united nations educational, scientific and cultural organization

international atomic energy agency

the
abdus salam
international centre for theoretical physics

SMR: 1343/9

EU ADVANCED COURSE IN COMPUTATIONAL NEUROSCIENCE

An IBRO Neuroscience School

(30 July - 24 August 2001)

"An Introduction to Artificial Neural Networks and Learning Theory"

presented by:

Zoubin GHAHRAMANI

Gatsby Computational Neuroscience Unit
University College London
Alexandra House, 17 Queen Square
London, WC1N 3AR
U.K.

These are preliminary lecture notes, intended only for distribution to participants.

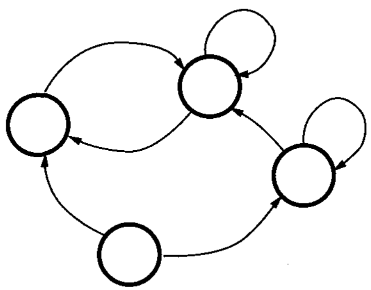
An Introduction to Artificial Neural Networks and Learning Theory

Zoubin Ghahramani

Gatsby Computational Neuroscience Unit
University College London
http://www.gatsby.ucl.ac.uk/

Trieste, August 2001

Two Views of Modelling



- Given some network of neurons, how does it behave?
  - biology, physics
- Given some computational problem, how can it be solved with neurons?
  - computer science, engineering

View: To understand how the brain functions we need to understand the computational problems it solves (Marr).

Problems: e.g. vision, audition, olfaction, linguistic communication, decision making, movement control, ... *learning*

Three Types of Learning

Imagine an organism or machine that experiences a series of sensory inputs:

$$x_1, x_2, x_3, x_4, \ldots$$

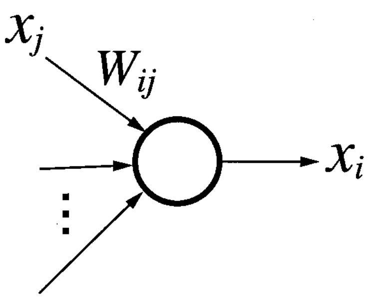
Supervised learning: The machine is also given desired outputs $y_1, y_2, \ldots$, and its goal is to learn to produce the correct output given a new input.

Unsupervised learning: The goal of the machine is to build representations from x that can be used for reasoning, decision making, predicting things, communicating etc.

Reinforcement learning: The machine can also produce actions $a_1, a_2, \ldots$ which affect the state of the world, and receives rewards (or punishments) $r_1, r_2, \ldots$. Its goal is to learn to act in a way that maximises rewards in the long term.

Binary Hopfield Networks

McCulloch-Pitts Neurons:

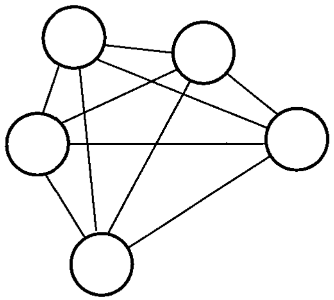


$$x_i \to 0 \quad \text{if} \quad \sum_{j \neq i} W_{ij} x_j < \theta$$
$$x_i \to 1 \quad \text{if} \quad \sum_{j \neq i} W_{ij} x_j \geq \theta$$

Binary neuron with threshold θ, activities x_i, and connections W_{ij}.

Binary Hopfield Networks:

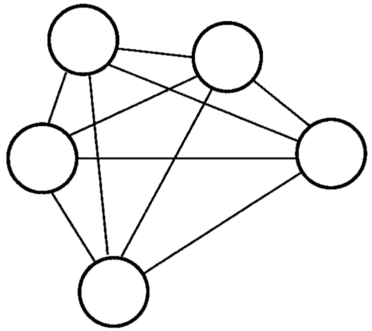
Recurrent network of symmetrically connected McCulloch-Pitts neurons



Q: Can you store information (memories) in these nets?

Q: What happens if you run these dynamics? Stable?

Storing Associative Memories in a Hopfield Network



Storage Rule: $$W_{ij} = \sum_n (2x_i^n - 1)(2x_j^n - 1)$$

Hebb learning rule: $$\Delta W_{ij} = \langle x_i x_j \rangle \quad \text{(possibly with decay)}$$

Activation rule: $$x_i \rightarrow 1 \text{ if } \sum_{j \neq i} w_{ij} x_j \geq 0$$

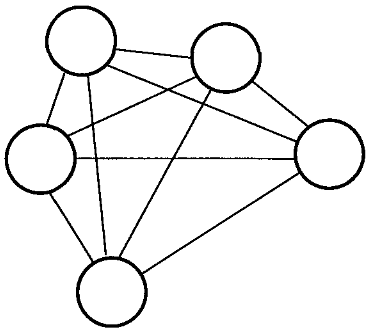
Given pattern x^m: $$\sum_j W_{ij} x_j^m = \sum_n (2x_i^n - 1) \left[\sum_j (2x_j^n - 1) x_j^m \right]$$

For random uncorrelated patterns, on average $[\sum_j (2x_j^n - 1)x_j^m] = 0$ except if $n = m$ in which case it's $D/2$.

$$\sum_j W_{ij} x_j^m \approx (2x_i^m - 1)\frac{D}{2} \qquad \begin{cases} \geq 0 & \text{if } x_i^m > 0 \\ < 0 & \text{if } x_i^m < 0 \end{cases}$$

i.e. x^m is a stable pattern (i.e. a memory)

Asynchronous Hopfield Dynamics Converge



Activation Rule:

$$x_i \rightarrow 1 \text{ if } \sum_{j \neq i} W_{ij} x_j > 0,$$

$$x_i \rightarrow 0 \text{ otherwise}$$

Define an Energy Function : $E(x) = -\frac{1}{2} \sum_{i,j \neq i} W_{ij} x_i x_j$

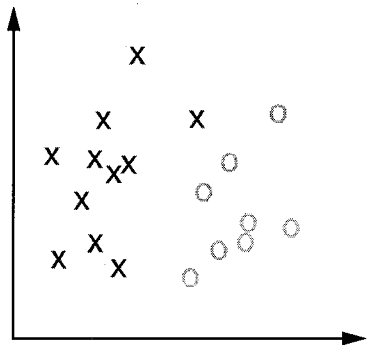
Activation rule decreases energy: $\Delta E = -\Delta x_i \sum_{j \neq i} W_{ij} x_j \leq 0$

E is bounded below, so Hopfield dynamics converge to a stable fixed point.

Problems and limitations with binary Hopfield networks:

- low capacity; slow recall
- complex basins of attraction; spurious memories
- symmetric connections unphysiological
- no hidden units or internal representations

Perceptrons

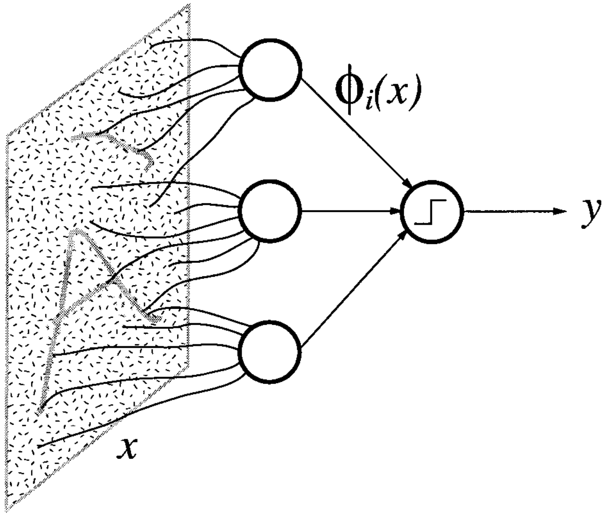


The Classification Problem

Data: $\{(x^n, t^n)\}$ where x^n are input vectors and t^n are class labels:

$t^n = +1$ when $x^n \in C_1$,

$t^n = -1$ when $x^n \in C_2$.



Model:

$$y^n = g\left(\sum_j W_j \phi_j(x^n)\right)$$

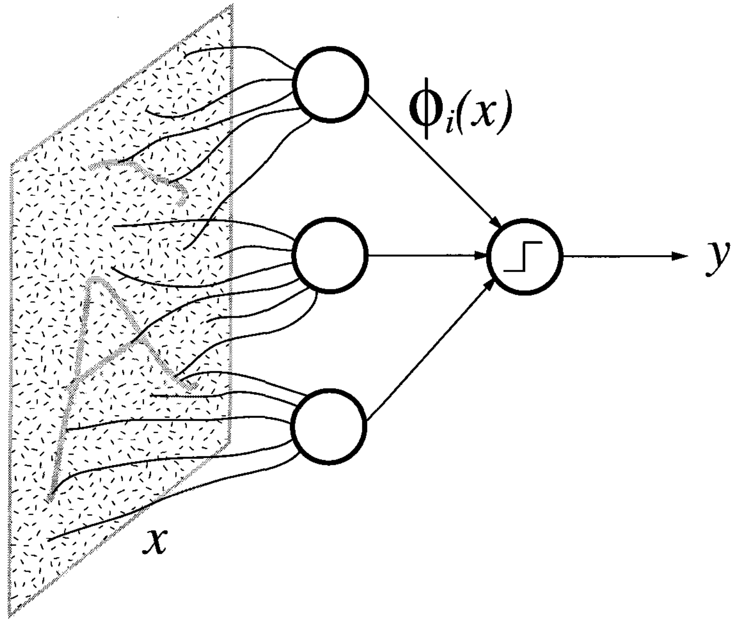
where $\phi_j(x)$ are fixed features (weights connected to the pixels of x with a threshold activation function).

$$g(z) = \begin{cases} -1 & \text{if} \quad z < 0 \\ +1 & \text{if} \quad z \geq 0 \end{cases}$$

Goal: correct classification, i.e. y^n should be equal to t^n on both training data and new data (generalization).

(Rosenblatt 1962; Widrow-Hoff 1960)

The Perceptron Cost Function



Training Data: $\{(x^n, t^n)\}$.

Model: $y^n = g(\sum_j W_j \phi_j(x^n))$

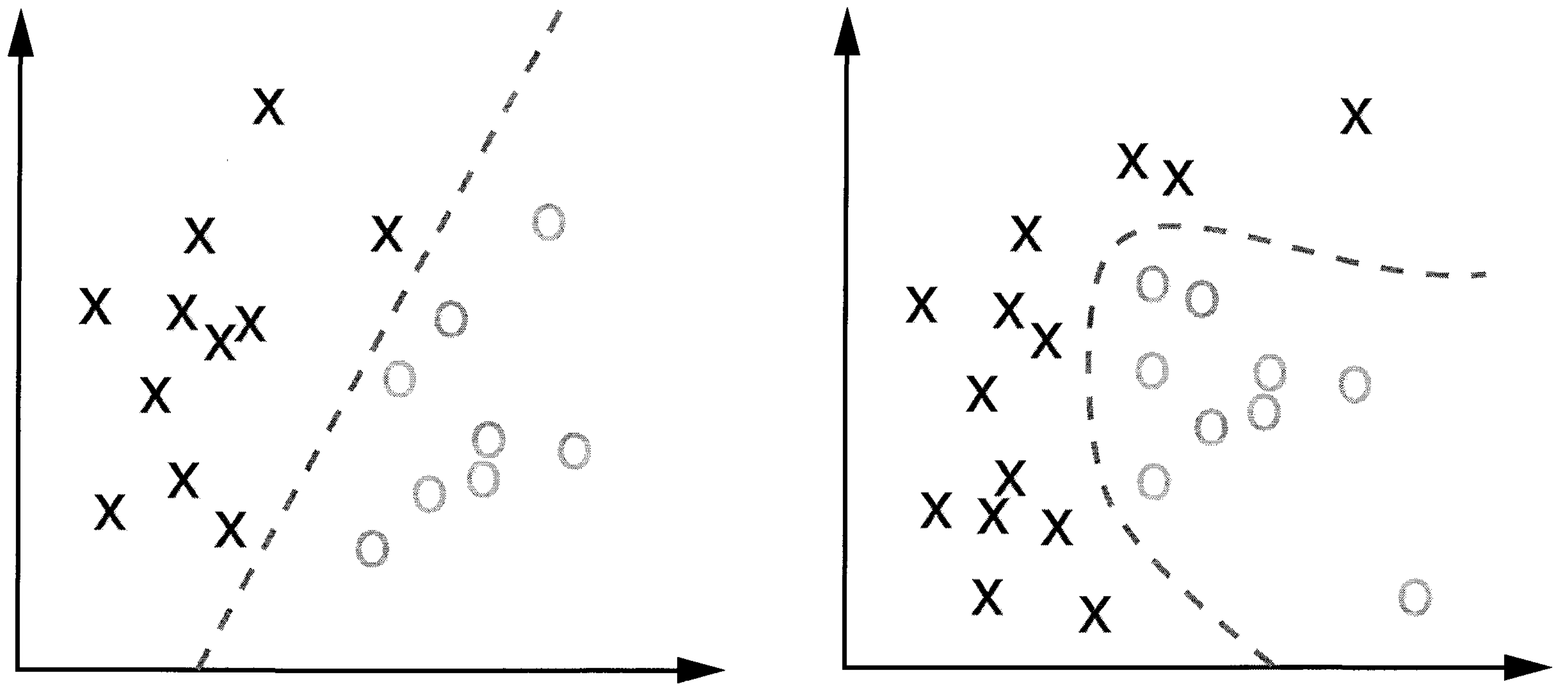
We want correct classification: $\sum_j W_j \phi_j(x^n) > 0$ if $t^n = +1$
and $\sum_j W_j \phi_j(x^n) \leq 0$ if $t^n = -1$

Goal: minimize the cost function: $$E(W) = -\sum_n t^n \sum_j W_j \phi_j(x^n)$$

Learning Rule: $$W_j^{(t+1)} = W_j^{(t)} + \eta\, \phi_j(x^n)\, t^n$$

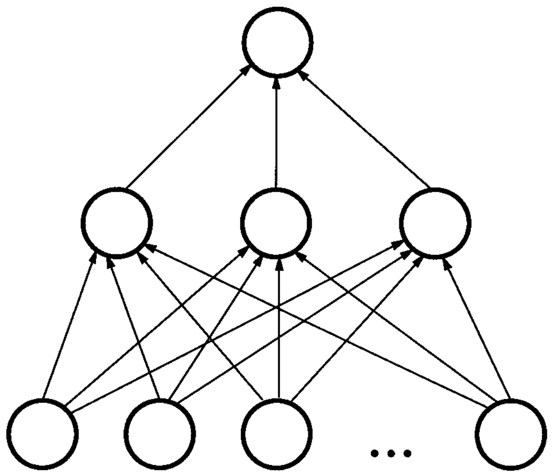
For any data set which is linearly separable this learning rule will find a solution in a finite number of steps.

Linear Separability



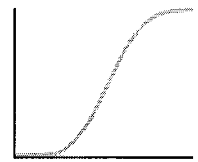
(Minsky and Papert, 1969)

Multi-Layer Perceptrons



Activation function for a unit in layer $\ell + 1$:

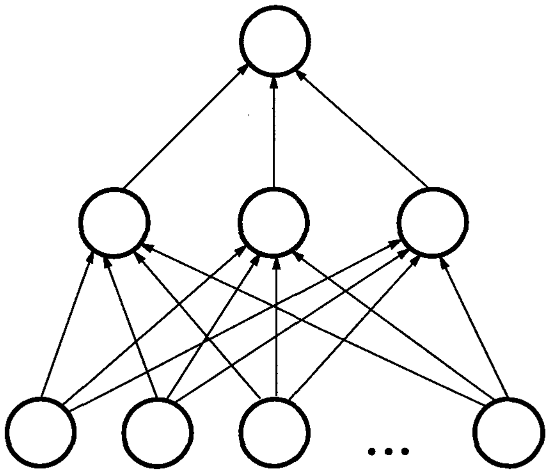
$$y_i^{(\ell+1)} = \sigma\left(\sum_j W_{ij}^{(\ell)} y_j^{(\ell)}\right), \qquad \text{where} \quad \sigma(x) = \frac{1}{1+e^{-x}}$$



is the logistic "sigmoid" function. Note: sigmoid is only one kind of non-linearity, many others are possible.

Universal approximation property with sufficiently many (∞) hidden units.

Learning MLPs by Error Backpropagation



$$E(W) = \frac{1}{2}\sum_n (t^n - y(x^n, W))^2 = \sum_n E^n$$

Idea, Chain Rule: $$\frac{\partial E^n}{\partial W_{ij}^{(\ell)}} = \frac{\partial E^n}{\partial y_i^{(\ell+1)}} \frac{\partial y_i^{(\ell+1)}}{\partial W_{ij}^{(\ell)}}$$

Computes gradients efficiently.

(Werbos 1974; Parker 1985; Rumelhart, Hinton & Williams 1986)

Error Functions and Noise Models

Why squared error?

- does not make sense for classification
- sensitive to outliers
- what if predicting only positive numbers
- what if scales of outputs differ?

seems, and is, *ad-hoc*

Noise models/generative models: $p(t|x, W)$

Maximizing log likelihood $\Leftrightarrow$ minimizing error

$$\ln p(t|x, W) = \ln \prod_n p(t^n|x^n, W) = \sum_n \ln p(t^n|x^n, W) = -E(W)$$

Noise Models

- Gaussian

$$
\begin{aligned}
p(t^n|x^n, W) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}(t^n - y(x^n, W))^2\} \\
E(W) &= -\ln p(t^n|x^n, W) = \frac{1}{2\sigma^2}(t^n - y(x^n, W))^2 + \text{const}
\end{aligned}
$$

$\Rightarrow$ Squared Error

- Exponential

$$
\begin{aligned}
p(t^n|x^n, W) &= \lambda \exp\{-\lambda |t^n - y(x^n, W)|\} \\
E(W) &= -\ln p(t^n|x^n, W) = \lambda |t^n - y(x^n, W)| - \ln \lambda
\end{aligned}
$$

$\Rightarrow$ Absolute Error

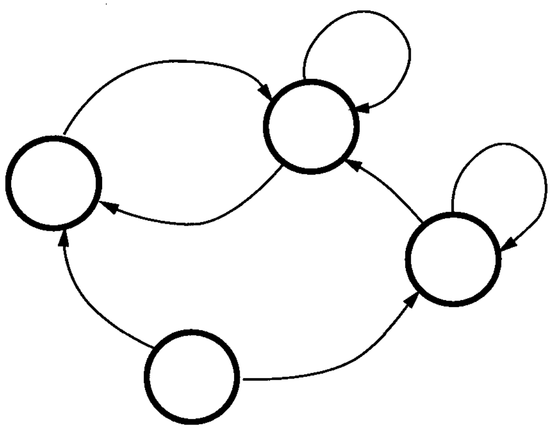
- Bernoulli

$$
\begin{aligned}
p(t^n|x^n, W) &= y(x^n, W)^{t^n} (1 - y(x^n, W))^{1-t^n} \\
E(W) &= -\ln p(t^n|x^n, W) = -t^n \ln y(x^n, W) - (1 - t^n) \ln(1 - y(x^n, W))
\end{aligned}
$$

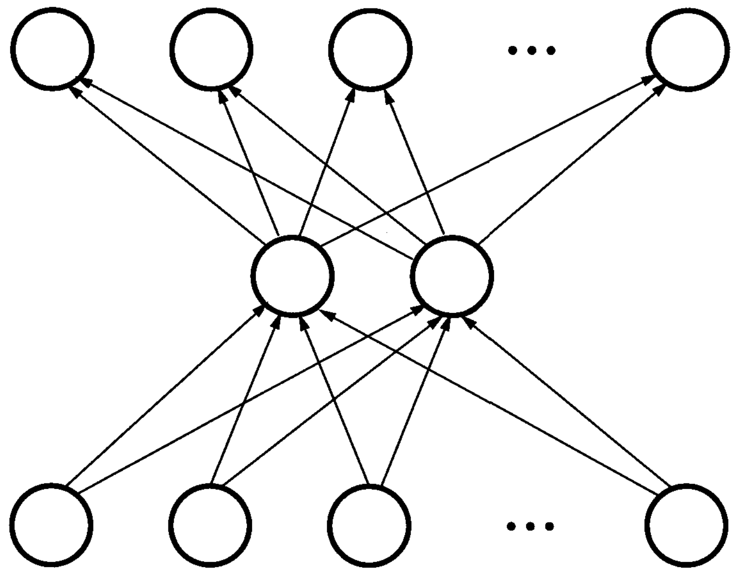
$\Rightarrow$ Cross-Entropy Error

Varieties of MLP

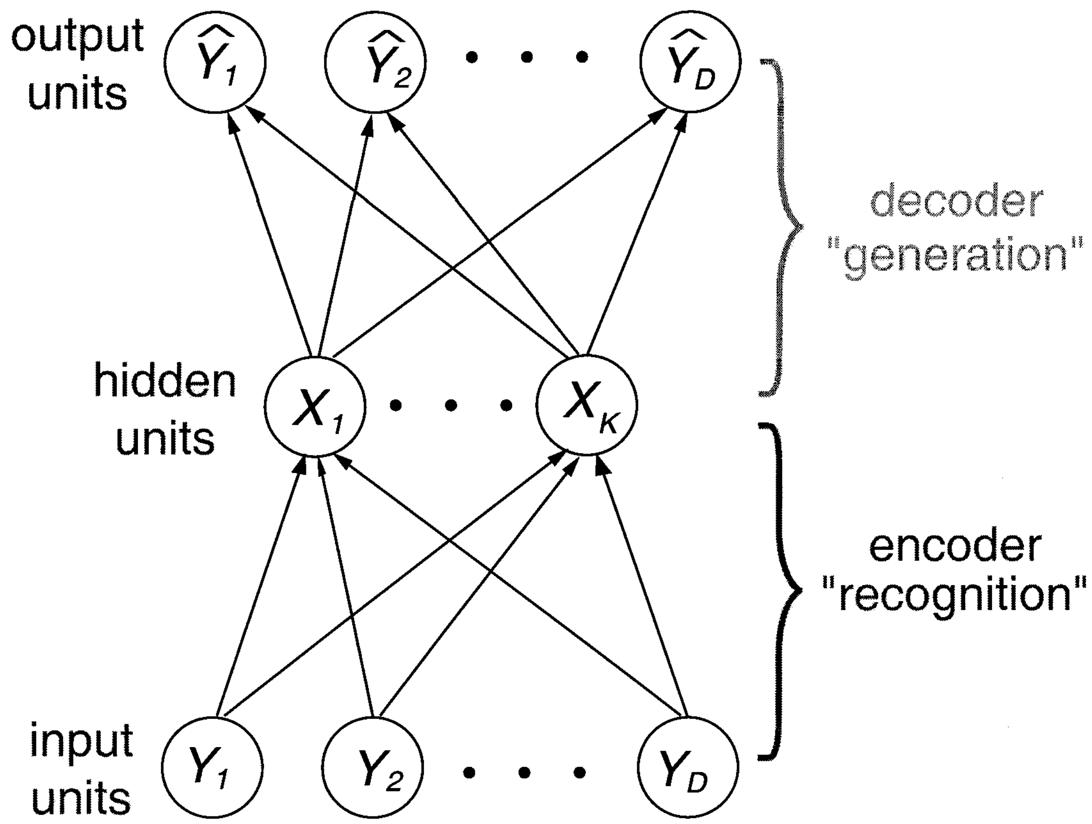
Recurrent Networks



Autoencoders



Autoencoders and Unsupervised Learning



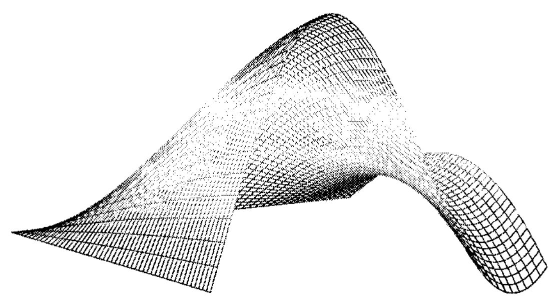
Goals of Unsupervised Learning

To find useful representations of the data, for example:

- finding clusters — e.g. k-means, MoG, ART
- dimensionality reduction — e.g. PCA, Hebbian learning, MDS, LLE, Isomap
- building topographic maps — e.g. elastic networks, Kohonen maps
- finding the hidden causes or sources of the data
- modeling the data density

Clustering

Dimensionality Reduction



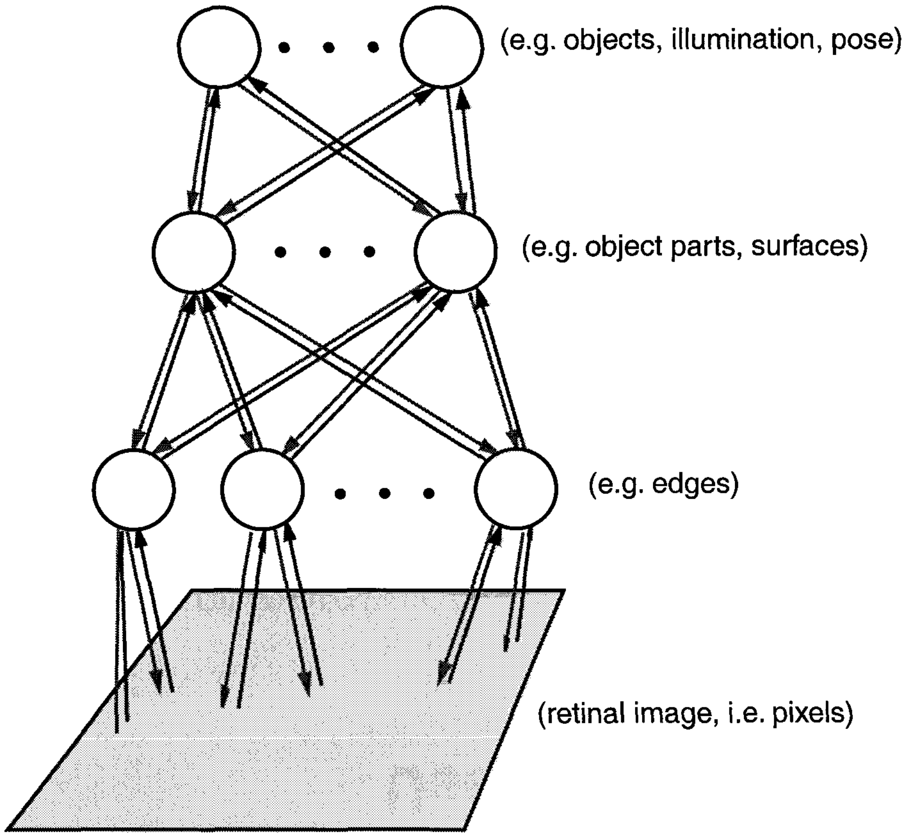
Uses of Unsupervised Learning

- data compression
- outlier detection
- classification
- make other learning tasks easier

View: The (sensory) brain is a statistical inference engine .

The brain extracts statistical regularities from data (words, objects, theories) and builds probabilistic models of the data.

Generative Models and Recognition Models



Assume we have a generative model of the sensory world.

We invert this model (using Bayes rule) for perception/recognition/inference.

$$P(\mathsf{E}|\mathsf{D}) = \frac{P(\mathsf{D}|\mathsf{E})P(\mathsf{E})}{P(\mathsf{D})}$$

D="sensory data"

E="explanation" = hypothesis about what's out there

A possible role for feedback connections in cortex?

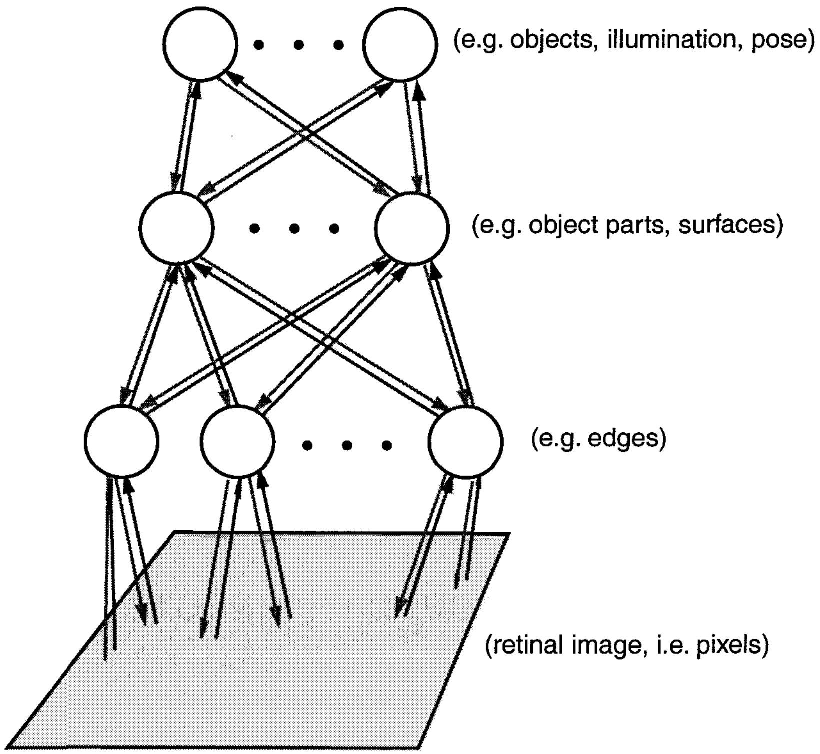
Probabilistic Models

- A probabilistic model of sensory inputs can be used to:
  - make optimal decisions under a given loss function
  - make inferences about missing inputs
  - generate predictions/fantasies/imagery
  - communicate the data in an efficient way

- Probabilistic modeling is equivalent to other views of learning:
  - information theoretic:
    finding compact representations of the data

  - physical analogies: minimising free energy of a corresponding
    statistical mechanical system

The EM (Expectation–Maximization) Algorithm



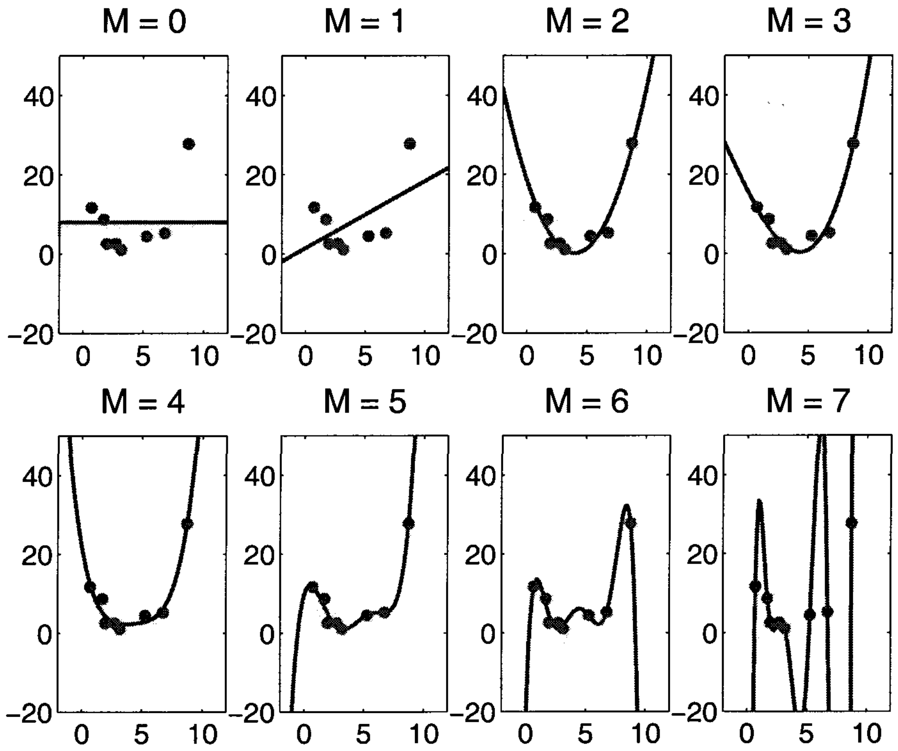
How to learn a generative model of the sensory world...

Start from some model with hidden causes/explanations, E.

- Do recognition to infer the hidden causes given the observed data $P(\mathrm{E}|\mathrm{D})$. (E-step)
- Assume the inferred causes are true and refine your model, i.e. by changing connection strengths. (M-step)
- Repeat

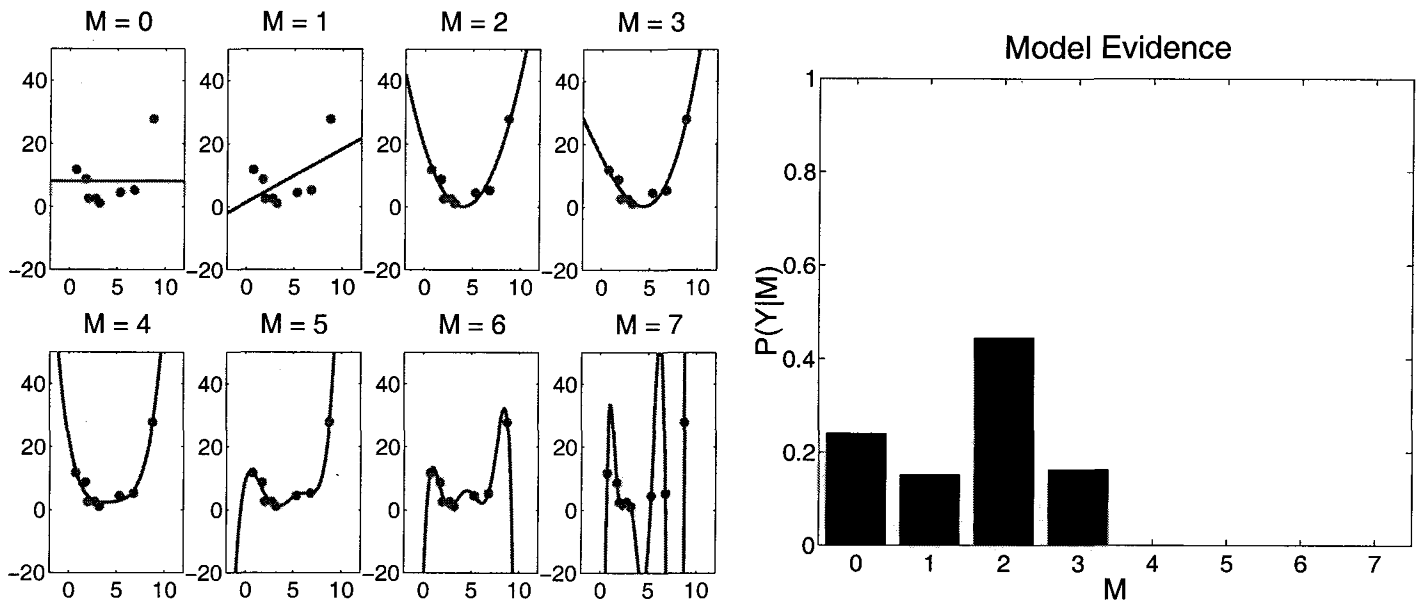
Proven to converge to a local maximum of the likelihood.

Overfitting



- weight decay, regularizers: $\tilde{E}(W) = E(W) + \frac{1}{2}\sum_{ij} W_{ij}^2$
- early stopping
- averaging over many random runs (bagging)
- cross-validation
- Bayesian methods

Bayes Rule and Model Selection



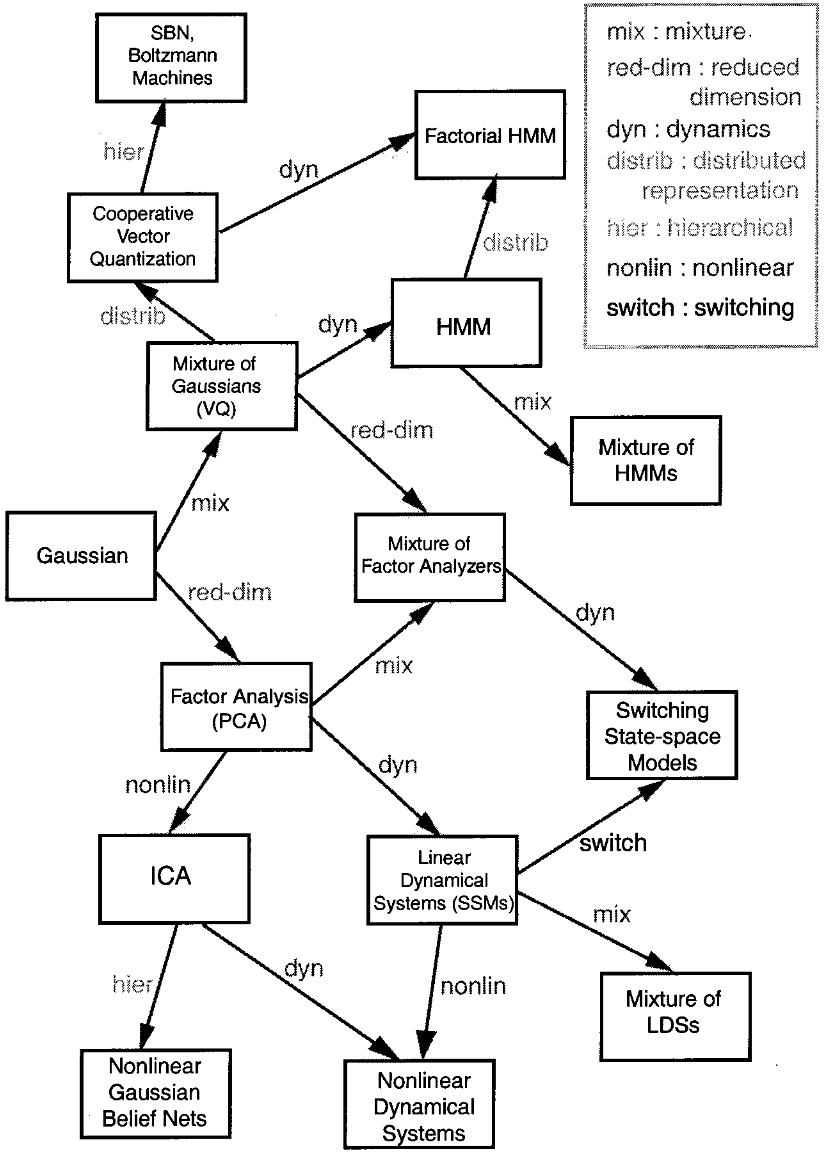
D - data $\qquad M_1, \ldots, M_n$ - models $\qquad \theta_1, \ldots, \theta_n$ - parameter vectors

$$P(M_i|D) = \frac{P(D|M_i)P(M_i)}{P(D)}$$

$$P(D|M_i) = \int p(D|\theta_i, M_i)p(\theta_i|M_i)d\theta_i$$

Note: we don't try to find a single parameter setting, we average over *all possible* parameter settings.

A Generative Model for Generative Models



Summary of Key Ideas

- The brain solves computational problems
- Supervised, unsupervised and reinforcement learning
- Hopfield networks
- Perceptrons, multi-layer perceptrons, and backpropagation
- Error functions and noise models
- Autoencoders and unsupervised learning
- Clustering and dimensionality reduction
- [Overfitting and Bayes Rule]

Selected References

- **General:** *Neural Networks for Pattern Recognition* (1995) C. Bishop. Oxford University Press.
- *Learning in Graphical Models* (1998) Edited by M.I. Jordan. Dordrecht: Kluwer Academic Press. Also available from MIT Press (paperback).
- Roweis, S.T and Ghahramani, Z. (1999) A unifying review of linear Gaussian models. *Neural Computation* **11**(2): 305–345.
- **Motivation for Bayes Rule**: "Probability Theory - the Logic of Science," E.T.Jaynes, http://www.math.albany.edu:8008/JaynesBook.html
- **EM:**

  Dempster, A., Laird, N., and Rubin, D. (1977).
  Maximum likelihood from incomplete data via the EM algorithm.
  J. Royal Statistical Society Series B, 39:1–38;

  Neal, R. M. and Hinton, G. E. (1998).
  A new view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models.*
- **Recognition Models:**

  Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161.
- **Bayesian Ockham's Razor:**

  Jefferys, W.H., Berger, J.O. (1992)
  Ockham's Razor and Bayesian Analysis. *American Scientist* **80**:64-72;