expected the net was developing a base 2 representation to encode/decode the input patterns.

# RBM

- Restricted Boltzmann Machine
  - An active research topic (2006)
  - A more efficient & practical form of the Boltzmann Machine
  - Used to model faces and digits
    - See Hinton, G. E., Osindero, S. and Teh, Y. (2006) A fast learning algorithm for deep belief nets. Neural Computation, 18: 1527-1554

## RBMs cont.

- Restrictions are:
  - Hidden and visible not connected
  - Learning is via Contrastive Divergence
  - Many layers of hidden neurons possible.
  - So called *deep belief network*.



Regarding restrictions. More specifically, no intra-layer conections are permitted within a hidden layer or a input or output layer.

# RBMs cont

http://www.cs.toronto.edu/~hinton/adi/index.htm

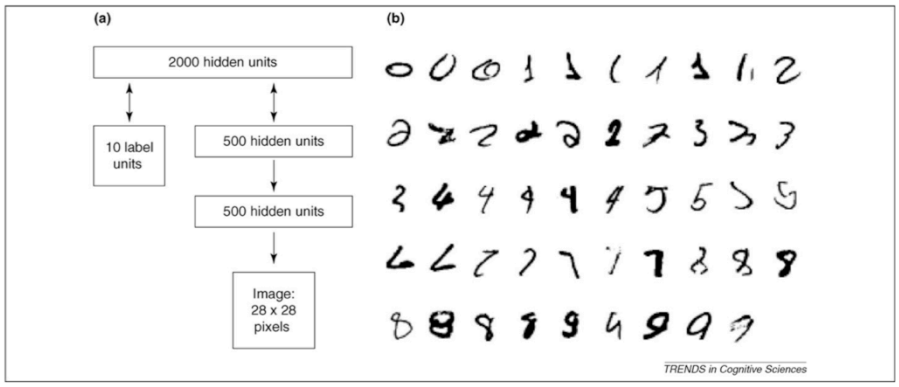



**Figure 1.** **(a)** The generative model used to learn the joint distribution of digit images and digit labels. **(b)** Some test images that the network classifies correctly even though it has never seen them before.