Liu Xiao

DEEP BELIEF NETWORK

It has been obvious since the 1980s that backpropagation through deep autoencoders would be very effective for nonlinear dimensionality reduction, provided that **computers were fast enough**, **data sets were big enough**, **and the initial weights were close enough** to a good solution.

All three conditions are now satisfied.

———————— Hinton . 2006.

WHAT IS DBN?

Hinton, G. E., Osindero, S. and Teh, Y. (2006)

A fast learning algorithm for deep belief nets.

Neural Computation, 18, pp 1527-1554.

G. E. Hinton* and R. R. Salakhutdinov .(2006)

Reducing the Dimensionality of Data with Neural Networks

Science, 2006, 313(5786): 504-507.

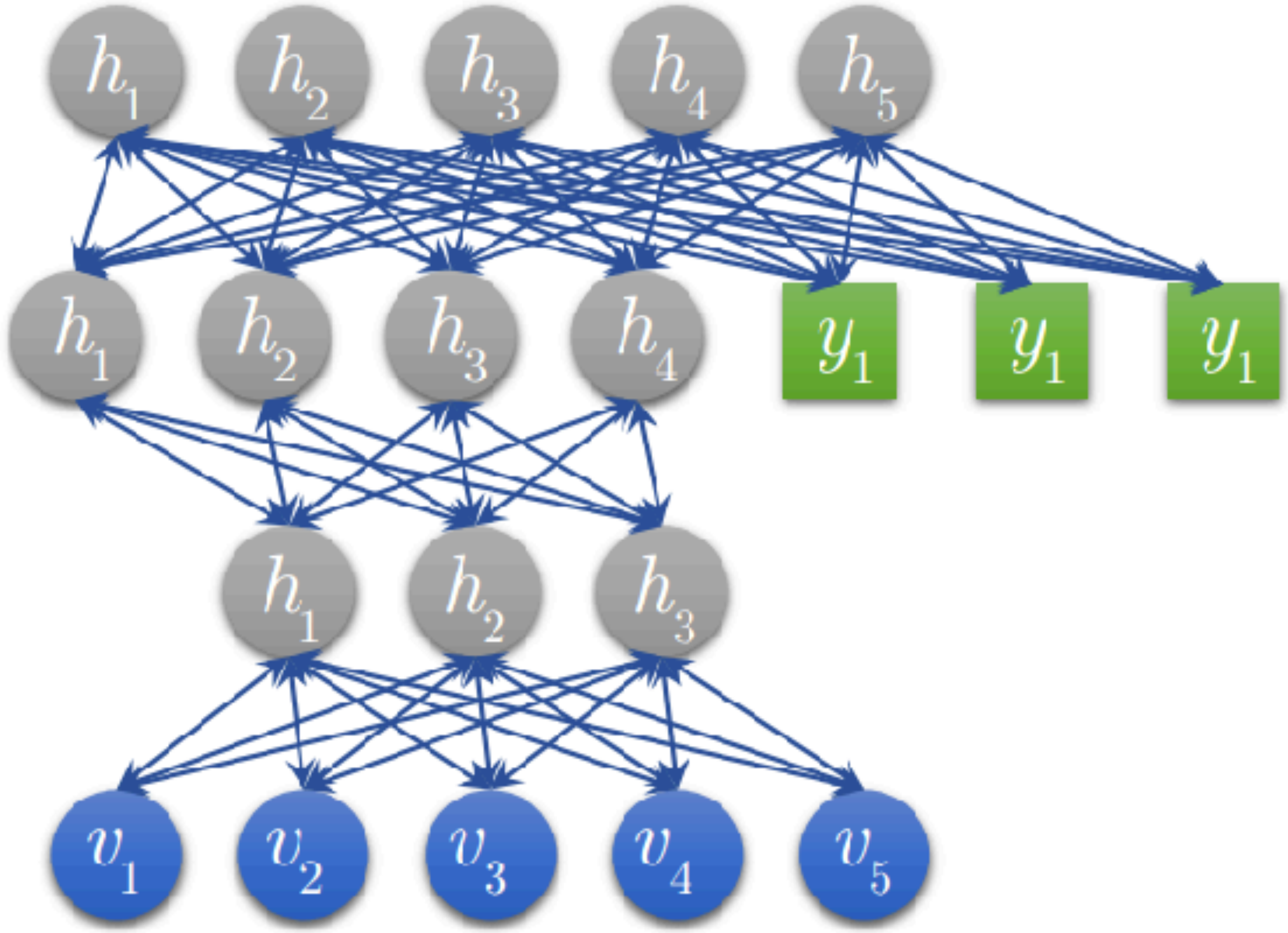
QUESTIONS

1. What is Deep Belief Network?

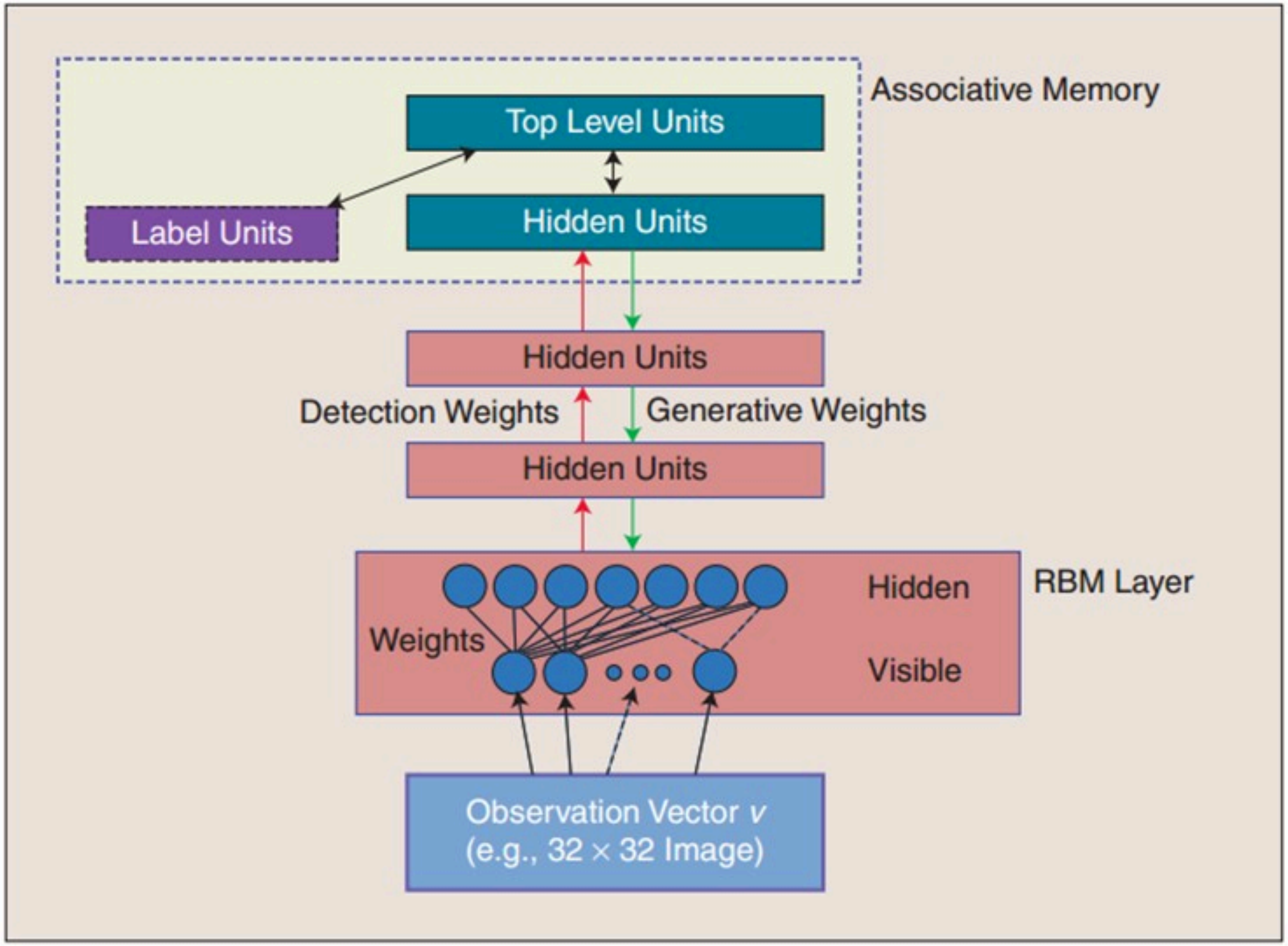
2. How it works?

WHAT IS DBN?

DBNs are **stacks** of **restricted Boltzmann machines** forming **deep (multi-layer)** architecture.



WHAT IS DBN?

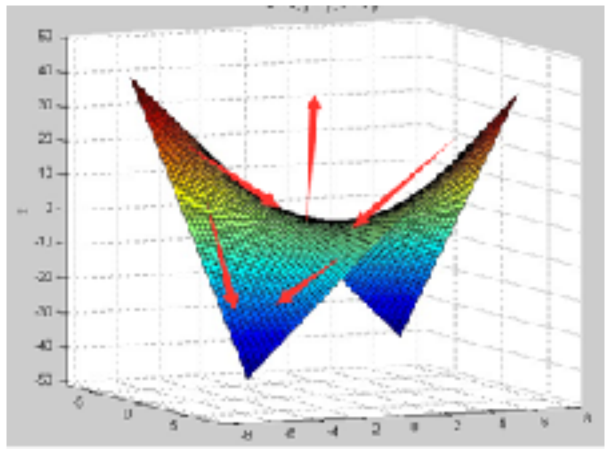

Associative Memory
Top Level Units
Label Units
Hidden Units
Hidden Units
Detection Weights
Generative Weights
Hidden Units
Hidden
RBM Layer
Weights
Visible
Observation Vector v
(e.g., 32 × 32 Image)

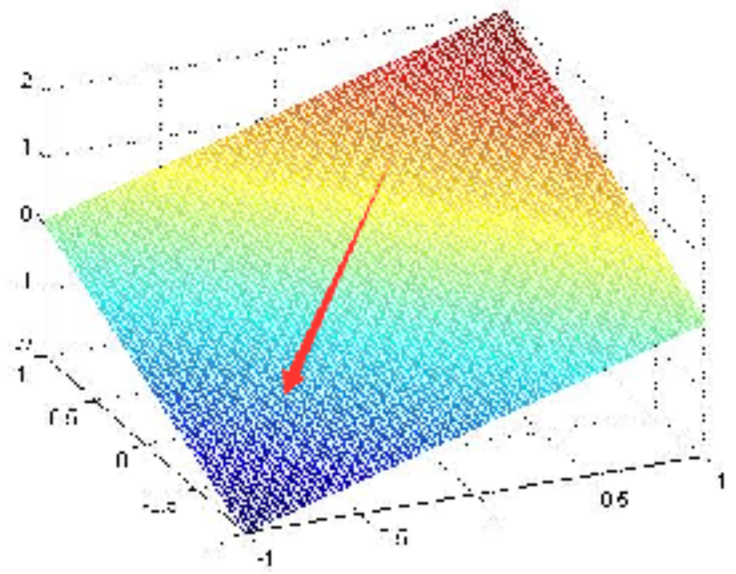
1. Problems with deep networks?

2. How DBN solve them?

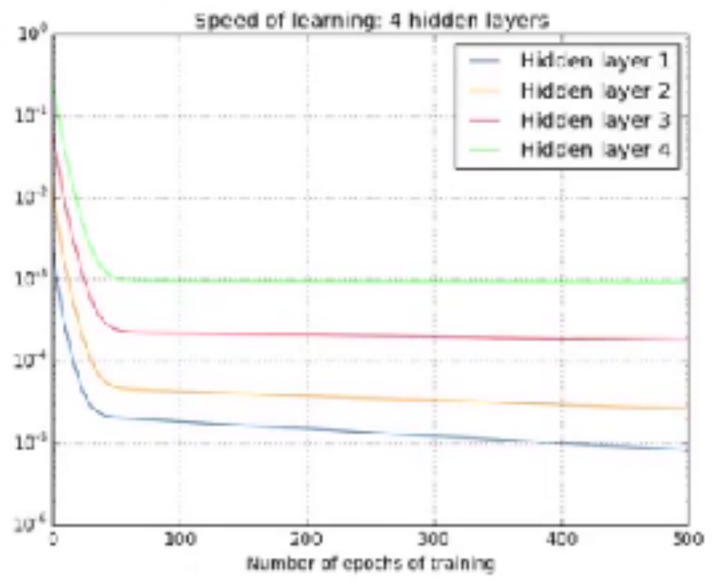
PROBLEM WITH DEEP

1.NonConvex Optimization

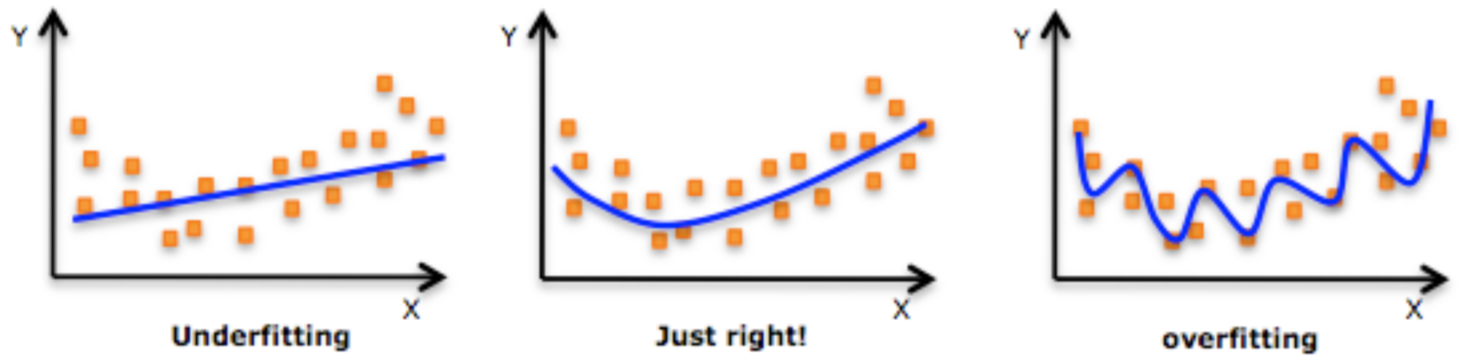




2.Gradient Vanish



3.Overfitting



HOW DBN WORKS

Hinton proposed greedy **unsupervised layer-wise training**:

• Greedy layer-wise: Train layers sequentially starting from bottom (input) layer.（逐层向上学习）

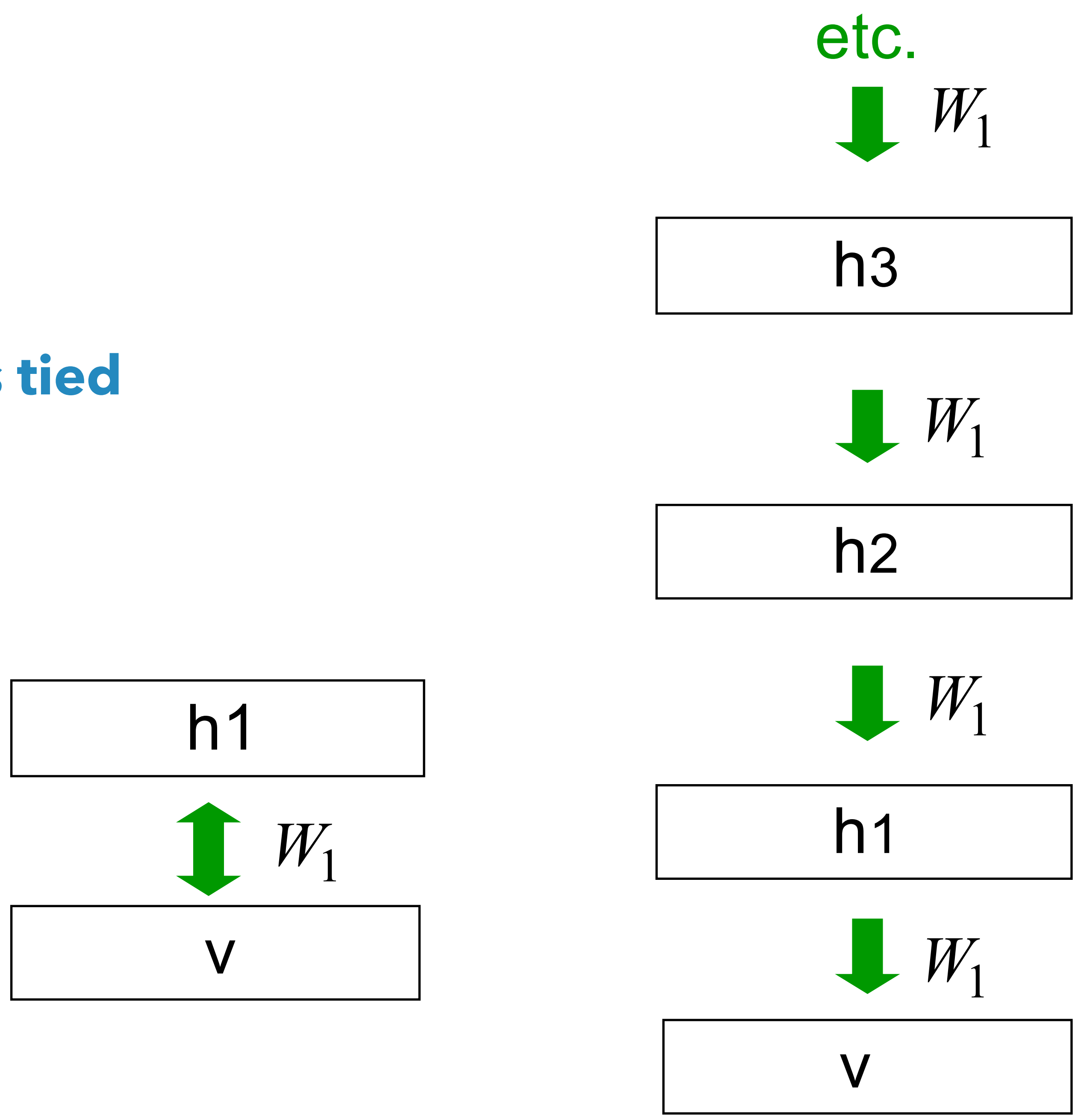
• Unsupervised: Each layer learns a higher-level representation of the layer below. The training criterion does not depend on the labels.（无监督）

• Each layer is trained as a Restricted Boltzman Machine. (RBM is the building block of Deep Belief Networks).（训练单个RBM）

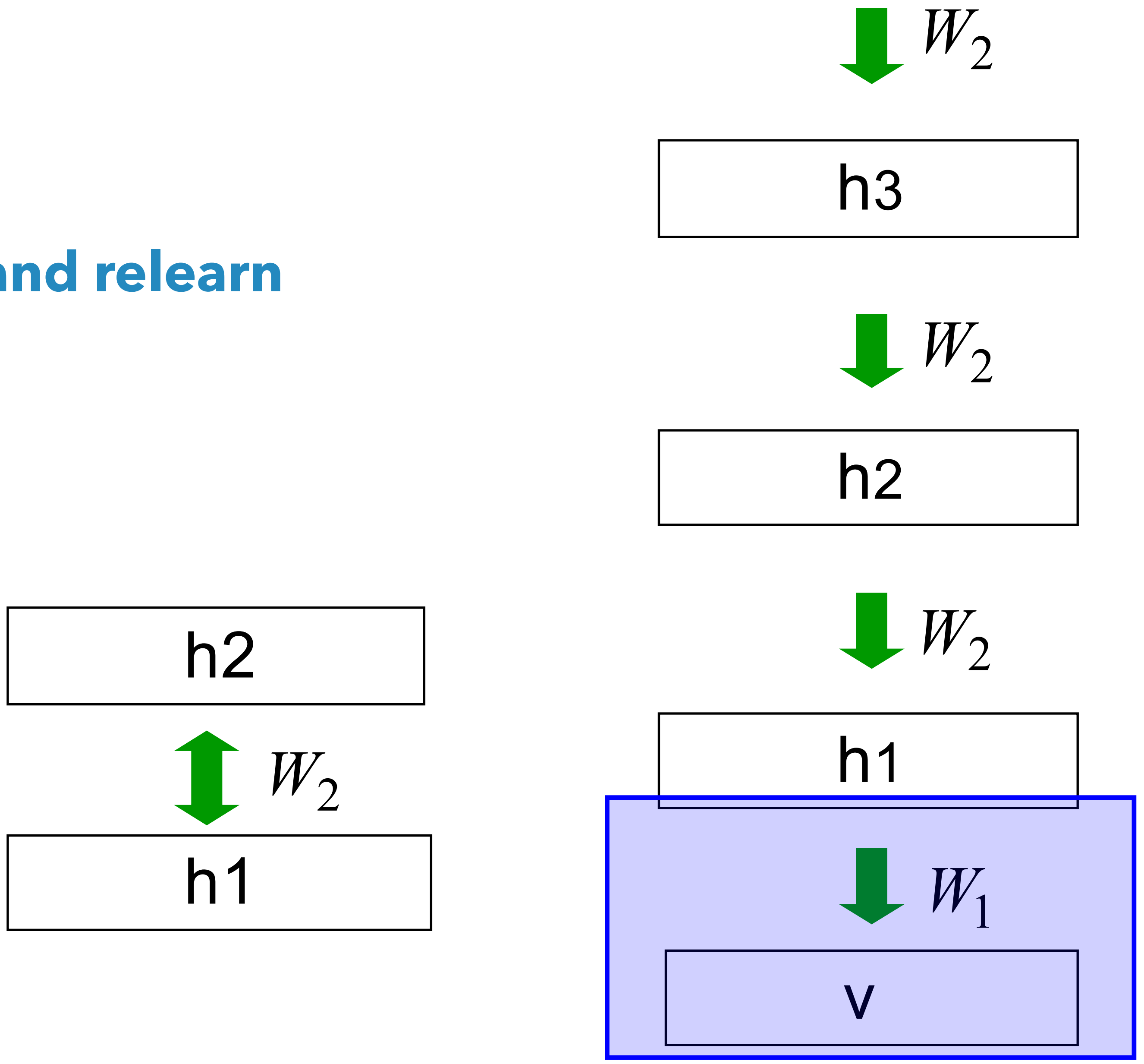
• The trained model can be fine tuned using a supervised method.（有监督调优）

HOW IT WORKS

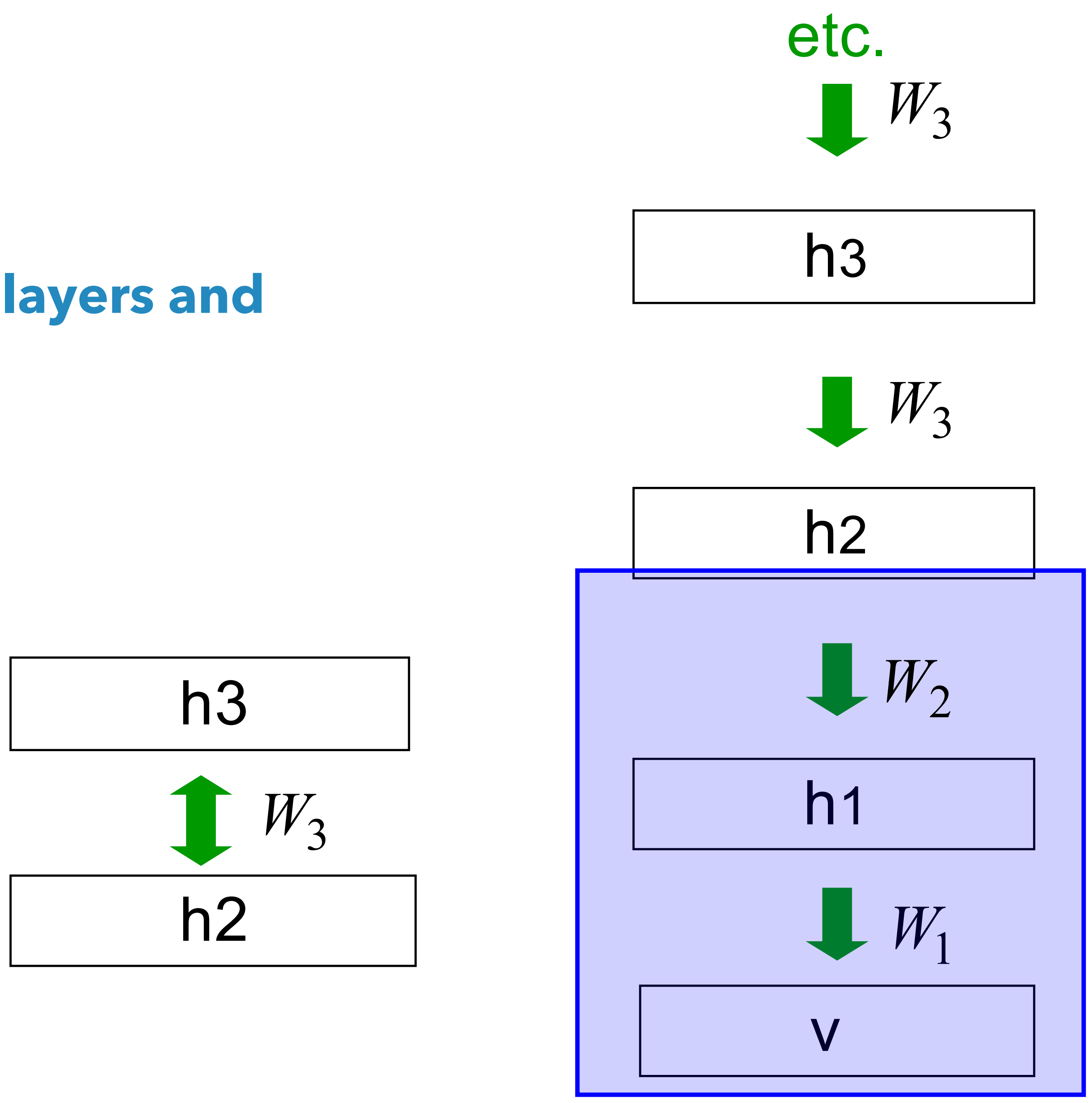
1. First learn with all the weights tied



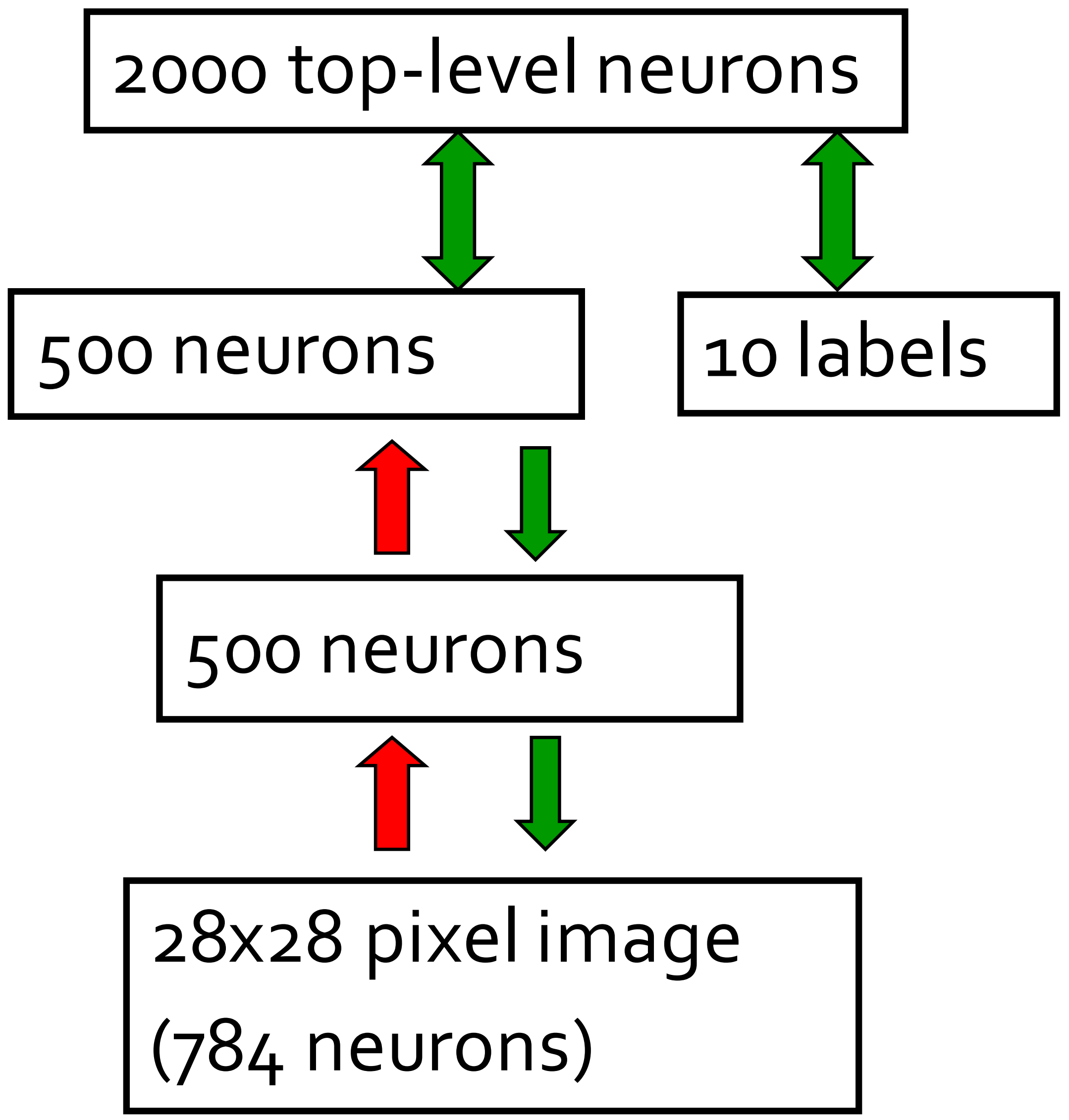
2 .Then freeze the bottom layer and relearn all the other layers.



3 .Then freeze the bottom two layers and relearn all the other layers.



HOW IT WORKS



1.The first two hidden layers are learned without using labels

2.The top layers is learned as an RBM for modeling the labels concatenated with features in the second hidden layer.

3.The weight are the fine-tuned to be a better generative model using wake-sleep

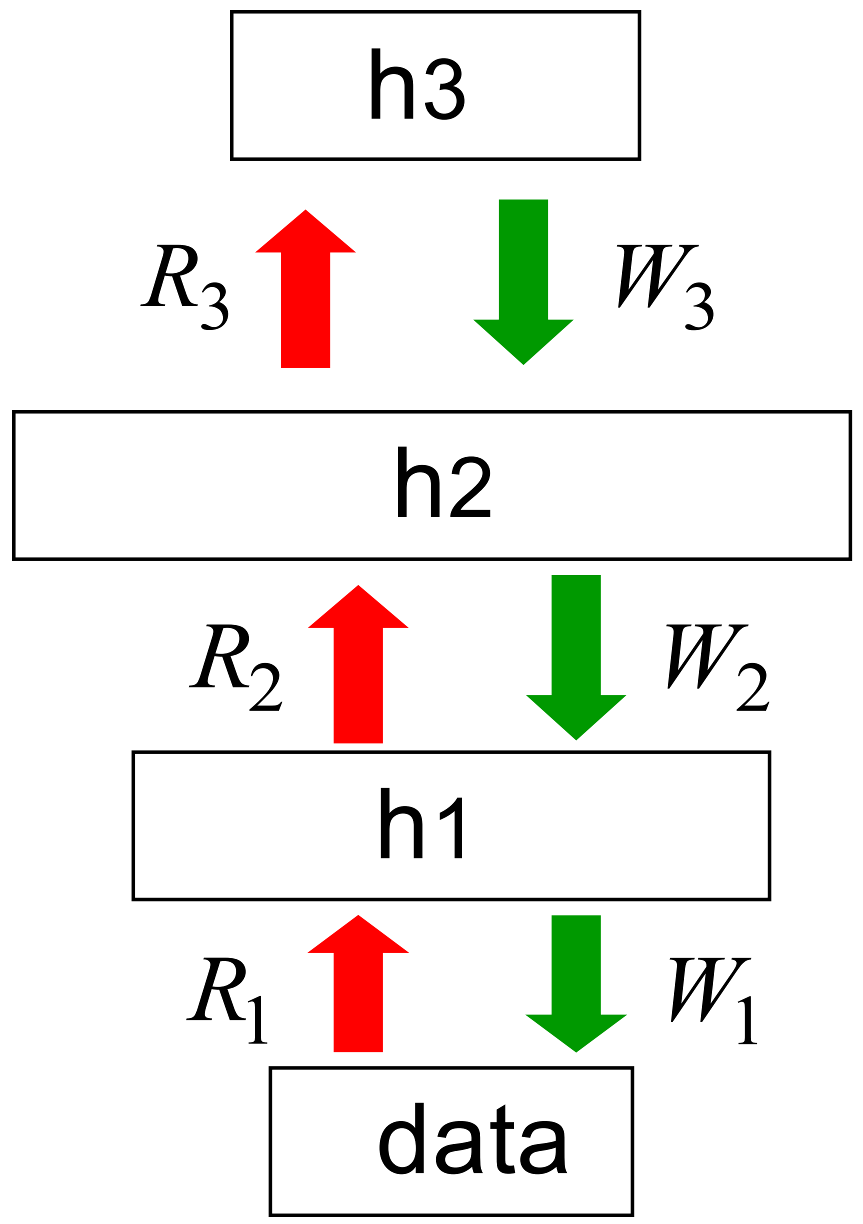
The wake-sleep algorithm

Wake phase: Use the recognition weights to perform a bottom-up pass.

Train the generative weights to reconstruct activities in each layer from the layer above.

Sleep phase: Use the generative weights to generate samples from the model.

Train the recognition weights to reconstruct activities in each layer from the layer below.



Why the hidden configurations should be treated as data when learning the next layer of weights

After learning the first layer of weights:

$$\log p(x) \geq -<energy(x)> \quad + \quad entropy(h1|x)$$
$$\geq \sum_{\alpha} p(h1=\alpha \mid x)\left[\log p(h1=\alpha) + \log p(x \mid h1=\alpha)\right] + entropy$$

If we freeze the generative weights that define the likelihood term and the recognition weights that define the distribution over hidden configurations, we get:

$$\log p(x) \geq \sum_{\alpha} p(h1=\alpha \mid x)\left[\log p(h1=\alpha)\right] + constant$$

Maximizing the RHS is equivalent to maximizing the log prob of “data” α that occurs with probability $p(h1=\alpha \mid x)$

Why greedy learning works?

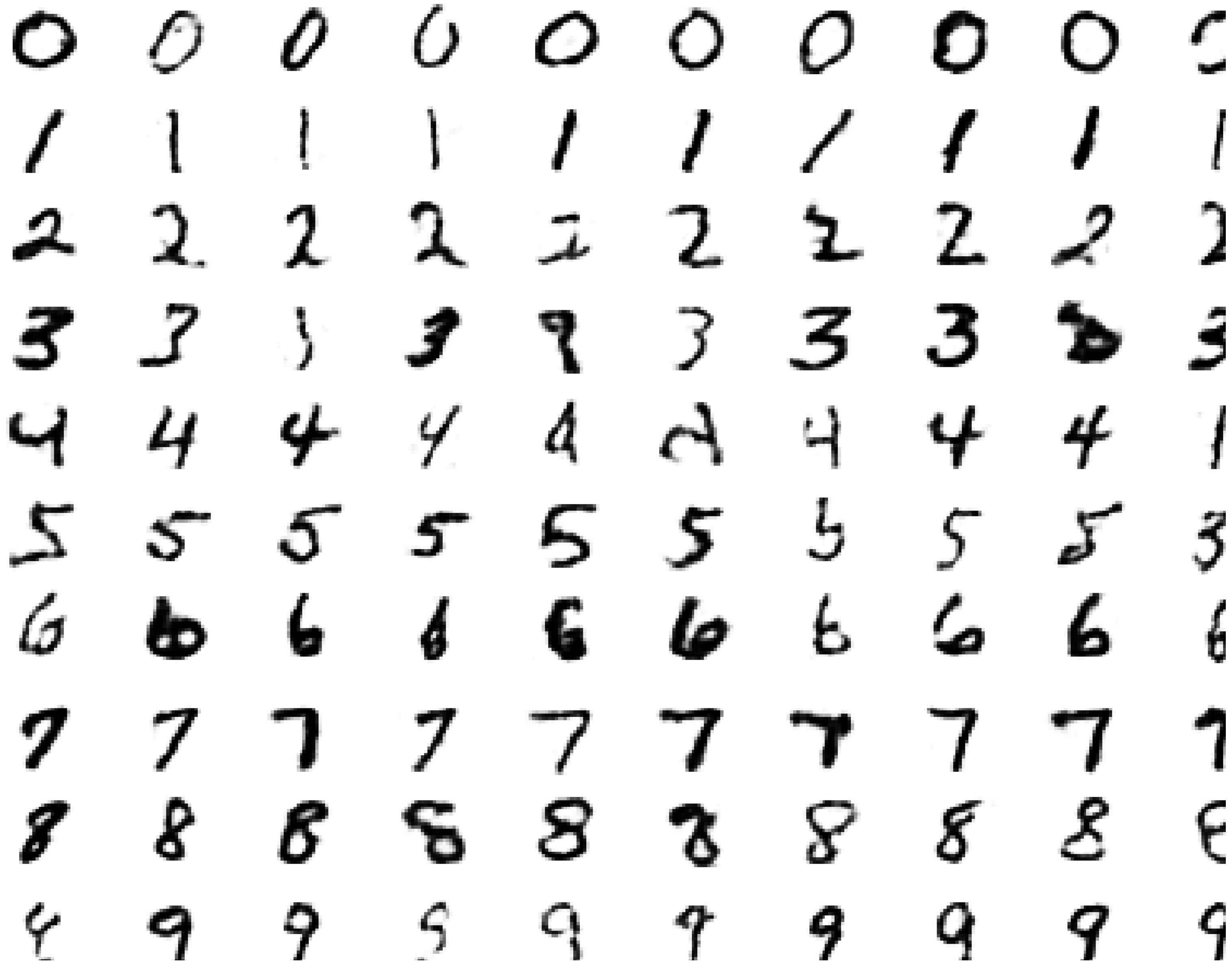
- Each time we learn a new layer, the inference at the layer below becomes incorrect, but the variational bound on the log prob of the data improves provided we start the learning from the tied weights that implement the complementary prior.

- Now that we have a guarantee we can loosen the restrictions and still feel confident.

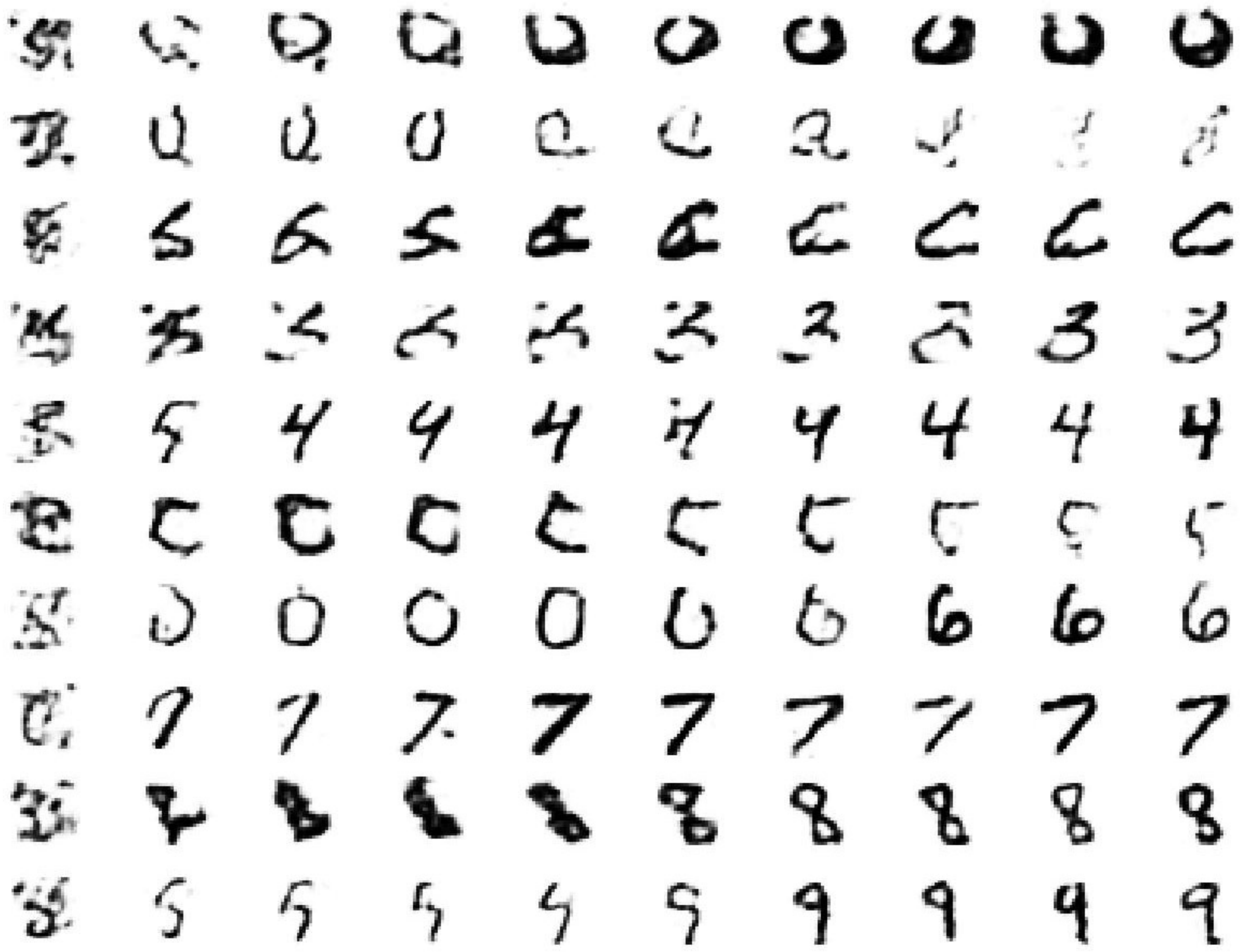
  Allow layers to vary in size.

  Do not start the learning at each layer from the weights in the layer below

Samples generated by letting the associative memory run with one label clamped.



Providing a random binary image as input



DeepLearning ToolBox

Includes Deep Belief Nets, Stacked Autoencoders, Convolutional Neural Nets, Convolutional Autoencoders and vanilla Neural Nets.

https://github.com/rasmusbergpalm/DeepLearnToolbox

Matlab演示DBN

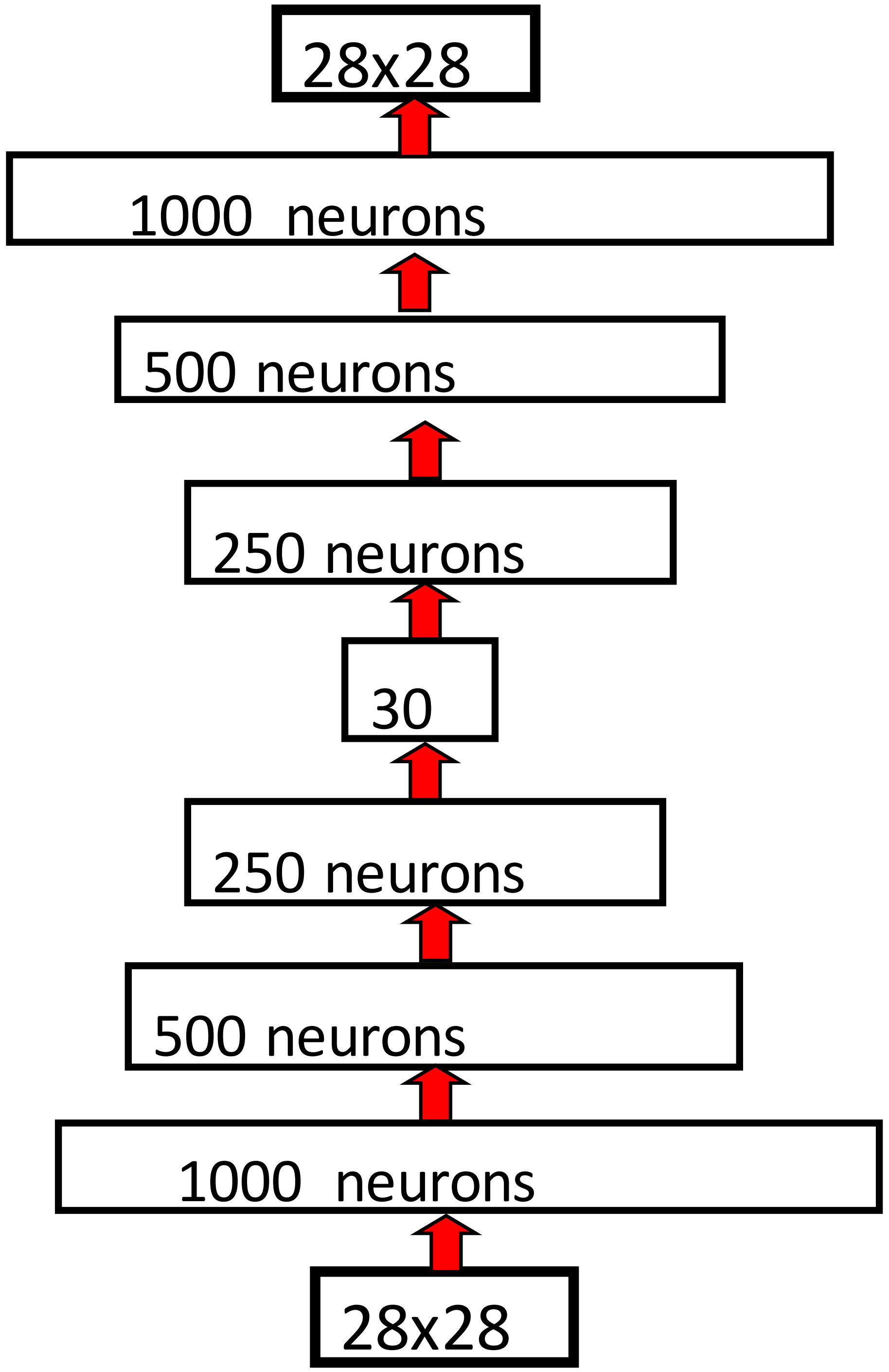
REDUCING THE DIMENSIONALITY OF DATA

They always looked like a really nice way to do non-linear dimensionality reduction:

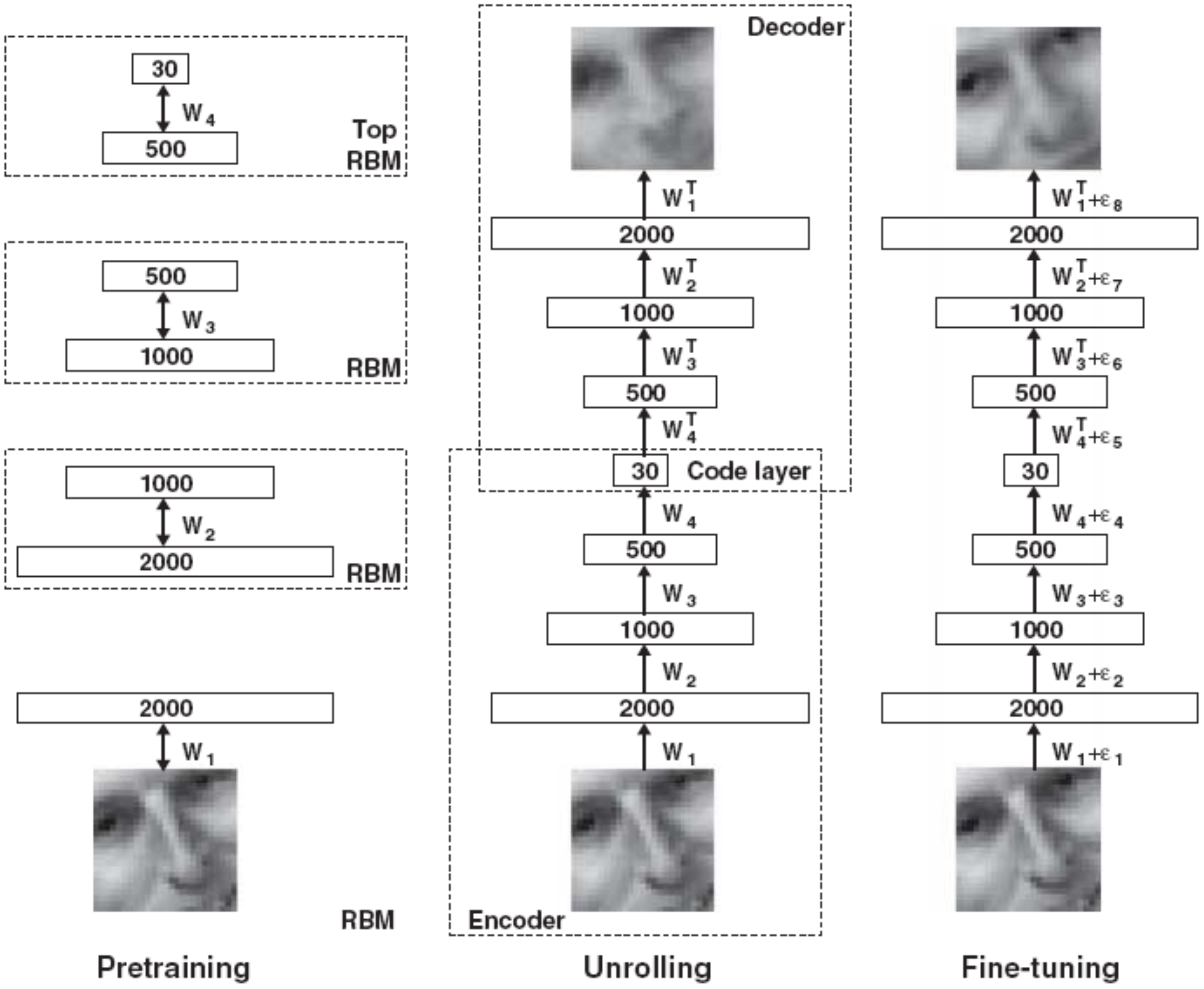
But it is very difficult to optimize deep autoencoders using backpropagation.

We now have a much better way to optimize them:

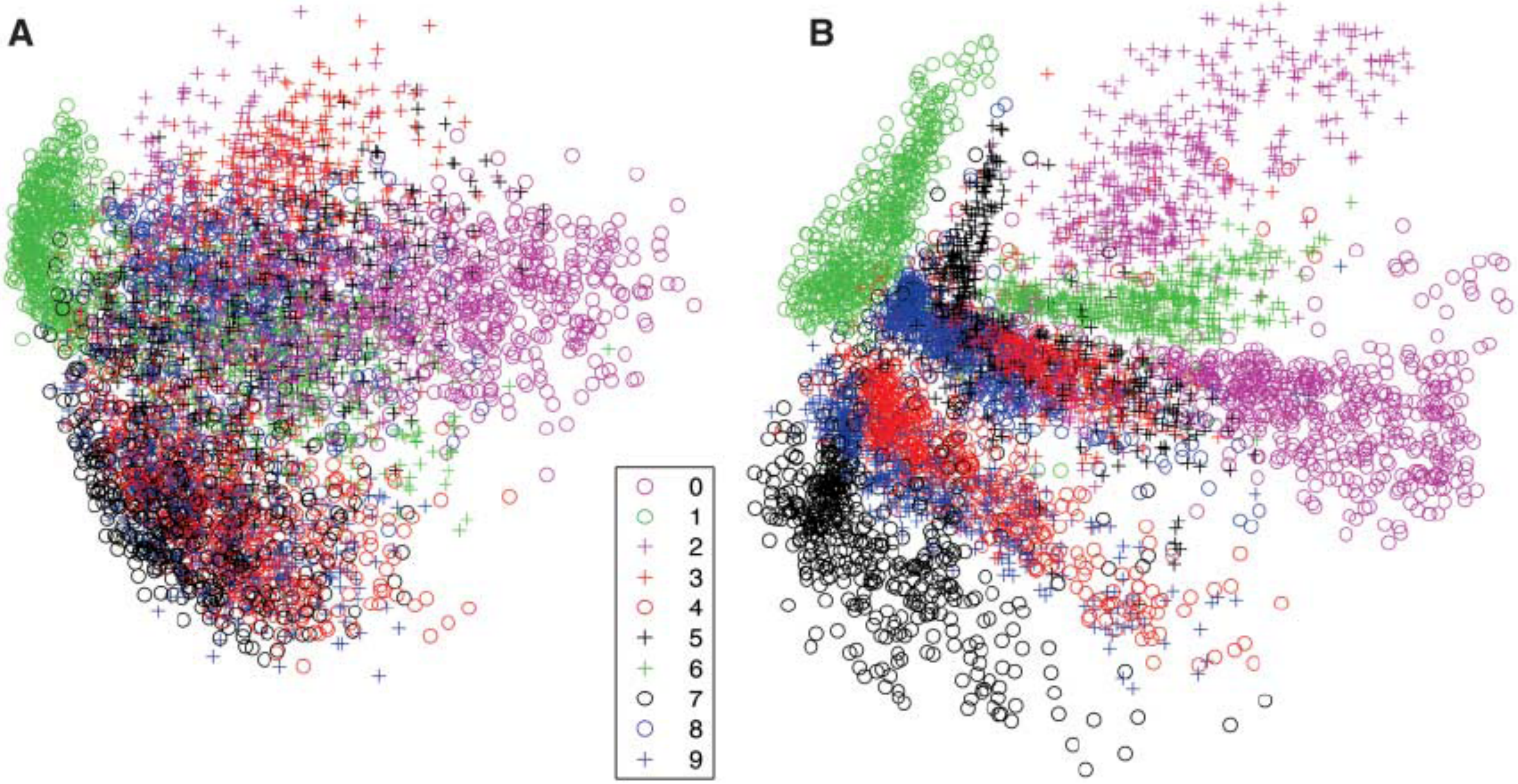
1. First train a stack of 4 RBM's
2. Then "unroll" them.
3. Then fine-tune with backpropagation



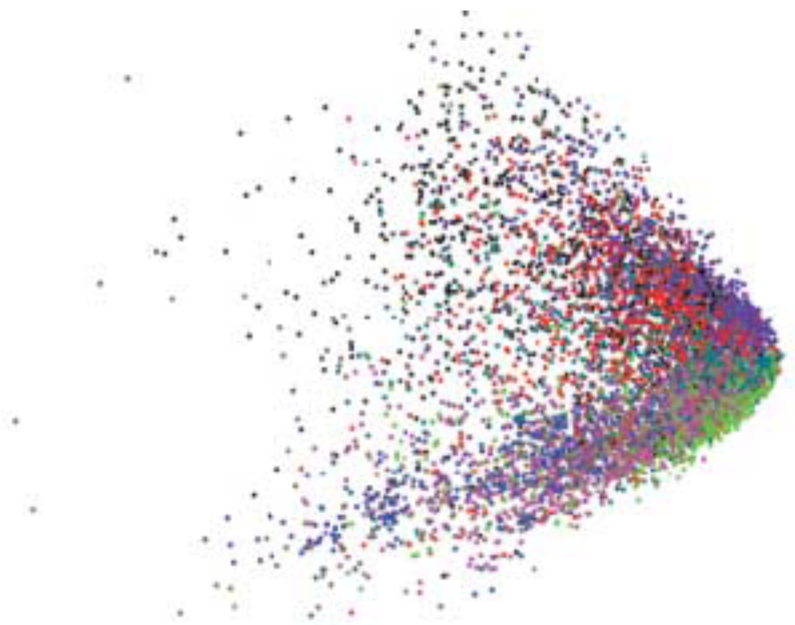
LEARNING STEP

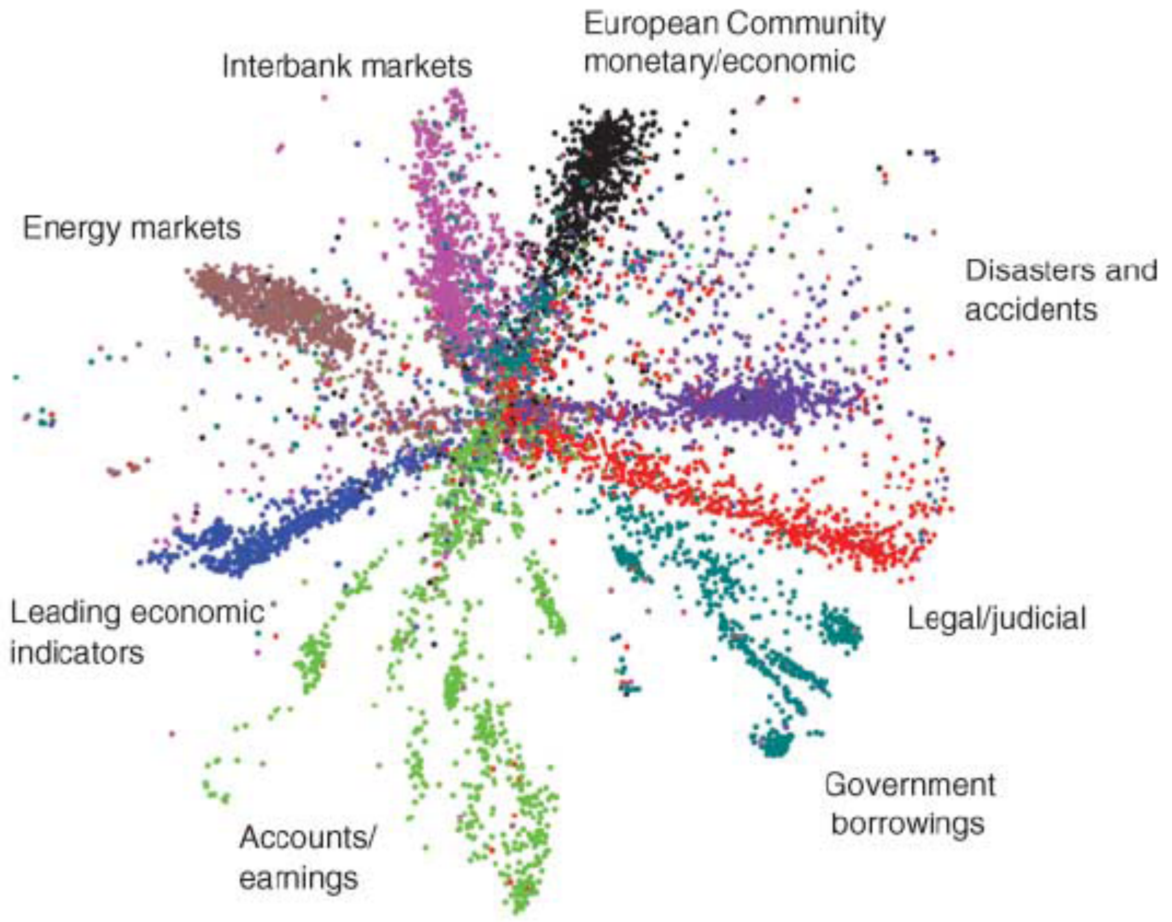


AUTOENCODER VS. PCA



AUTOENCODER VS. LSA





Thank You!