Unsupervised Deep nets

Bryan Pardo

Northwestern University

(updated fall 2020)

Hebbian Learning

Donald Hebb

- A cognitive psychologist active mid 20th century
- Influential book: The Organization of Behavior (1949)
- Hebb’s postulate

"Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability.… When an axon of cell *A* is near enough to excite a cell *B* and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that *A*'s efficiency, as one of the cells firing *B*, is increased.

Pithy version of Hebb’s postulate

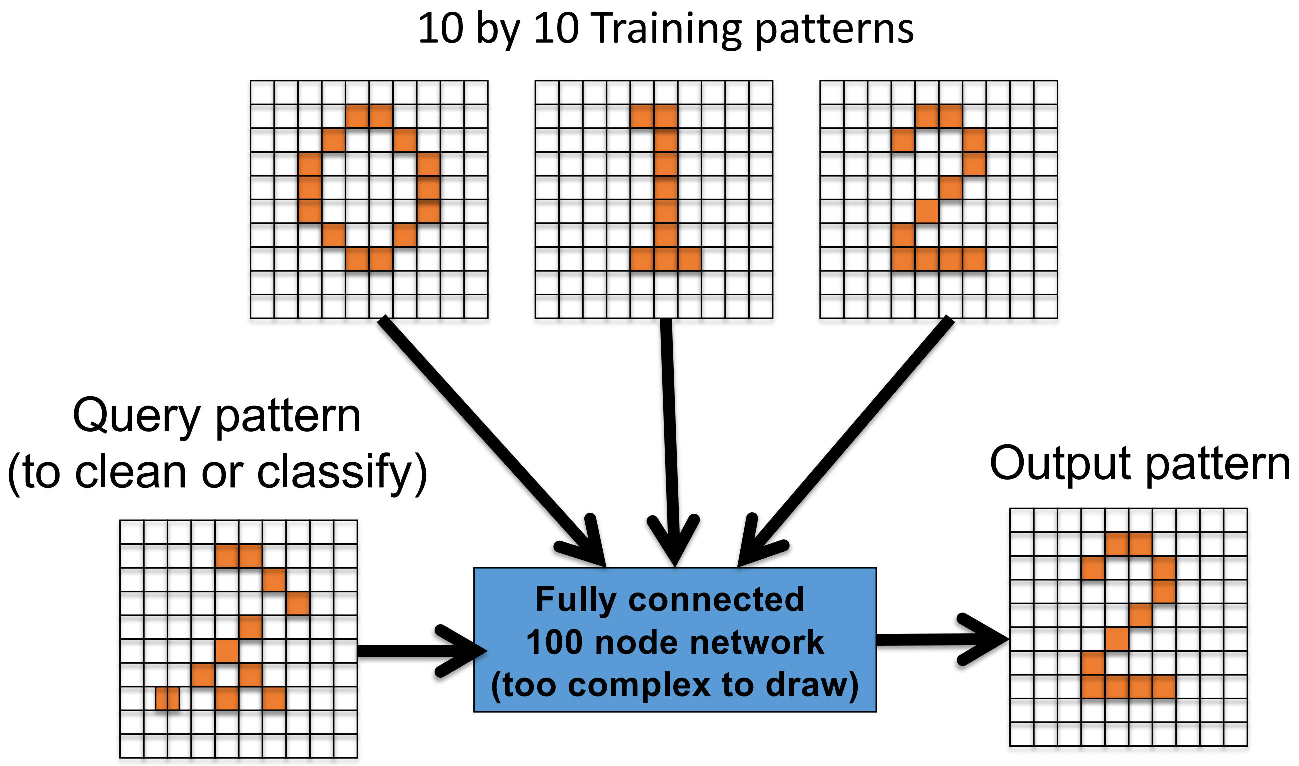
Cells that fire together, wire together.

Hopfield networks

Hopfield nets are

- “Hebbian” in their learning approach
- Old technology. These are proof of concept things from the 80s and earlier.
- Fast to train, slow to use
- Weights are symmetric
- All nodes are input & output nodes
- Use binary (1, -1) inputs and output
- Used as Associative memory (image cleanup)Classifier
- Precursors to Auto Encoders and Restricted Boltzman Machines

Using a Hopfield Net



Training a Hopfield Net

- Assign connection weights as follows

$$w_{ij} = \begin{cases} \displaystyle\sum_{c=1}^{C} x_{c,i} x_{c,j} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}$$

c	index number for C many class exemplars
i, j	index numbers for nodes
$w_{i,j}$	connection weight from node i to node j
$x_{c,i} \in \{+1, -1\}$	element i of the exemplar for class c

Using a Hopfield Net

Force output to match an unknown input pattern

$$s_i(0) = x_i \qquad \forall i$$

Here, $s_i(t)$ is the state of node i at time t
and x_i is the value of the input pattern at node i

Iterate the following function until convergence

$$s_i(t+1) = \begin{cases} 1 & \text{if } 0 \le \sum_{j=1}^{N} w_{ij} s_j(t) \\ -1 & else \end{cases}$$

Note: this means you have to pick an order for updating nodes.
People often update all the nodes in random order

Using a Hopfield Net

Once it has converged…

FOR INPUT CLEANUP: You’re done. Look at the final state of the network.

FOR CLASSIFICATION: Compare the final state of the network to each of your input examples. Classify it as the one it matches best.

Input Training Examples

Why isn't 5 in the set of examples?

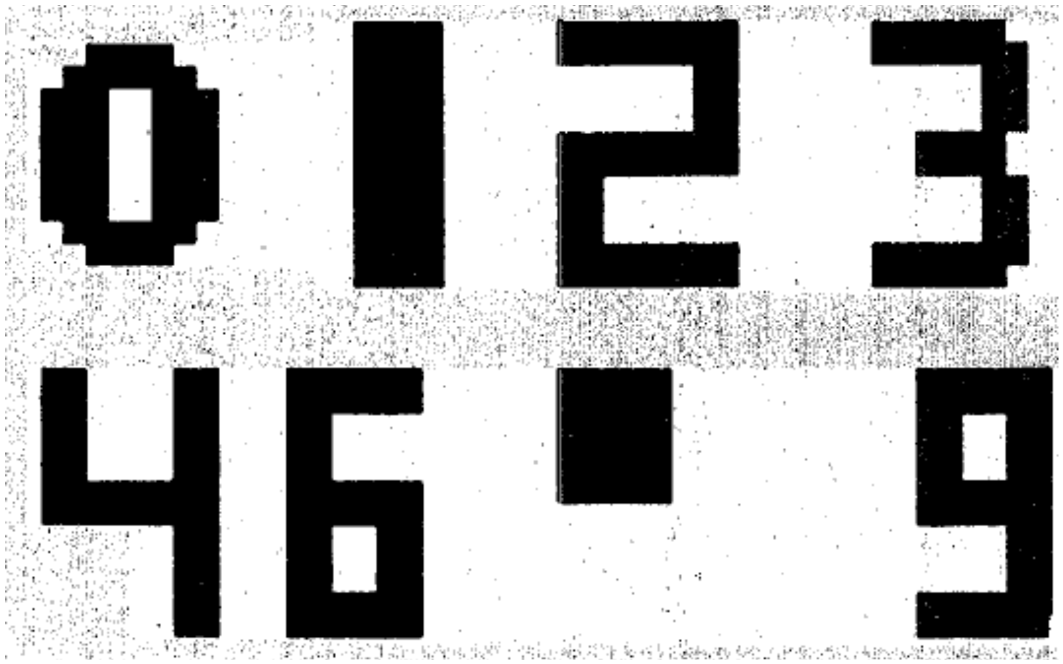


Image from:R. Lippman, An Introduction to Computing with Neural Nets, IEEE ASSP Magazine, April 1987

Output of network over 8 iterations

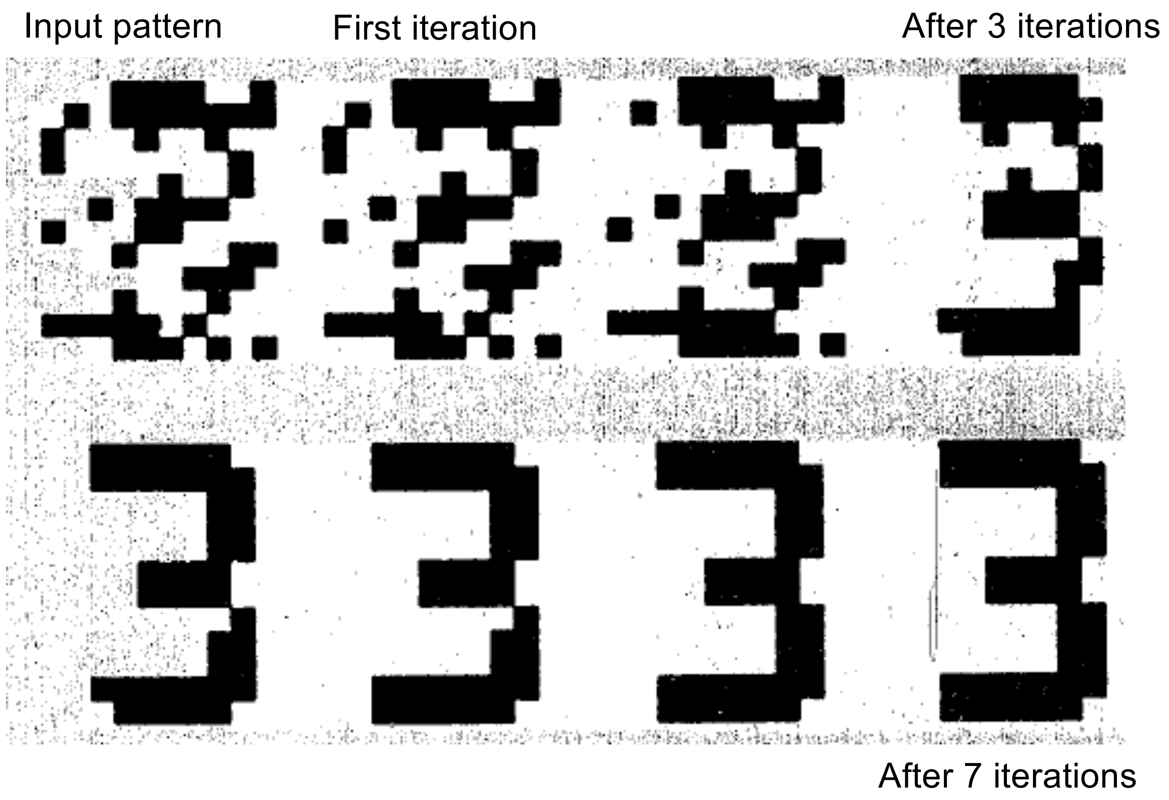
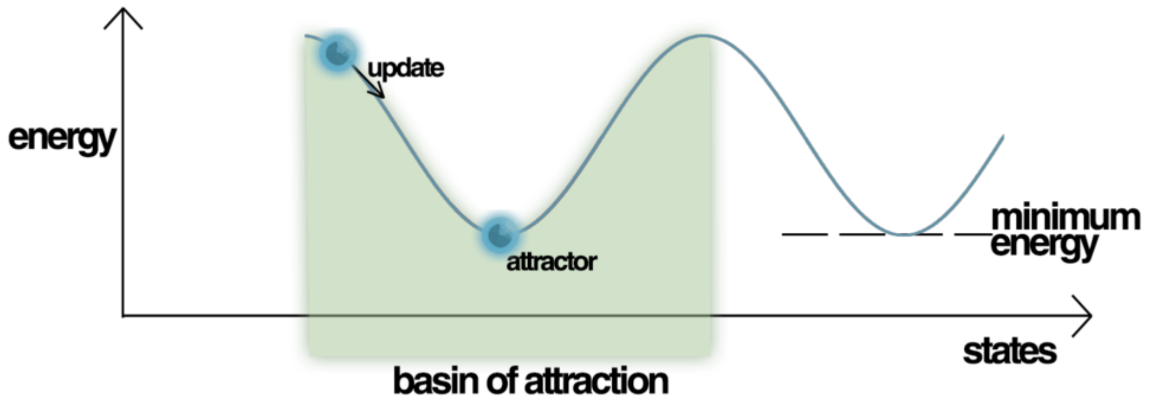


Image from:R. Lippman, An Introduction to Computing with Neural Nets, IEEE ASSP Magazine, April 1987

Characterizing “Energy”



- As $s_i(t)$ is updated, the state of the system converges on an “attractor”, where $s_i(t+1) = s_i(t)$
- Convergence is measured with this “Energy” function:

$$E(t) = -\frac{1}{2}\sum_{i,j} w_{ij} s_i(t) s_j(t)$$

Note: people often add a “bias” term to this function. I’m assuming we’ve added an extra “always on” node to make our “bias”

Image from: http://en.wikipedia.org/wiki/Hopfield_network#mediaviewer/File:Energy_landscape.png

Limits of Hopfield Networks

- Input patterns become confused if they overlap
- The number of patterns it can store is about 0.15 times the number of nodes
- Retrieval can be slow, if there are a lot of nodes (it can take thousands of updates to converge)

Restricted boltzman machine (RBM)

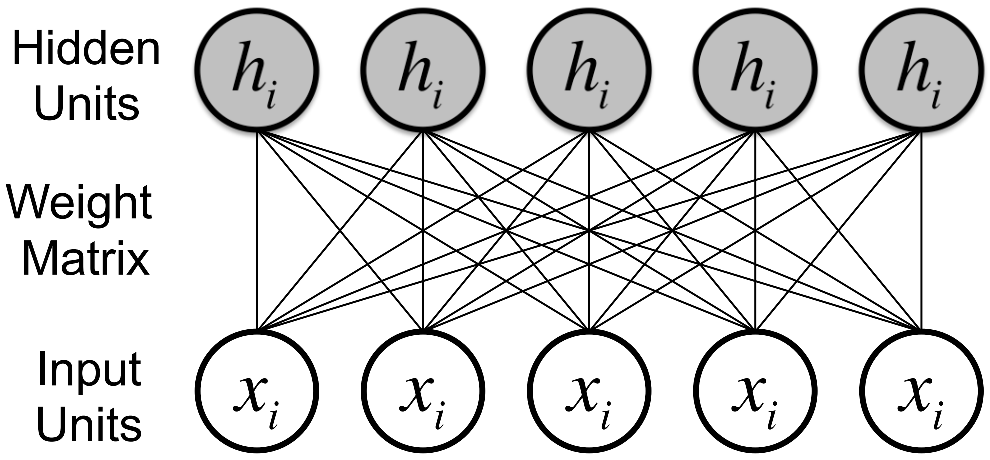
About RBNs

- Related to Hopfield nets
- Used extensively in Deep Belief Networks
- You can’t understand DBNs without understanding these

About RBNs

- Related to Hopfield nets
- Used extensively in Deep Belief Networks
- You can’t understand DBNs without understanding these

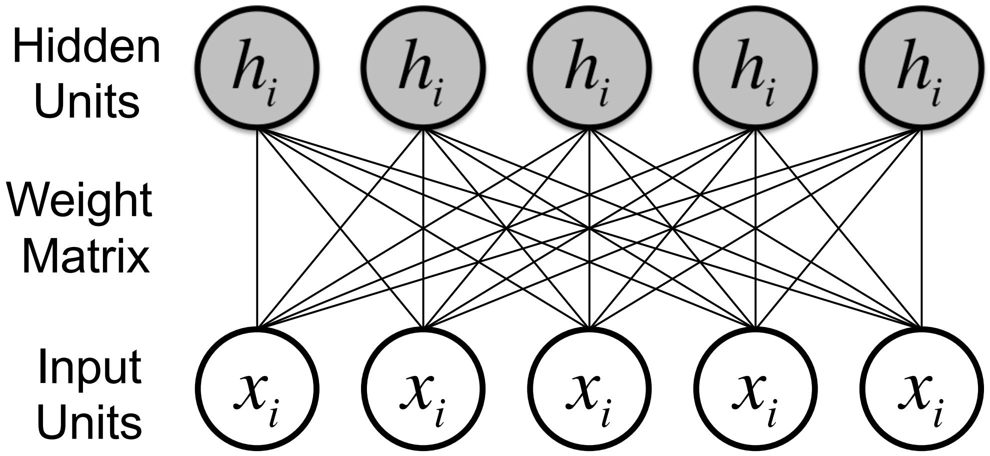
Standard RBM Architecture



2 layers (hidden & input) of Boolean nodes

Nodes only connected to the other layer

Standard RBM Architecture



Setting the hidden nodes to a vector of values updates the visible nodes…and vice versa

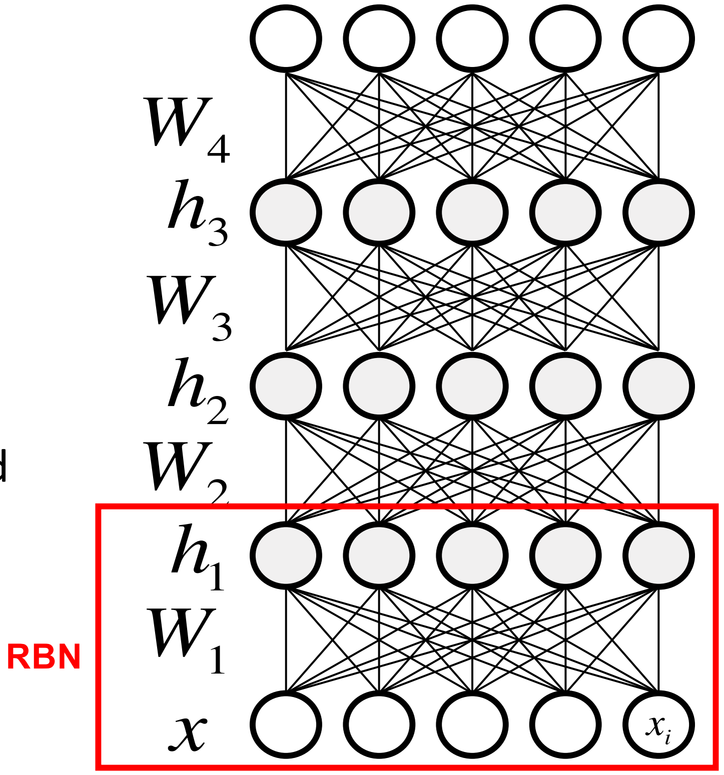
Contrastive Divergence Training

1. Pick a training example.
2. Set the input nodes to the values given by the example.
3. See what activations this gives the hidden nodes.
4. Set the hidden nodes at the values from step 3.
5. Set the input node values, given the hidden nodes
6. Compare the input node values from step 5 to the the input node values from step 2
7. Update the connection weights to decrease the difference found in step 6.
8. If that difference falls below some epsilon, quit. Else, go to step 1.

Deep BELIEF Network (DBN)

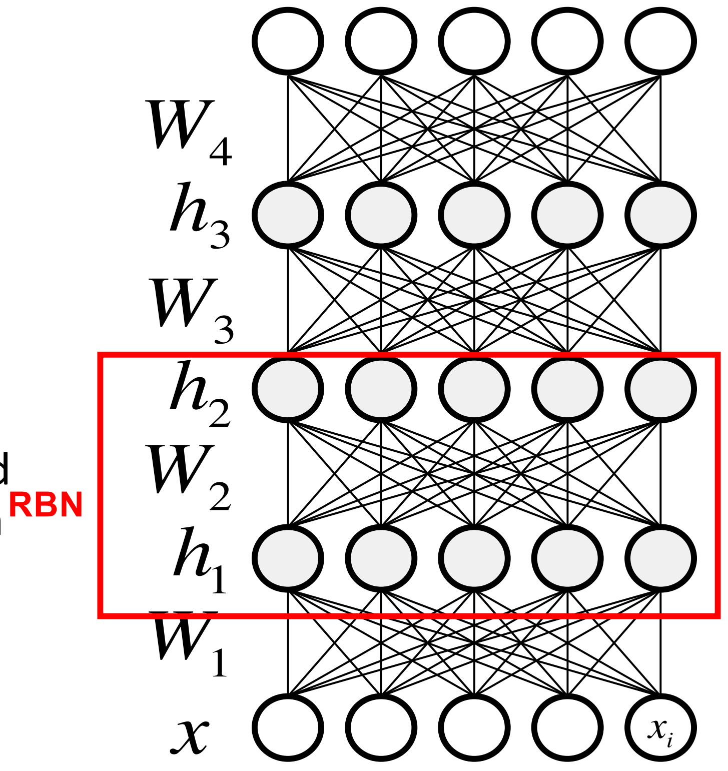
What is a Deep Belief Network?

- A stack of RBNS
- Trained bottom to top with Contrastive Divergence
- Trained AGAIN with supervised training (similar to backprop in MLPs)



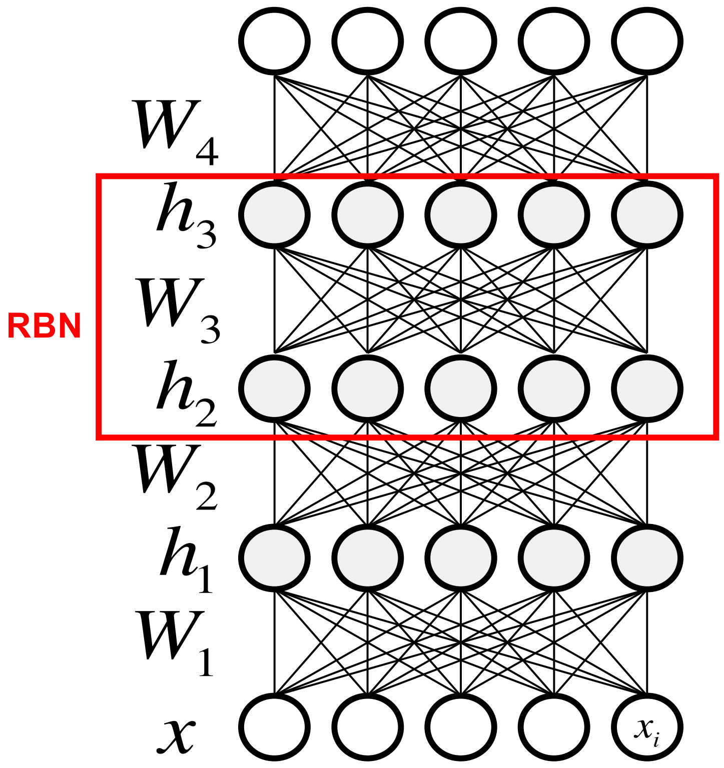
What is a Deep Belief Network?

- A stack of RBNS
- Trained bottom to top with Contrastive Divergence
- Trained AGAIN with supervised training (similar to backprop in MLPs)



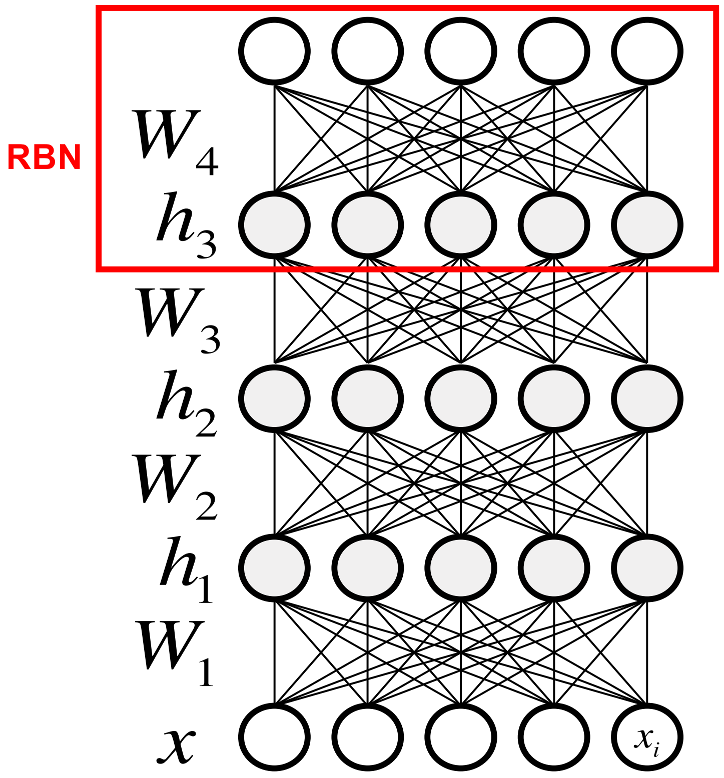
What is a Deep Belief Network?

- A stack of RBNS
- Trained bottom to top with Contrastive Divergence
- Trained AGAIN with supervised training (similar to backprop in MLPs)



What is a Deep Belief Network?

- A stack of RBNS
- Trained bottom to top with Contrastive Divergence
- Trained AGAIN with supervised training (similar to backprop in MLPs)



Why are DBNs important?

- Around 2005 they were state-of-the-art systems for doing certain recognition tasks
  - Handwritten digits
  - Phonemes
- They got the whole "Deep learning" thing going
- They have a good marketing campaign "Deep learning" vs "shallow learning"

How does “deep” help?

- It may be possible to much more naturally encode problems like the parity problem with deep representations than with shallow ones

Why not use standard MLP training?

- Fading signal from backprop
- The more complex the network, the more likely there are local minima
- Memorization issues
- Training set size and time to learn

Benefits

- Allowed relatively deep networks (e.g. 10 layers), compared to regular multilayer perceptrons with sigmoid activations
- In the mid 2000's this approach made networks better than anything else out there for some problems (e.g. digit recognition, phoneme recognition)
- Today, they've been superseded by other approaches

Autoencoders

Variational Autoencoders