Boltzmann Machines

Neural Networks

Bibliography

Ackley, D. H., Hinton, G. E, and T. J. Sejnowski, "A Learning Algorithm for Boltzmann Machines," *Cognitive Science* 9:147-169

Kirkpatrick, S., Optimization by Simulated Annealing: Quantitative Studies, *Journal of Statistical Studies*, Vol. 34, Nos. 5/6. 1984.

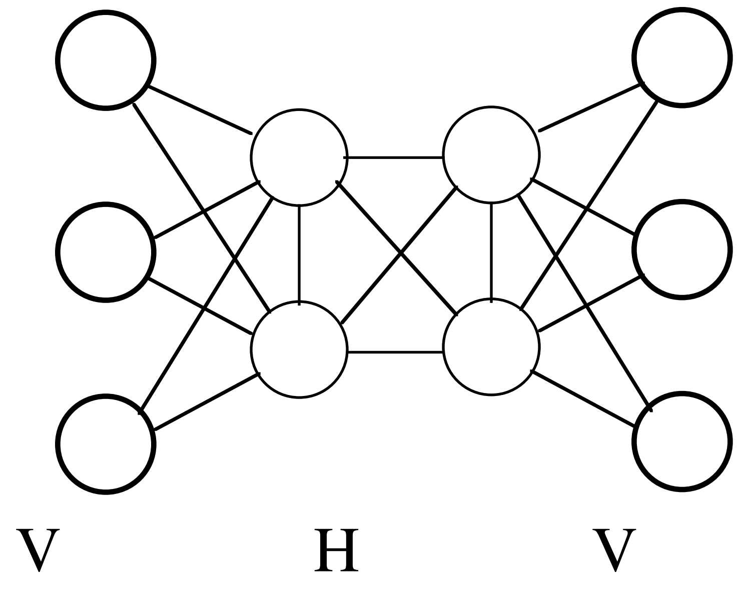
Boltzmann Machine

Bidirectional Net with Visible and Hidden Units

Learning Algorithm

Can Seek Global Minima

Avoids local minima (& speeds up a slow learning algorithm) through stochastic nodes and simulated annealing



Unit: Logistic Function

For a node

$$\Delta E_k = net = \sum_i (w_{ki} \cdot s_i) - \theta_k$$

Output: $s_k = 1$ with probability

$$P_k = \frac{1}{1 + e^{-\Delta E_K / T}}$$

where T = Temperature

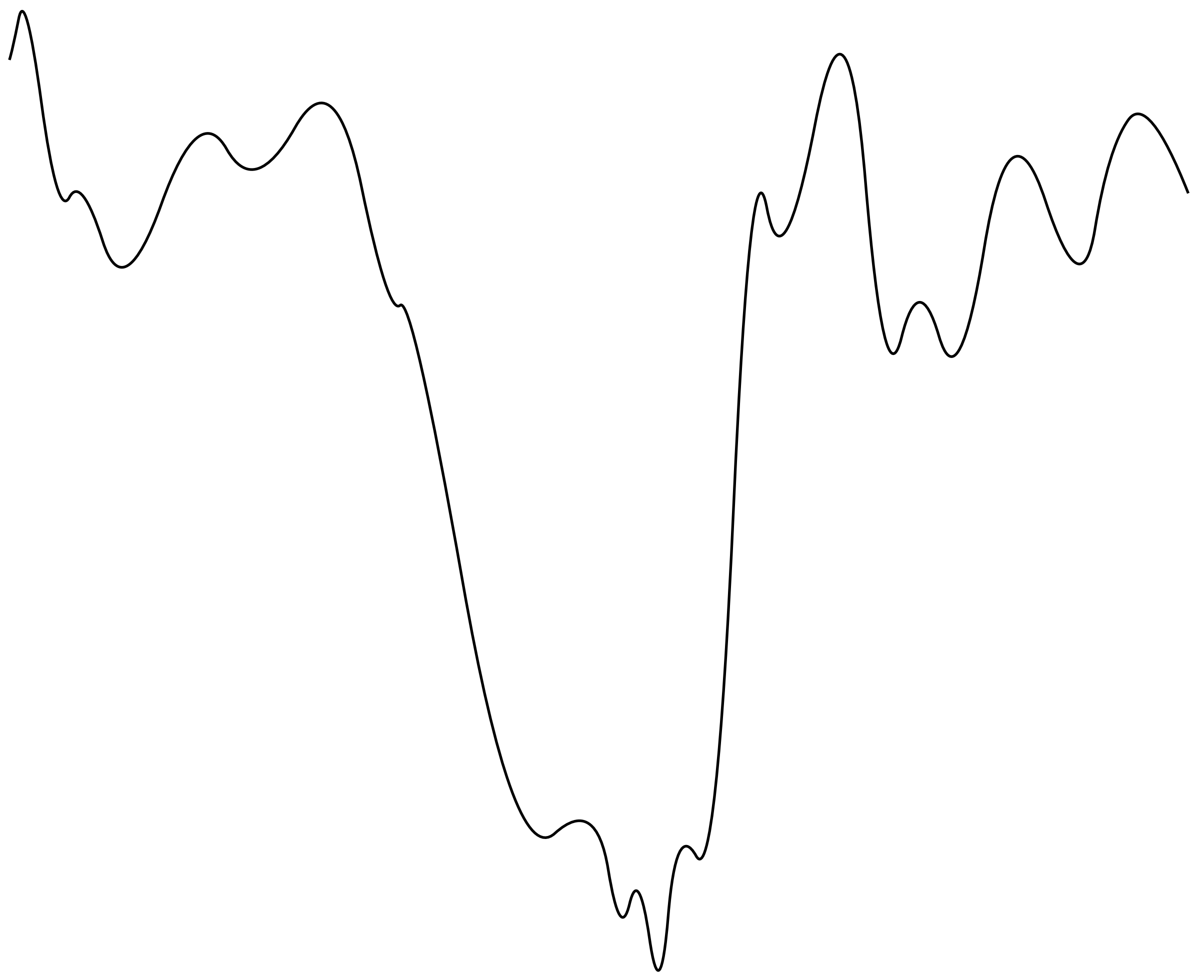
Asynchronous Random Updates

Global Energy Function Like Hopfield

$$E = \sum_{i \pi j} (wij \cdot sisj) + \sum_{i} \theta_i \cdot si$$

w: weights
s: outputs
θ**: Bias**

Simulated Annealing



1. Start with high T
More randomness in update and large jumps

2. Progressively lower T until equilibrium reached

(Minima Escape and Speed)

Learning Algorithm

System at thermal equilibrium obeys the Boltzmann Distribution

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

$P^+(V_\alpha)$ = Probability of state α when clamped

Depends only on the training set environment

$P^-(V_\alpha)$ = Probability of state α when free

Goal: $P^-(V_\alpha) \approx P^+(V_\alpha)$

For example, a training set

1 0 0 1
1 1 1 0
1 0 0 1
0 0 0 0

What are Probabilities

Could be auto or pattern associator

Learning Mechanisms

Information Gain (G) is a measure of similarity between $P^-(V_\alpha)$ and $P^+(V_\alpha)$

$$G = \sum_\alpha P^+(V_\alpha) \ln \frac{P^+(V_\alpha)}{P^-(V_\alpha)}$$

G = 0 if the same, positive otherwise

So, when can seek a gradient descent algorithm for weight change by taking the partial derivative

$$\frac{\partial G}{\partial w_{ij}} = - \frac{1}{T} (p^+_{ij} - p^-_{ij})$$

$$\Delta w_{ij} = C\,(p^+_{ij} - p^-_{ij})$$

p_{ij} = probability that p_i and p_j are simultaneously on when in equilibrium

Logistic Node & Annealing break out of local minima

Annealing and Statistics Gathering

A network time step is the period in which each node has updated ≈ once.

Initialize node outputs to random values (except for visible when in the clamped state)

Annealing Schedule

i.e.

2@30, 3@20, 3@10, 4@5

Then gather p_{ij} stats for 10 time steps

Learning Algorithm (Intuitive)

Separate Visible units into Input & Output units

```
Until Convergence ($\Delta w < \varepsilon$)
    Pick a pattern and clamp all visible units
    anneal and gather $p^{+}_{ij}$
    Unclamp Output units
    Anneal and gather $p^{-}_{ij}$
    Update weights
End
```

Might work, but not the true algorithm

Boltzmann Learning Algorithm

```
Until Convergence ($\Delta w < \varepsilon$)
    For each pattern in training set
        Clamp pattern on all visible units
        Anneal several times and gather $p^{+}_{ij}$
    end
    Average $p^{+}_{ij}$ for all patterns
    Unclamp all visible units
    Anneal several times and gather $p^{-}_{ij}$
    Update weights
End
```

Tricks

Noisy Input Vectors

To avoid infinite weights for non-trained states

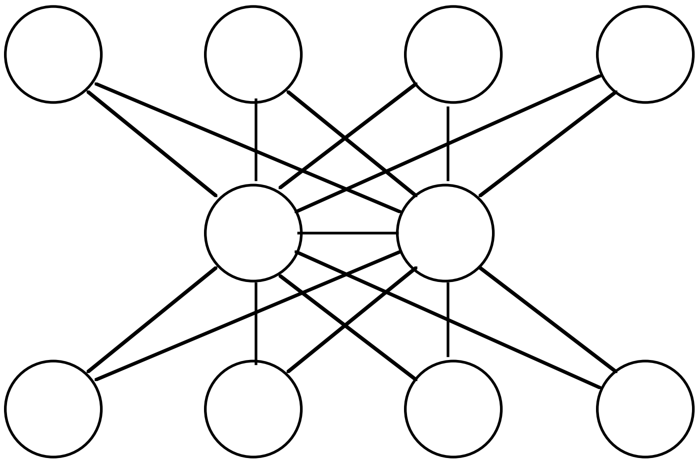
For each bit in a pattern during training, have a finite probability of toggling it.

Weight Decay

Fixed Magnitude Weight Changes

Encoder Problem

Map Single Input Node to Single Output Node



requires ≥ log(n) hidden units

For 4-2-4 Encoder

1. Anneal and gather p^{+}_{ij} for each pattern twice (10 time steps for gather). Noise .15 of 1 to 0, .05 of 0 to 1.

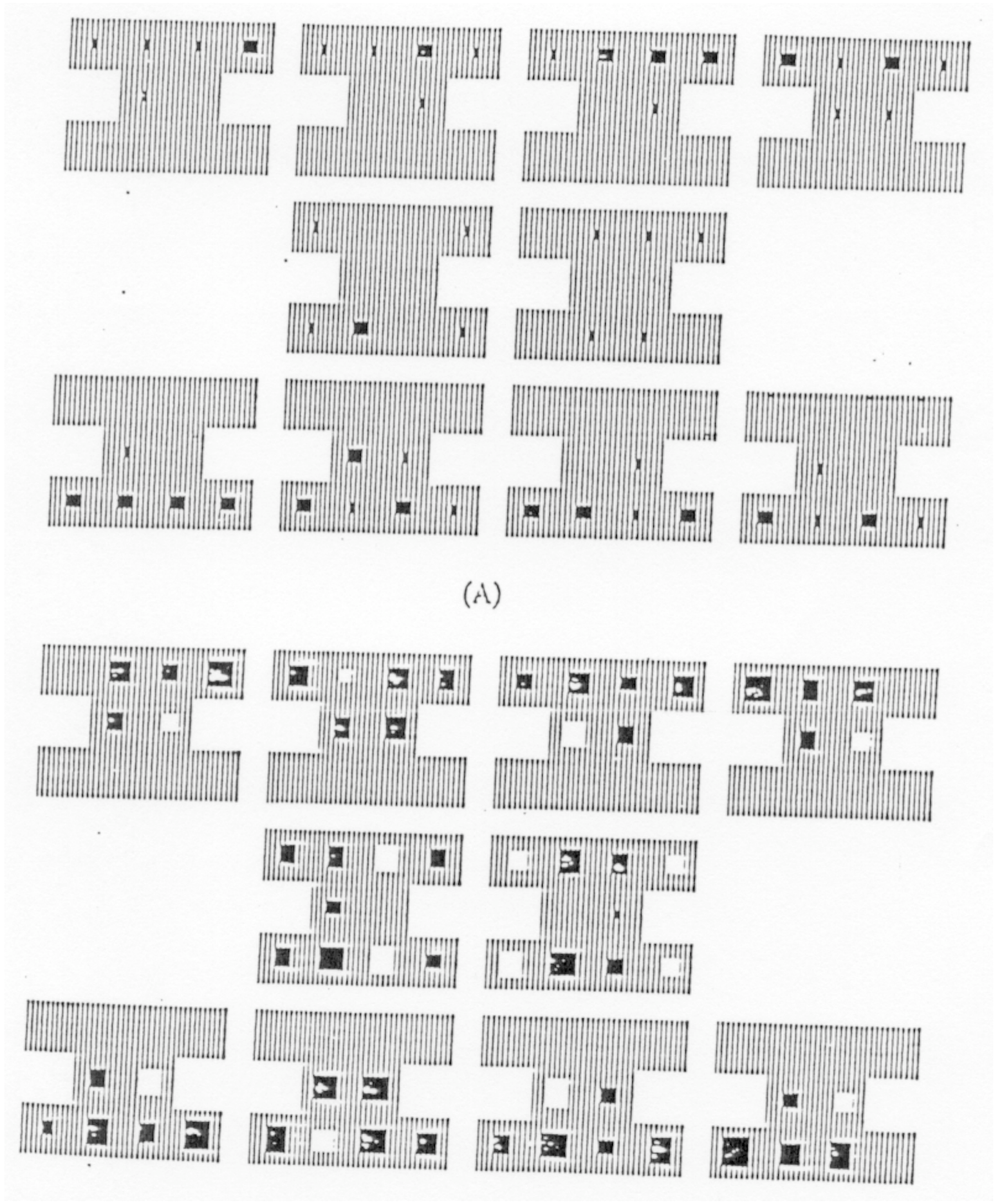
Annealing Schedule: 2@20,2@15,2@12,4@10

2. Anneal and gather p^{-}_{ij} in free state an equal number of times

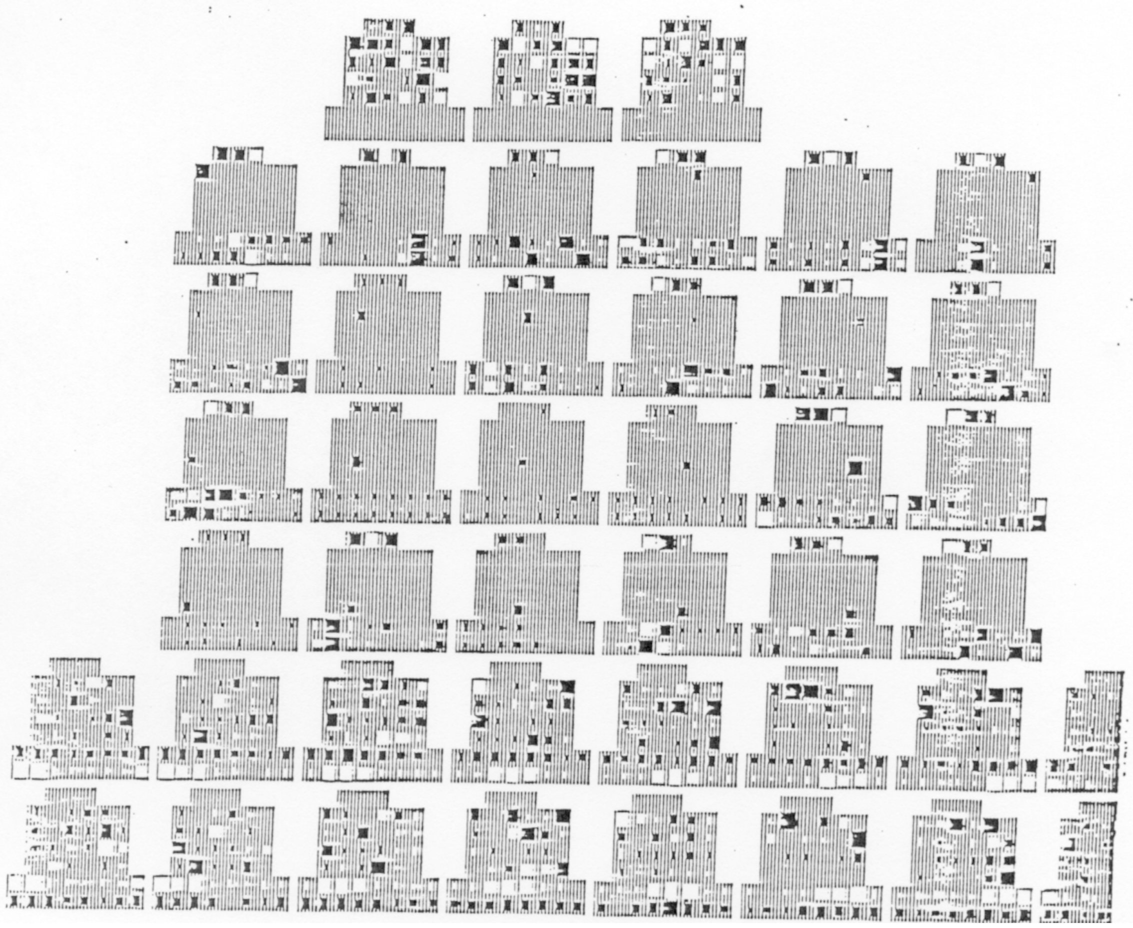
3. $\Delta w_{ij} = 2\,(p^{+}_{ij} - p^{-}_{ij})$

Average: 110 cycles

Example Encoder Weights (Before,After)



Shifting Network
9000 Cycles
No I/O Directionality



Boltzmann Summary

Stochastic Relaxation

More General than Hopfield - Can do arbitrary functions

Slow learning algorithm

Completely Probabalistic Model - Seeks to mimic the environment

Annealing and stochastic units help speed and minima escaping