Introduction to Neural Networks

Johns Hopkins University
Engineering for Professionals Program
605-447/625-438
Dr. Mark Fleischer

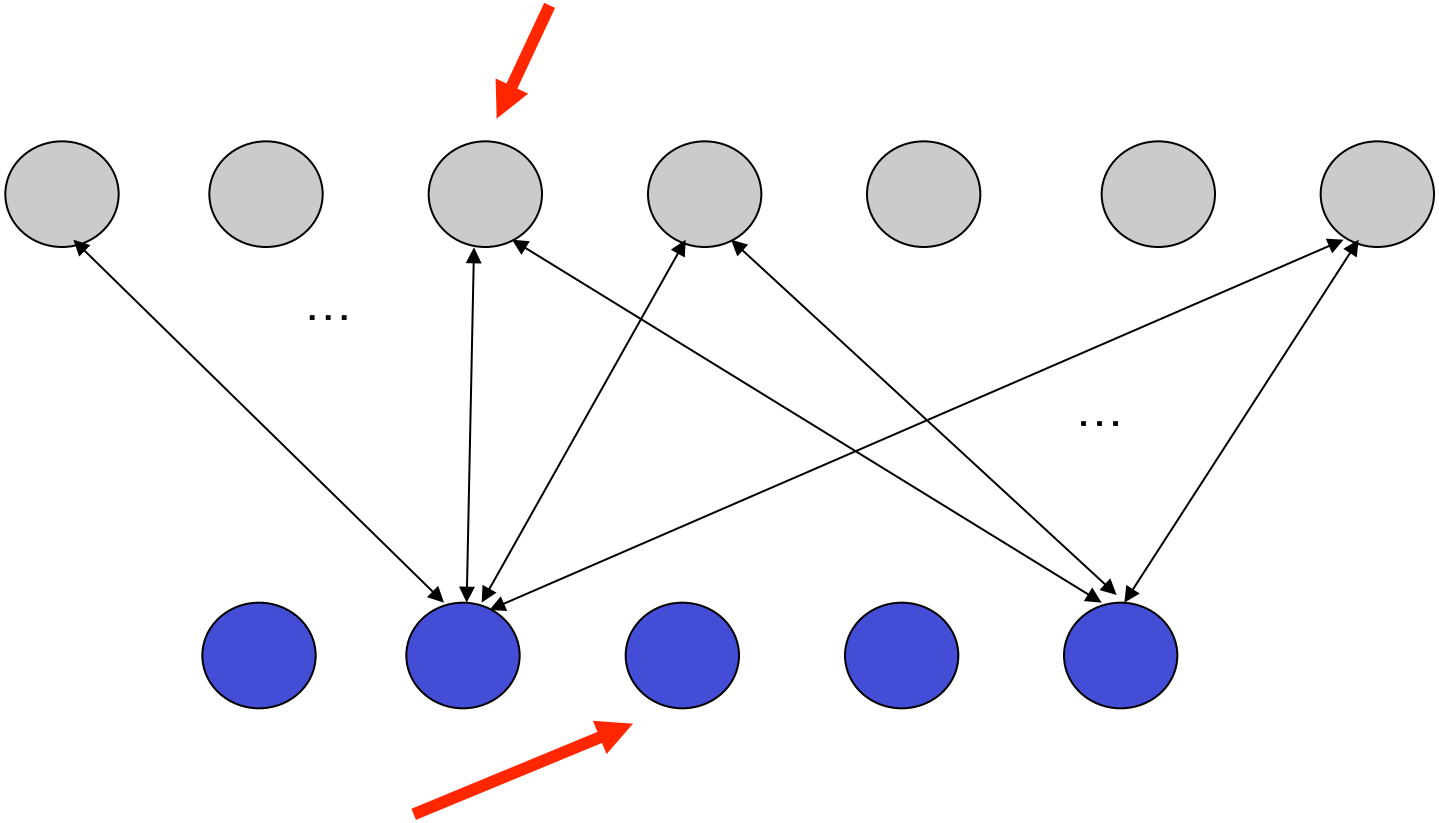
Module 11.3: An RBM Training Example

What We've Covered So Far

- Examined the mathematical foundations for an efficient approach to training RBMs.
- Derivative of the log probability of a vector **v.**
- Two expectations involved.
  - First term based on frequency of $v_i h_j$ over the training set of vectors **v.**
  - Second term based on frequency of $v_i h_j$ over all vectors **v** and **h.**

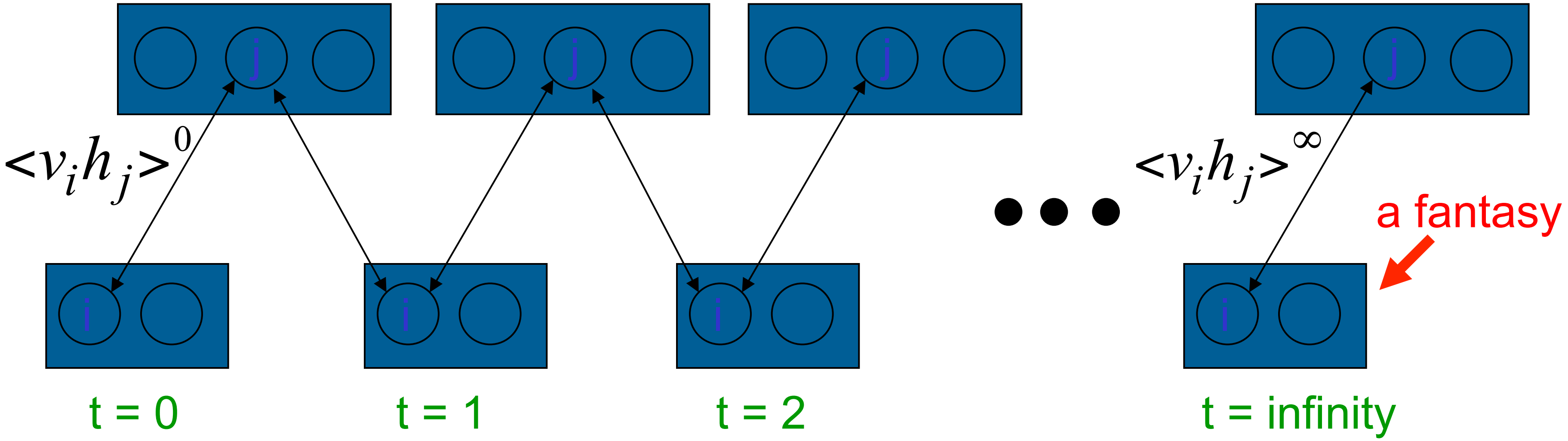
RBM

Hidden Layer of Nodes/Feature Detectors



Visible Layer of Nodes

A picture of the maximum likelihood learning algorithm for an RBM



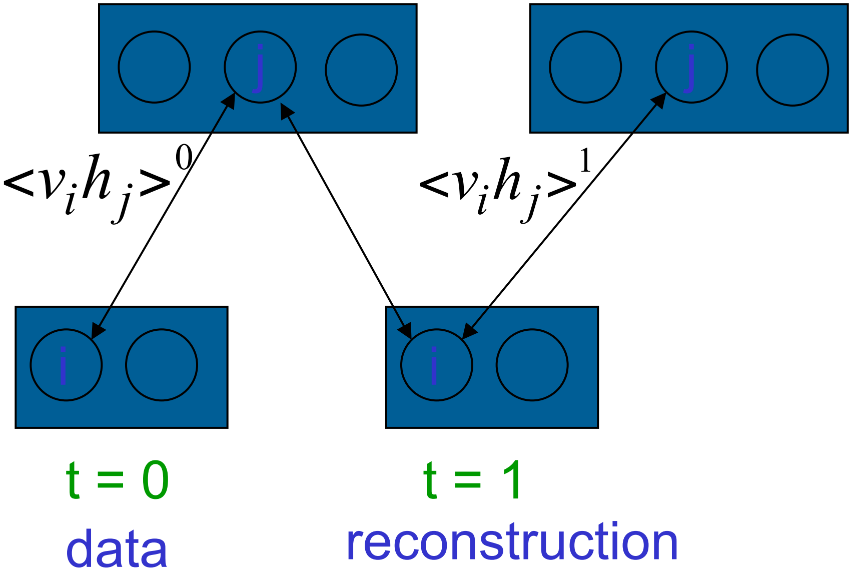
Start with a training vector on the visible units.

Then alternate between updating all the hidden units in parallel and updating all the visible units in parallel.

$$\frac{\partial \log p(v)}{\partial w_{ij}} = <v_i h_j>^0 - <v_i h_j>^\infty$$

From Hinton 2007

A quick way to learn an RBM



Start with a training vector on the visible units.

Update all the hidden units in parallel

Update the all the visible units in parallel to get a “reconstruction”.

Update the hidden units again.

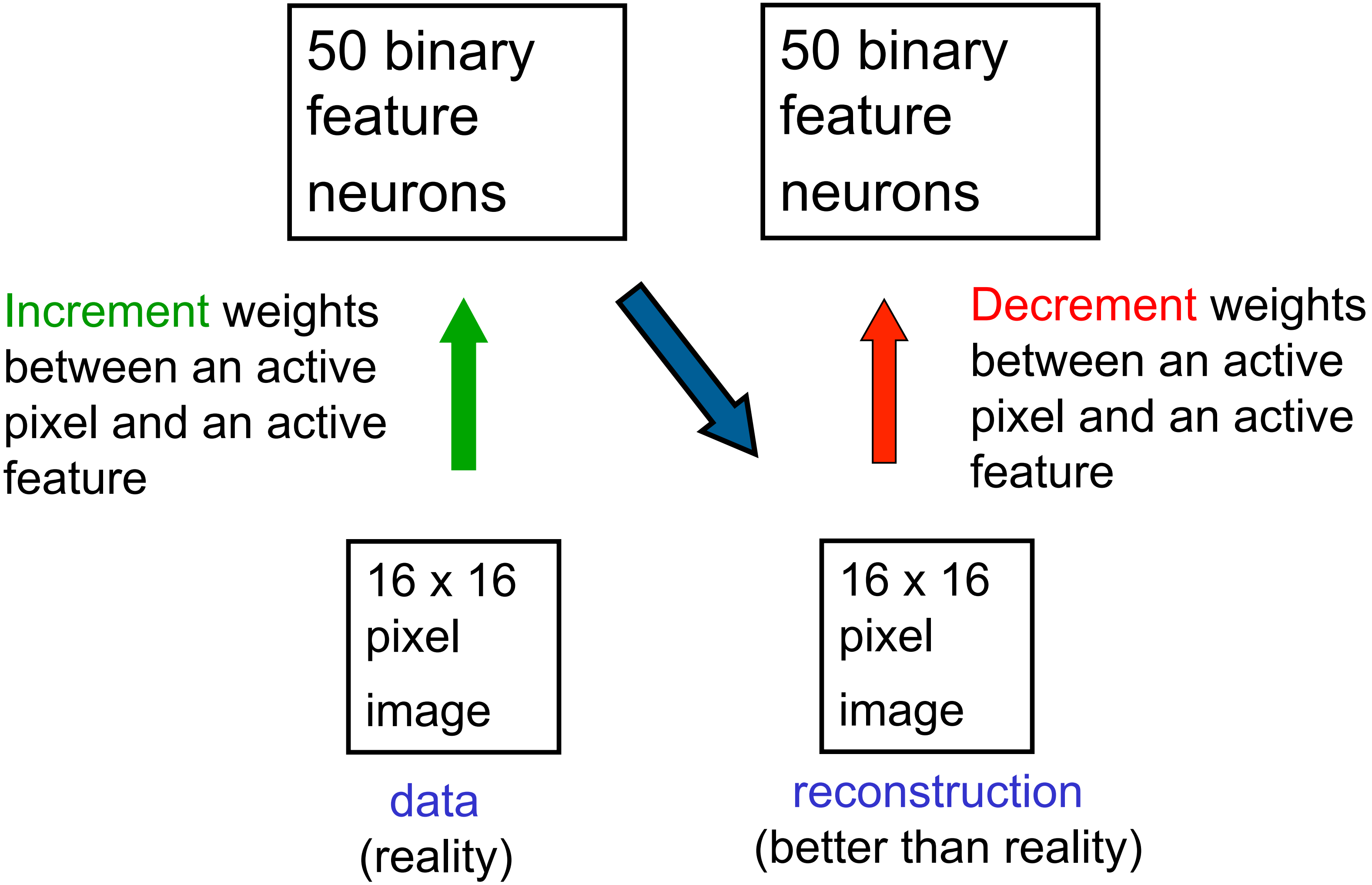
$$\Delta w_{ij} = \varepsilon \, (<v_i h_j>^0 - <v_i h_j>^1)$$

This is not following the gradient of the log likelihood. But it works well.

It is approximately following the gradient of another objective function.

From Hinton 2007

How to learn a set of features that are good for reconstructing images of the digit 2



50 binary feature neurons

50 binary feature neurons

Increment weights between an active pixel and an active feature

Decrement weights between an active pixel and an active feature

16 x 16 pixel image

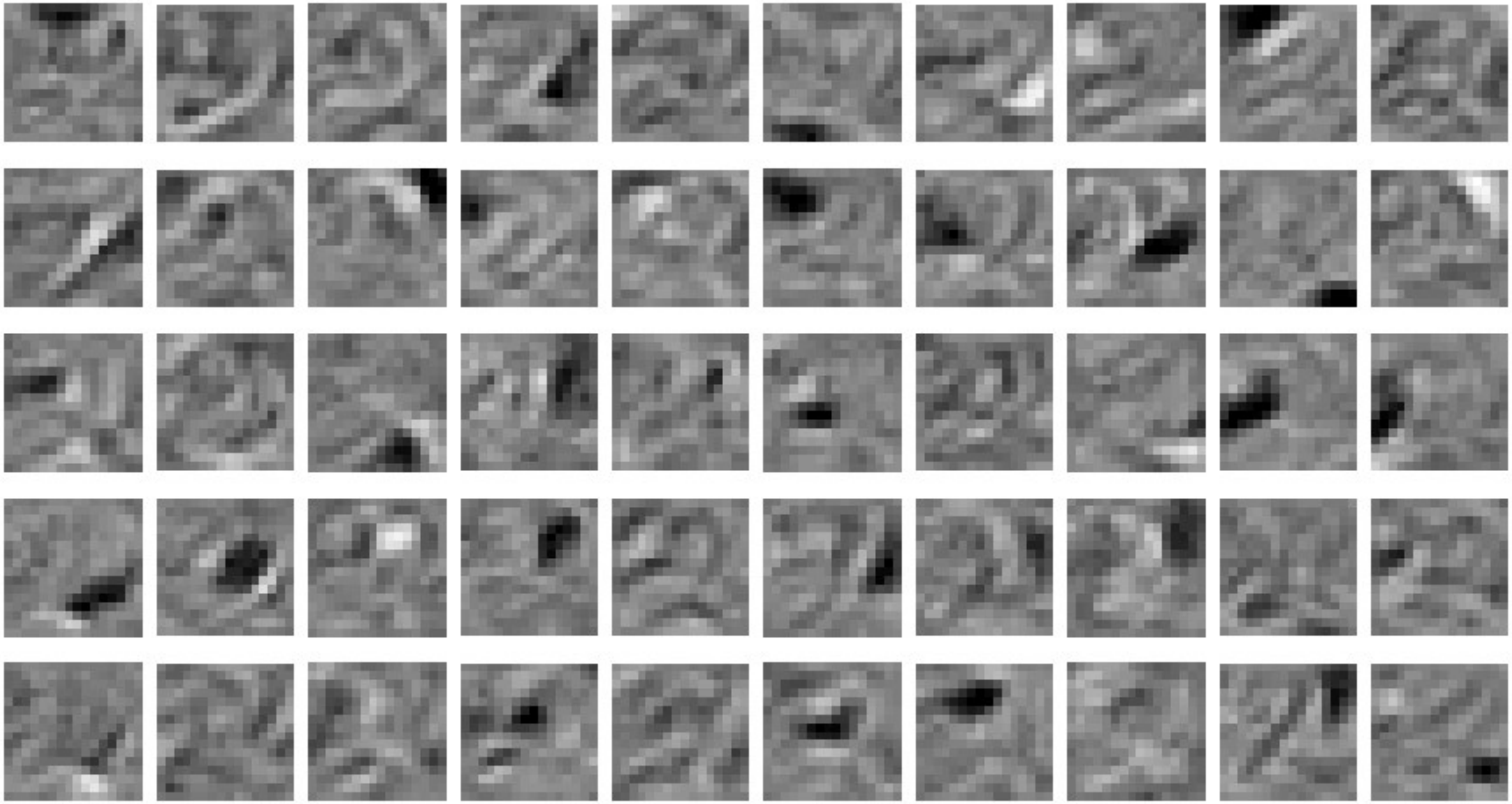
16 x 16 pixel image

data
(reality)

reconstruction
(better than reality)

From Hinton 2007

The final 50 x 256 weights



Each neuron grabs a different feature.

From Hinton 2007

Training a deep network

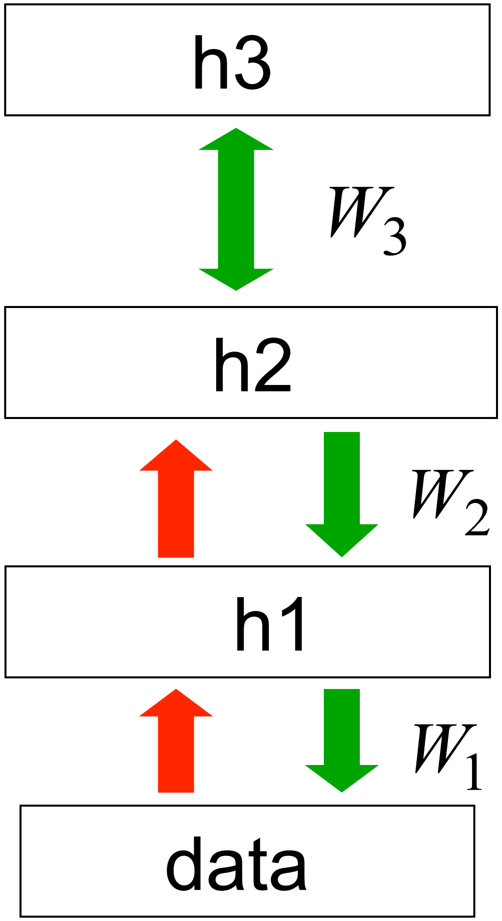
- First train a layer of features that receive input directly from the pixels.
- Then treat the activations of the trained features as if they were pixels and learn features of features in a second hidden layer.
- It can be proved that each time we add another layer of features we improve a variational lower bound on the log probability of the training data.
  - The proof is slightly complicated.
  - But it is based on a neat equivalence between an RBM and a deep directed model (described later)

From Hinton 2007

The generative model after learning 3 layers

- To generate data:
  1. Get an equilibrium sample from the top-level RBM by performing alternating Gibbs sampling.
  2. Perform a top-down pass to get states for all the other layers.

  So the lower level bottom-up connections are not part of the generative model. They are just used for inference.



From Hinton 2007

Why does greedy learning work?

The weights, W, in the bottom level RBM define p(v|h) and they also, indirectly, define p(h).

So we can express the RBM model as

$$p(v) = \sum_{h} p(h)\ p(v \mid h)$$

If we leave p(v|h) alone and improve p(h), we will improve p(v).

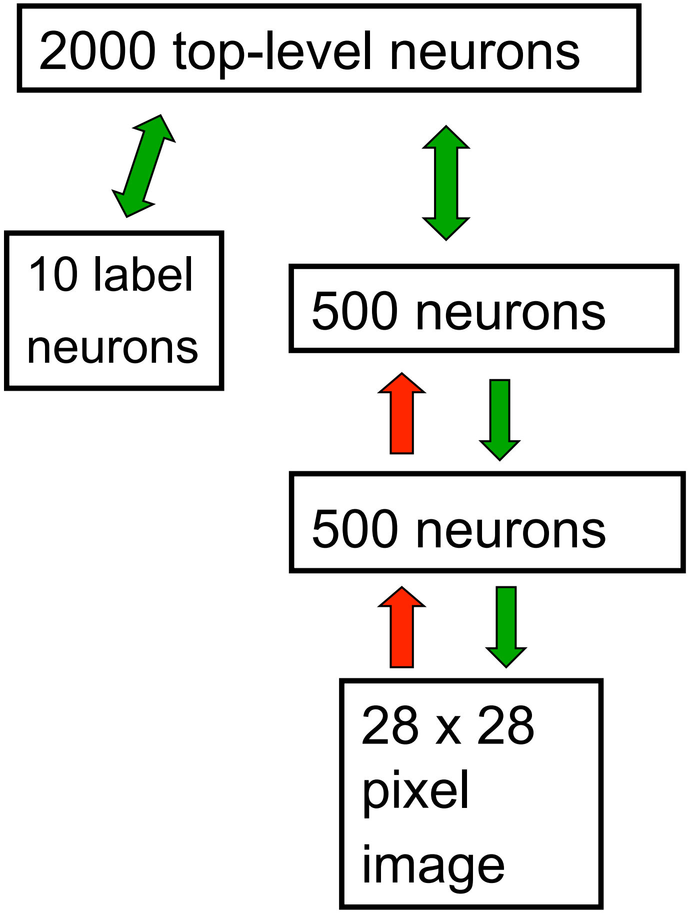
To improve p(h), we need it to be a better model of the aggregated posterior distribution over hidden vectors produced by applying W to the data.

From Hinton 2007

A neural model of digit recognition

The top two layers form an associative memory whose energy landscape models the low dimensional manifolds of the digits.

The energy valleys have names



2000 top-level neurons

10 label neurons

500 neurons

500 neurons

28 x 28 pixel image

The model learns to generate combinations of labels and images.

To perform recognition we start with a neutral state of the label units and do an up-pass from the image followed by a few iterations of the top-level associative memory.

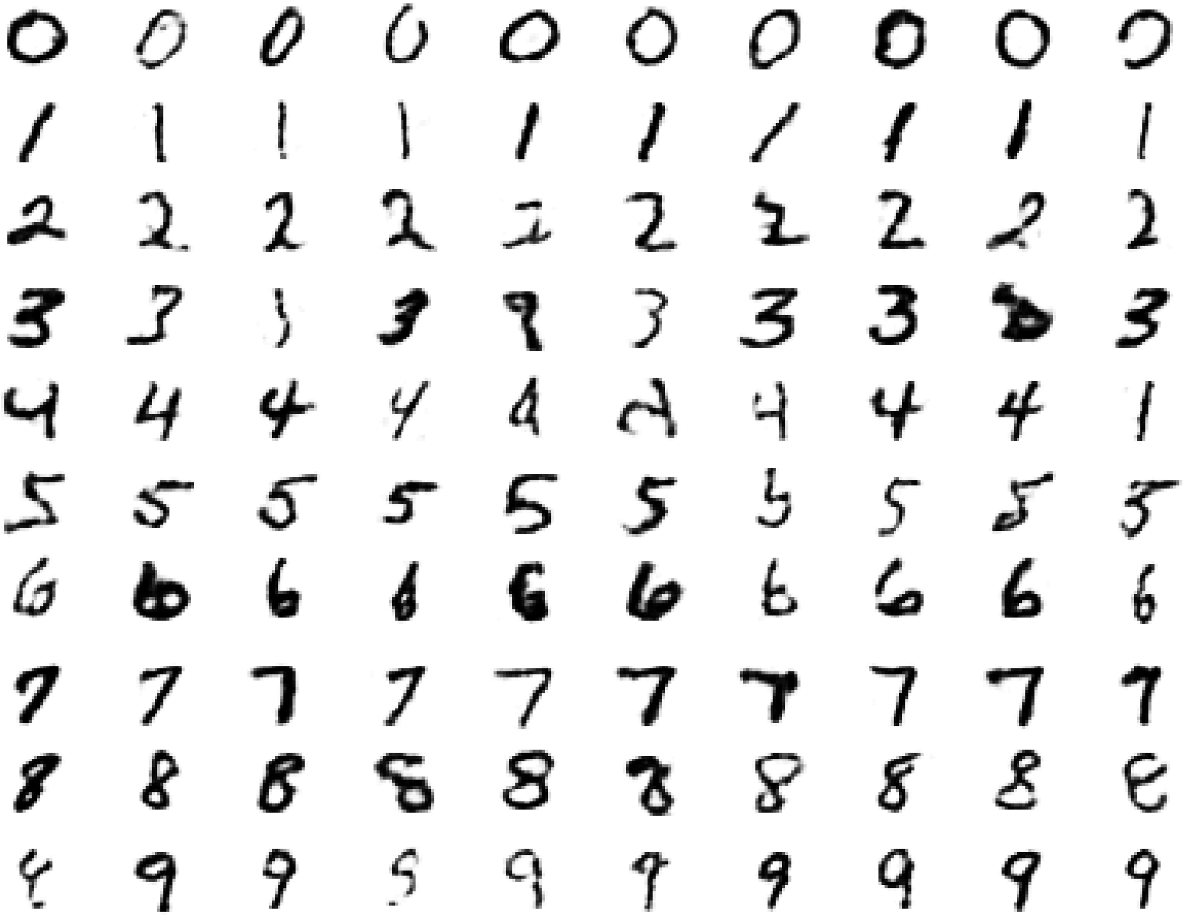
From Hinton 2007

Fine-tuning with a contrastive divergence version of the “wake-sleep” algorithm

- After learning many layers of features, we can fine-tune the features to improve generation.
- 1. Do a stochastic bottom-up pass
  - o Adjust the top-down weights to be good at reconstructing the feature activities in the layer below.
- 2. Do a few iterations of sampling in the top level RBM
  - o Use CD learning to improve the RBM
- 3. Do a stochastic top-down pass
  - o Adjust the bottom-up weights to be good at reconstructing the feature activities in the layer above.

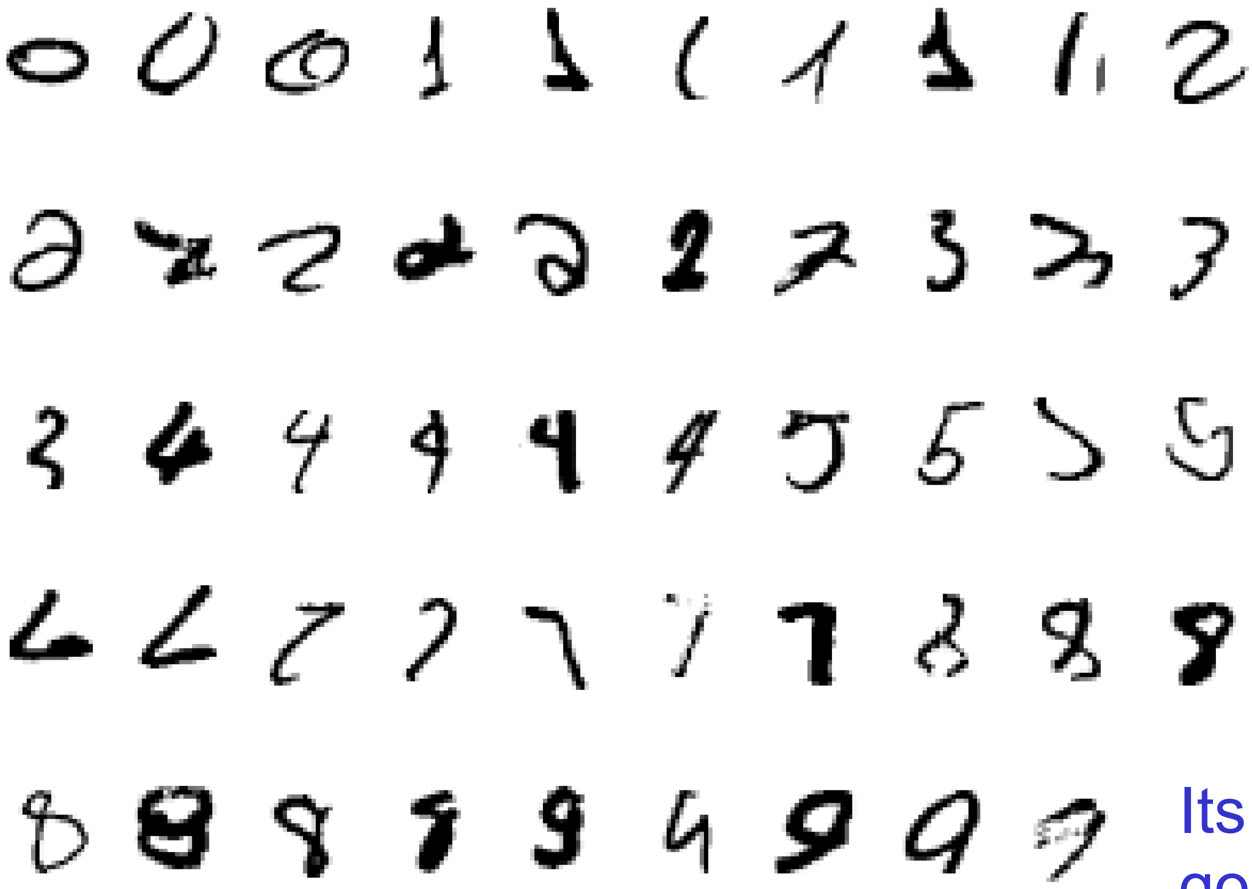
From Hinton 2007

Samples generated by letting the associative memory run with one label clamped. There are 1000 iterations of alternating Gibbs sampling between samples.



From Hinton 2007

Examples of correctly recognized handwritten digits that the neural network had never seen before



From Hinton 2007

Its very good

How well does it discriminate on MNIST test set with no extra information about geometric distortions?

• Generative model based on RBM’s	1.25%
• Support Vector Machine (Decoste et. al.)	1.4%
• Backprop with 1000 hiddens (Platt)	~1.6%
• Backprop with 500 -->300 hiddens	~1.6%
• K-Nearest Neighbor	~ 3.3%

- Its better than backprop and much more neurally plausible because the neurons only need to send one kind of signal, and the teacher can be another sensory input.

From Hinton 2007

Summary

- Examined an application using RBMs as deep belief networks to recognize handwritten digits.
- Showed an efficient approximation—*contrastive divergence*—of the derivative of the log of the probability.
  - Involved calculating the average of <v,h> after the initial presentation, and
  - Calculating the average of <v,h> after the first reconstruction.
  - Using their difference multiplied by a learning parameter as the basis of a gradient ascent scheme.