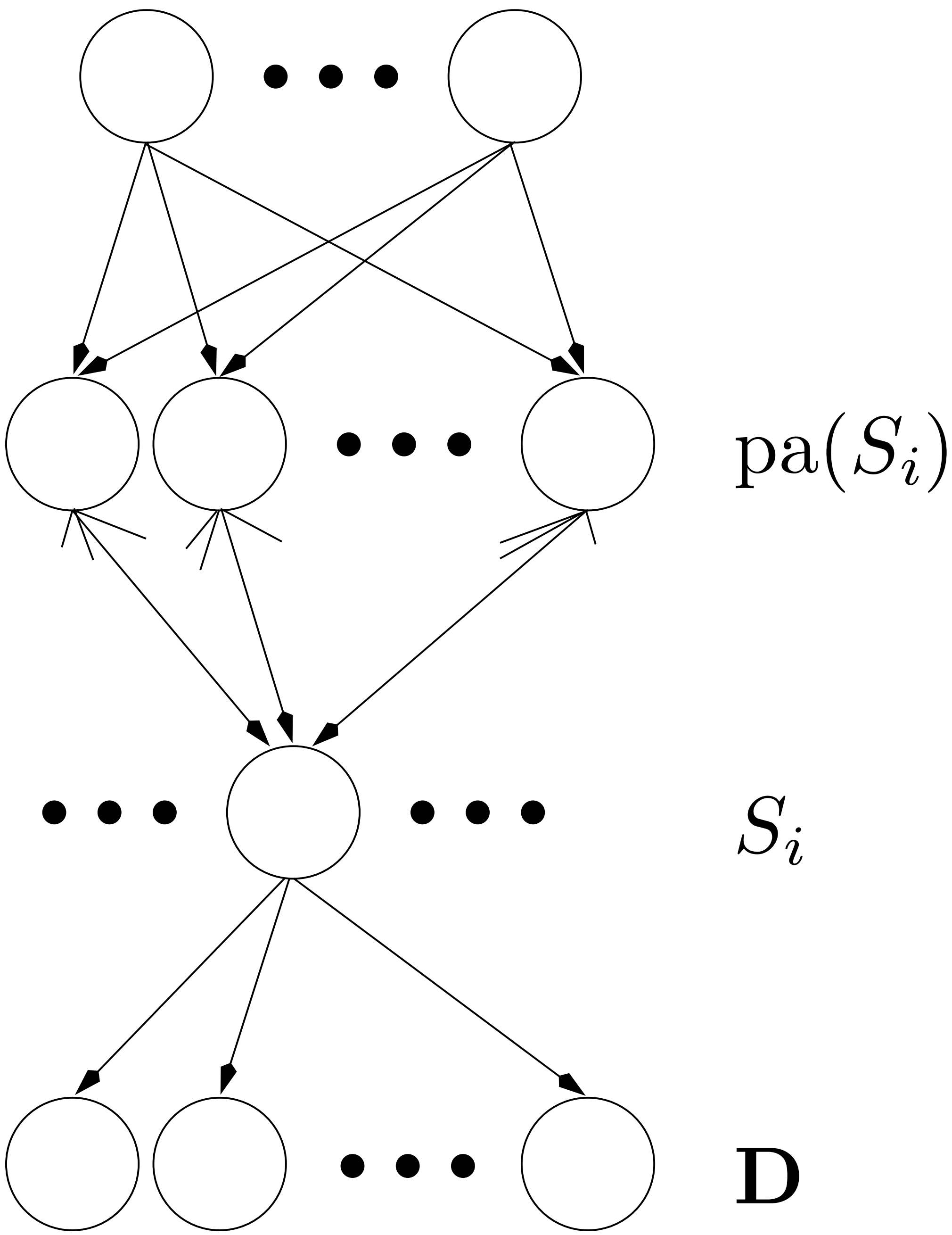
# EECS 391
# Intro to AI

## Deep Belief Networks

L16 Thu Nov 2

# Hierarchical Statistical Models

A Bayesian belief network:



The joint probability of binary states is

$$P(\mathbf{S}|\mathbf{W}) = \prod_i P(S_i|\mathrm{pa}(S_i), \mathbf{W})$$

The probability $S_i$ depends only on its parents:

$$P(S_i|\mathrm{pa}(S_i), \mathbf{W}) = \begin{cases} h(\sum_j S_j w_{ji}) & \text{if } S_i = 1 \\ 1 - h(\sum_j S_j w_{ji}) & \text{if } S_i = 0 \end{cases}$$

The function $h$ specifies how causes are combined, $h(u) = 1 - \exp(-u)$, $u > 0$.

Main points:

- hierarchical structure allows model to form high order representations
- upper states are priors for lower states
- weights encode higher order features

# Inferring the best representation of the observed variables

- Given on the input **D**, the is no simple way to determine which states are the input’s most likely causes.
  - Computing the most probable network state is an *inference* process
  - we want to find the explanation of the data with highest probability
  - this can be done efficiently with *Gibbs sampling*
- Gibbs sampling is another example of an MCMC method
- Key idea:

*The samples are guaranteed to converge to the*

*true posterior probability distribution*

# Gibbs Sampling

Gibbs sampling is a way to select an ensemble of states that are representative of the posterior distribution $P(\mathbf{S}|\mathbf{D}, \mathbf{W})$.

- Each state of the network is updated iteratively according to the probability of $S_i$ given the remaining states.

- this conditional probability can be computed using (Neal, 1992)

$$P(S_i = a|S_j : j \neq i, \mathbf{W}) \propto P(S_i = a|\mathrm{pa}(S_i), \mathbf{W}) \prod_{j \in \mathrm{ch}(S_i)} P(S_j|\mathrm{pa}(S_j), S_i = a, \mathbf{W})$$

- limiting ensemble of states will be typical samples from $P(\mathbf{S}|\mathbf{D}, \mathbf{W})$

- also works if any subset of states are fixed and the rest are sampled

# The Gibbs sampling equations (derivation omitted)

The probability of $S_i$ changing state given the remaining states is

$$P(S_i = 1 - S_i | S_j : j \neq i, \mathbf{W}) = \frac{1}{1 + \exp(-\Delta x_i)}$$

$\Delta x_i$ indicates how much changing the state $S_i$ changes the probability of the whole network state

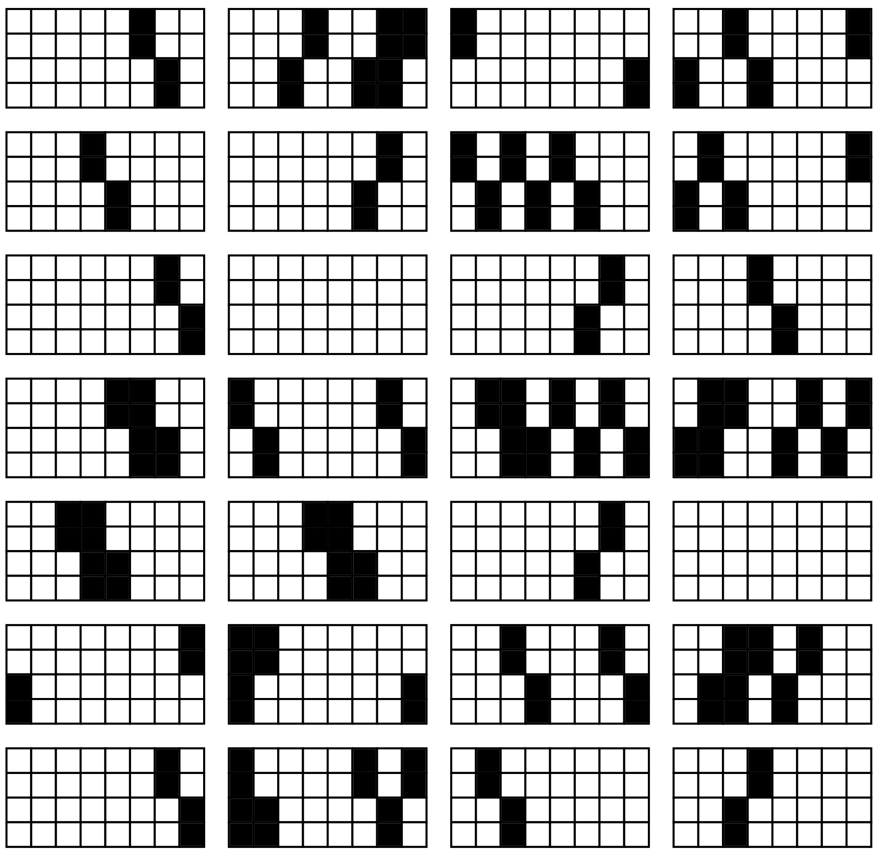
$$\begin{aligned}\Delta x_i = {} & \log h(u_i; 1 - S_i) - \log h(u_i; S_i) \\ & + \sum_{j \in \mathrm{ch}(S_i)} \log h(u_j + \delta_{ij}; S_j) - \log h(u_j; S_j)\end{aligned}$$

- $u_i$ is the causal input to $S_i$, $u_i = \sum_k S_k w_{ki}$
- $\delta_j$ specifies the change in $u_j$ for a change in $S_i$,
  $\delta_{ij} = +S_j w_{ij}$ if $S_i = 0$, or $-S_j w_{ij}$ if $S_i = 1$

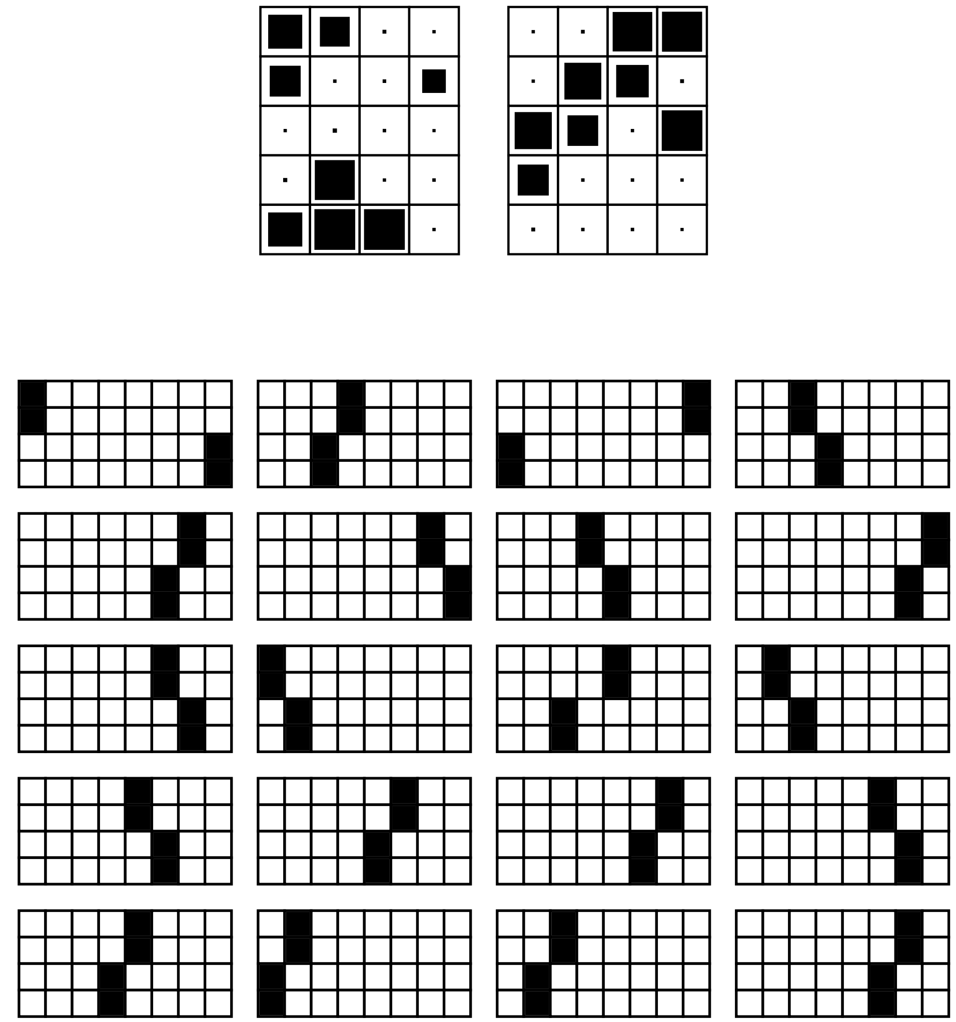
# Interpretation of the Gibbs sampling equation

- The Gibbs equation can be interpreted as: *feedback* + ∑ *feedforward*
- *feed-back:* how consistent is Si with current causes?
- ∑ *feedforward:* how likely is Si a cause of its children
- feedback allows the lower-level units to use information only computable at higher levels
- feedback determines (disambiguates) the state when the feedforward input is ambiguous

## The Shifter Problem

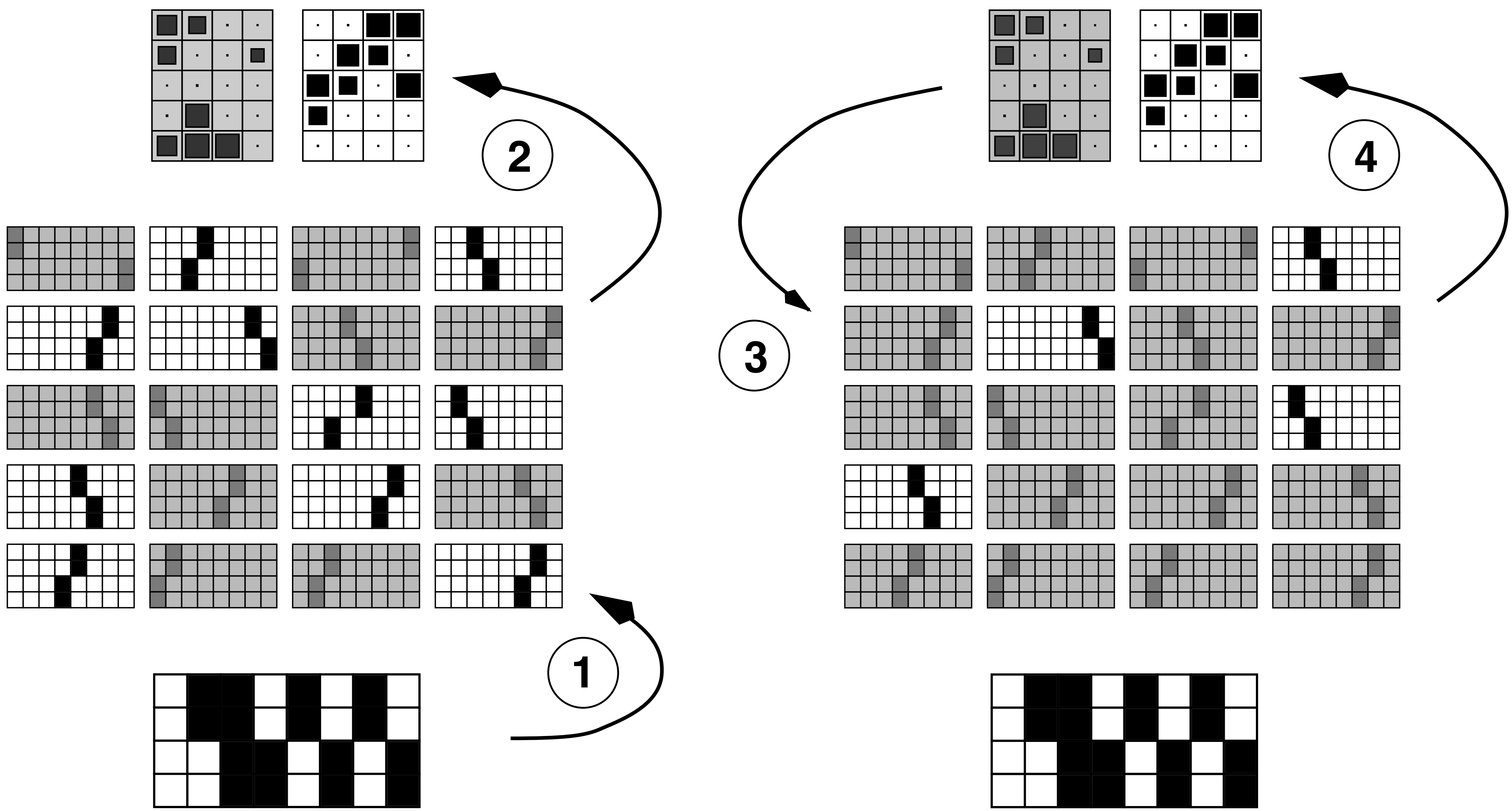


Shift patterns



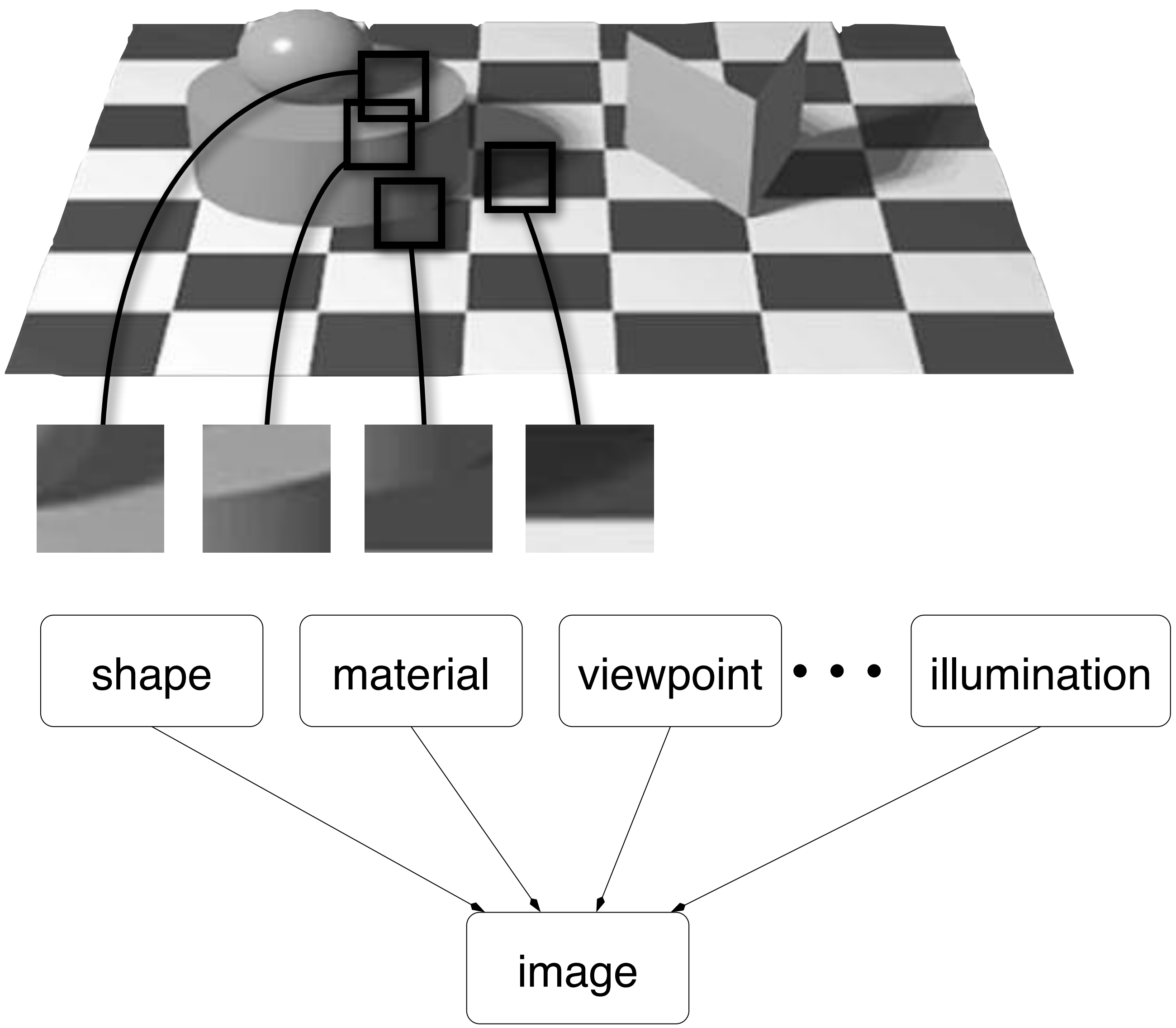
weights of a 32-20-2 network after learning

# Gibbs sampling: feedback disambiguates lower-level states




One the structure learned, the Gibbs updating convergences in two sweeps.

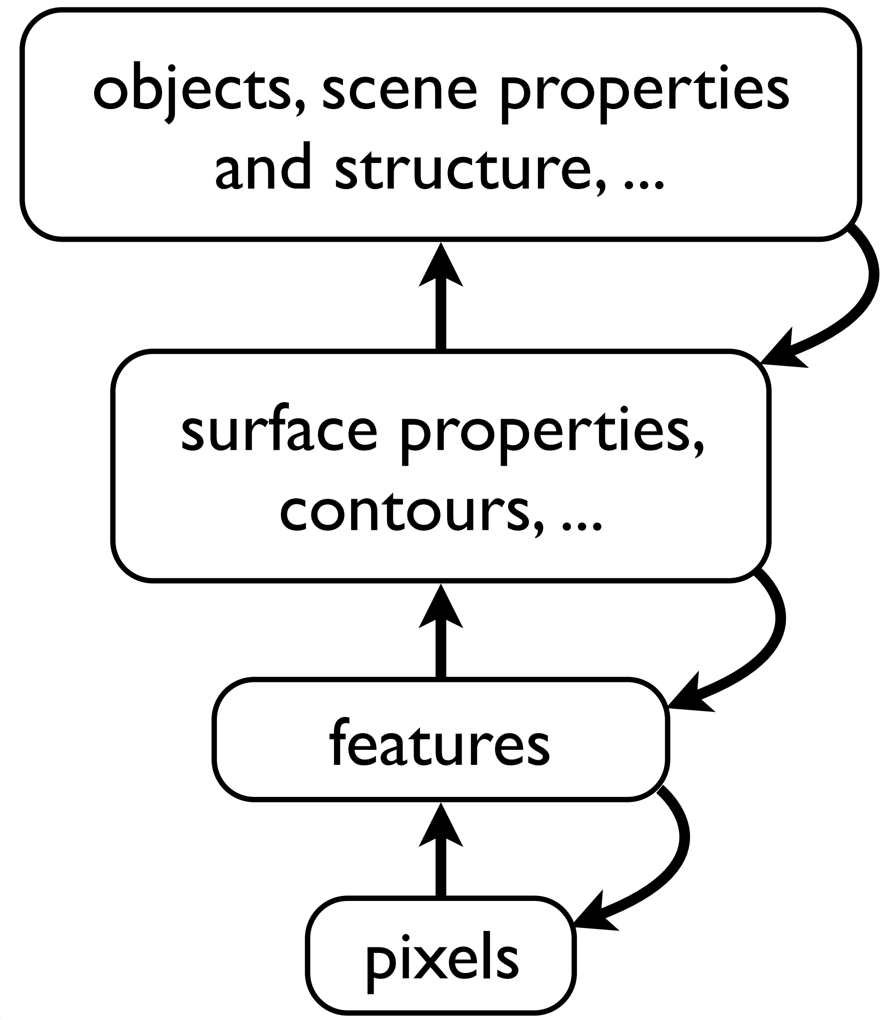
# Difficulties with identifying the causal structure

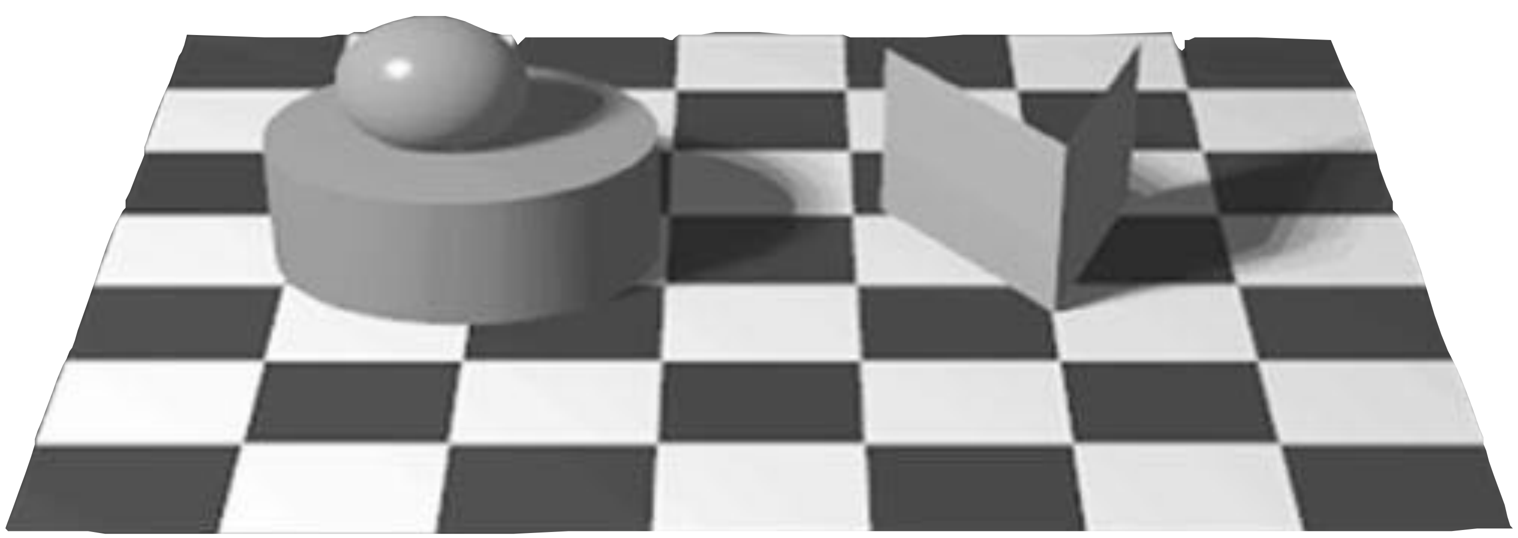



- Space of $S$, $C$ is huge
- If we define $P(I|S,C)$, must be able to invert it or search it
- Need efficient algorithms
- Many unknowns: identity of objects, types of scene elements, illuminations
- Might never have encountered some structures
- Is it even the right approach?
- Can we solve a simple case?

# Motivation: learn hierarchical, context dependent representations

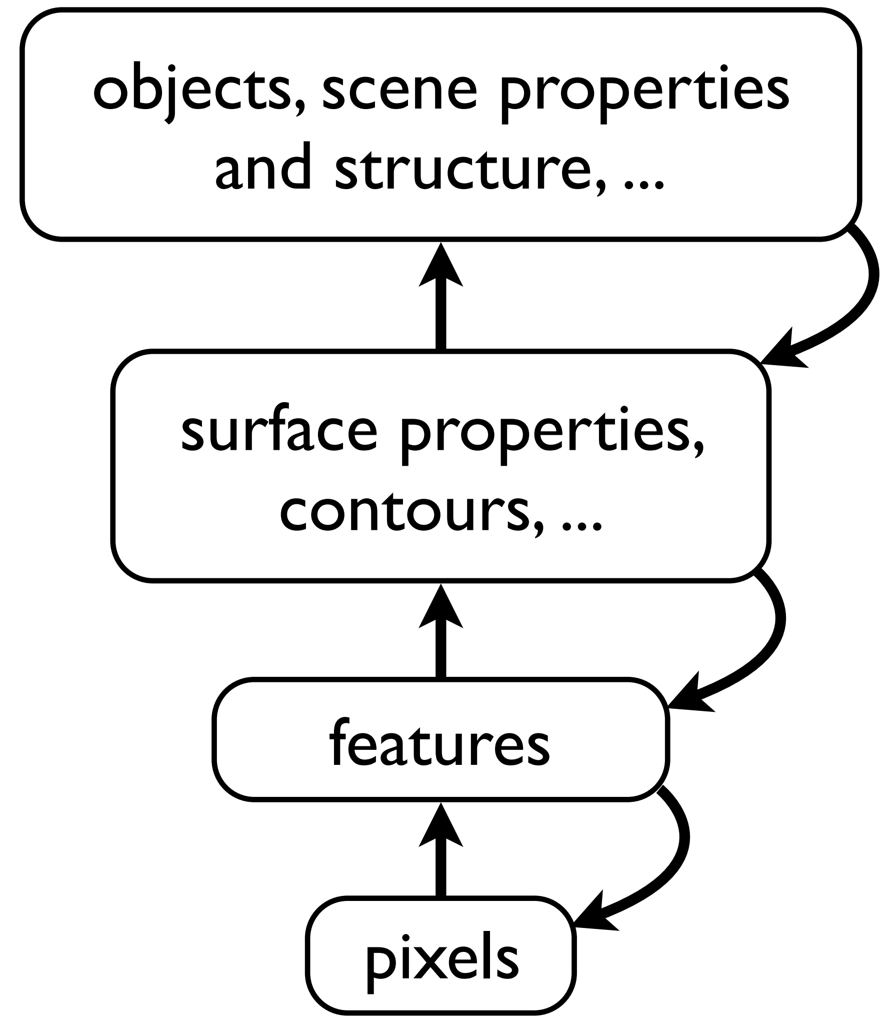
- many real-world patterns are hierarchical in structure
- interpretation of patterns depends on context
- essential for complex recognition tasks



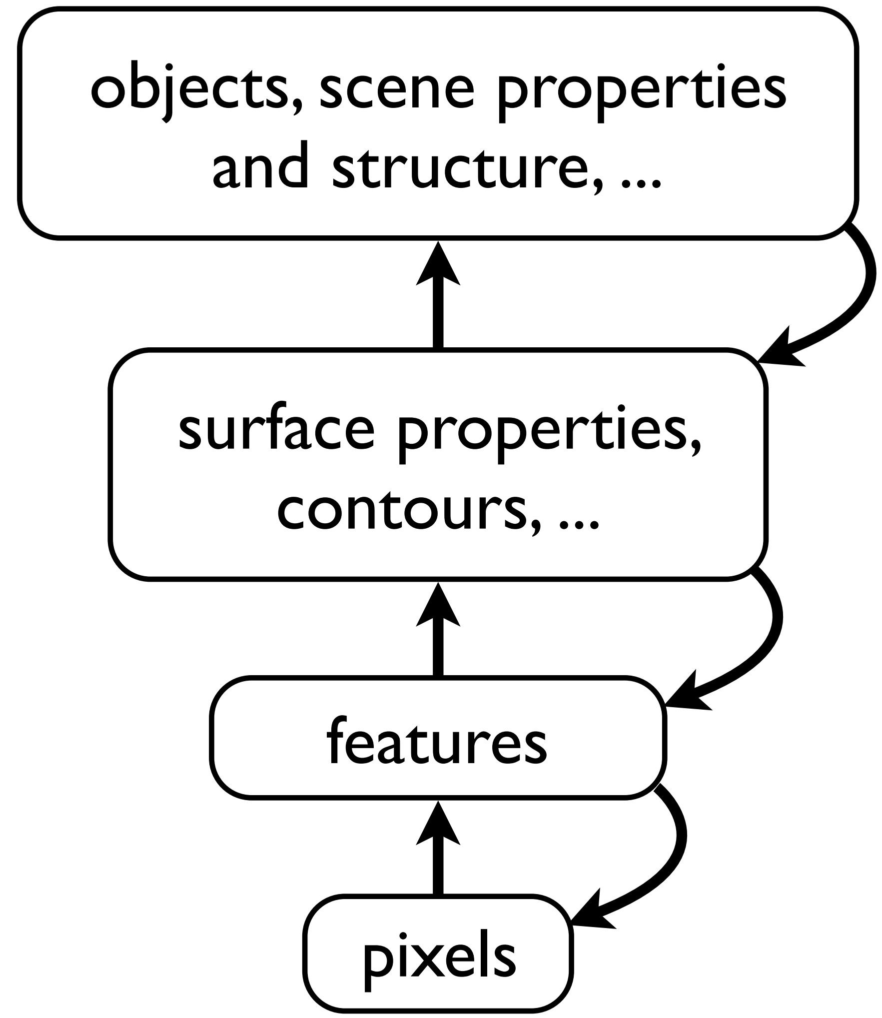
# Motivation: hierarchical, context dependent representations

- many real-world patterns are hierarchical in structure
- interpretation of patterns depends on context
- essential for complex recognition tasks



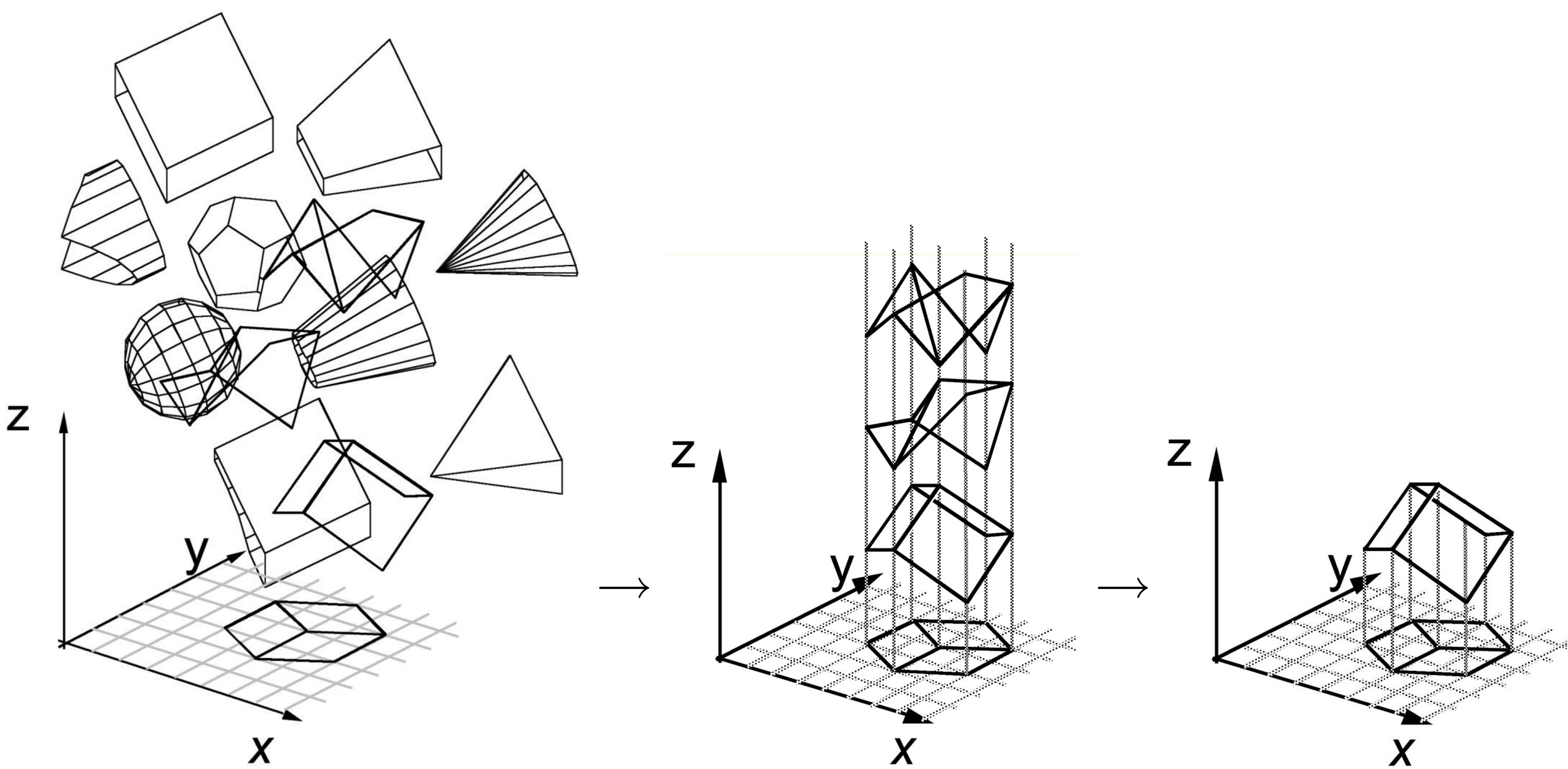
# Motivation: hierarchical, context dependent representations

- many real-world patterns are hierarchical in structure
- interpretation of patterns depends on context
- essential for complex recognition tasks



# The inferred explanation is the most probable scene

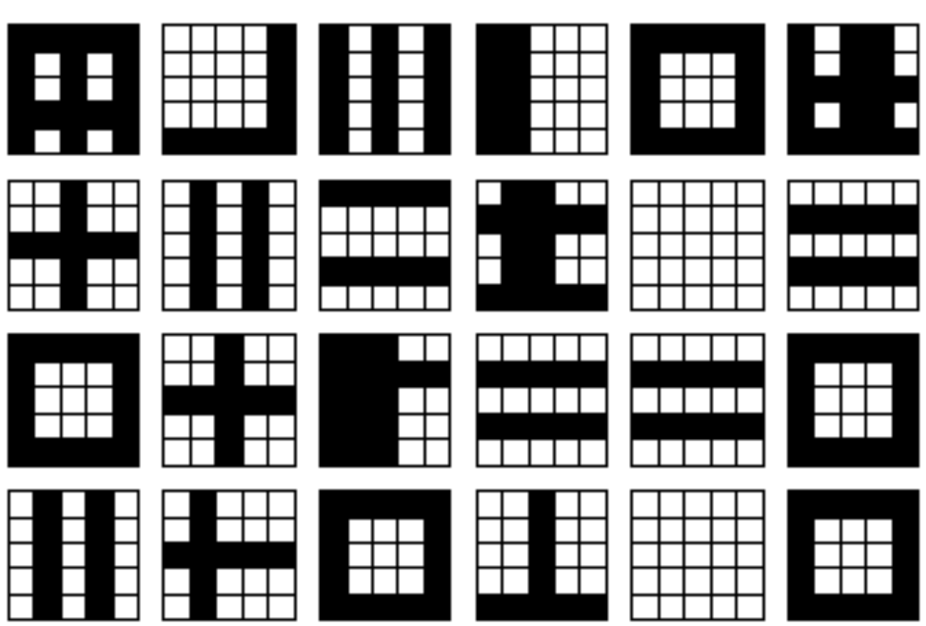
$$p(\hat{S}|I,C) = \arg\max_S \frac{p(I|S,C)p(S|C)}{p(I|C)}$$

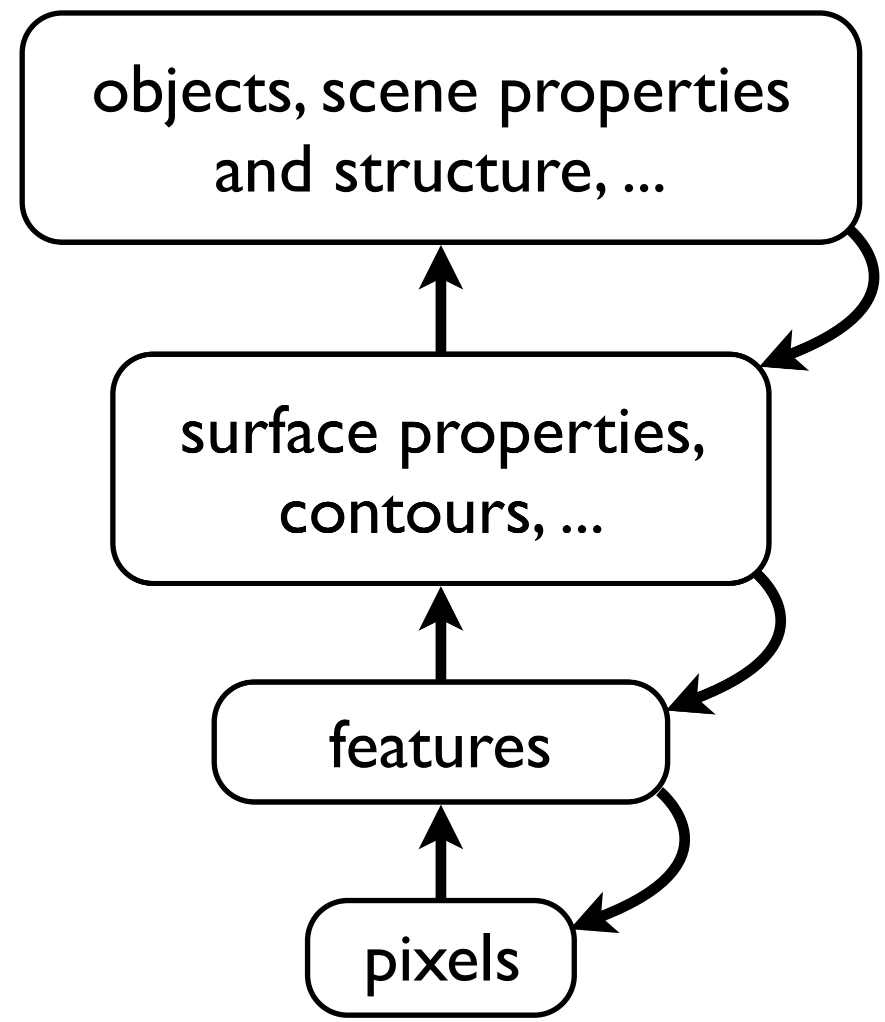


# Motivation: learn hierarchical, context dependent representations

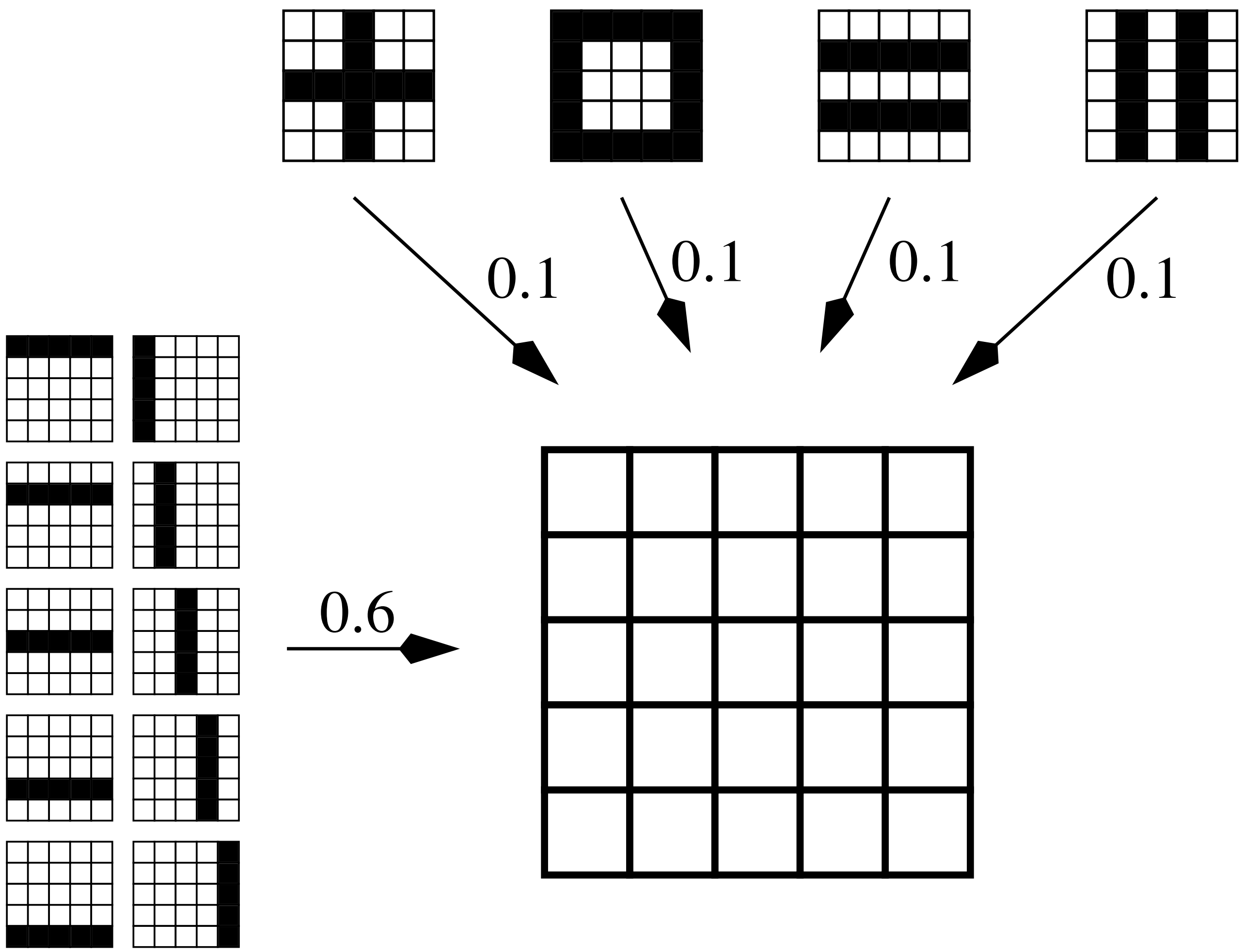
- many real-world patterns are hierarchical in structure
- interpretation of patterns depends on context
- essential for complex recognition tasks

The toy problem:
*is it a pattern or a collection of features?*

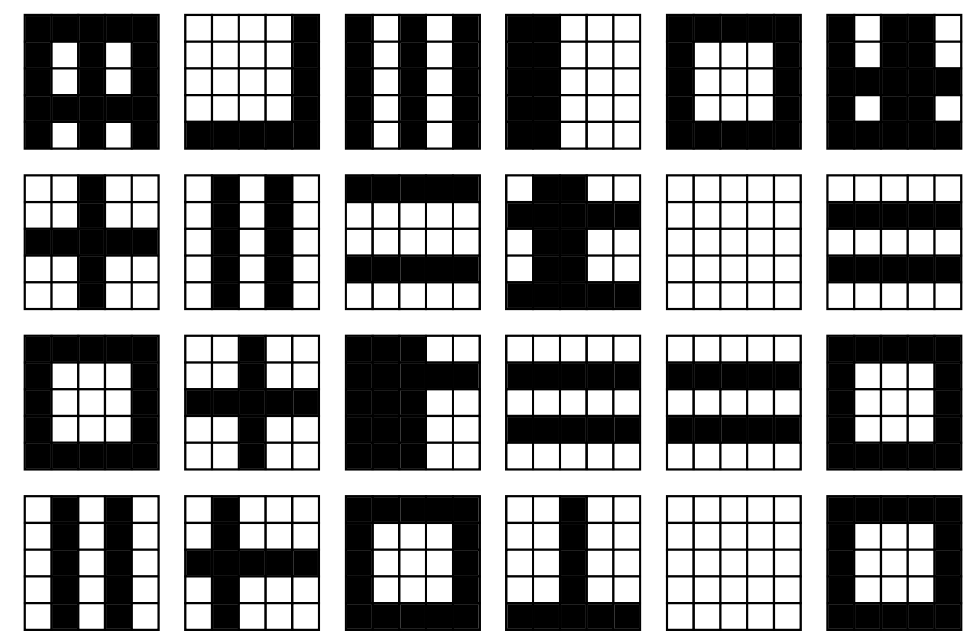




# The higher-order lines problem



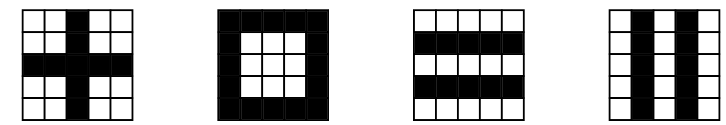

The true generative model



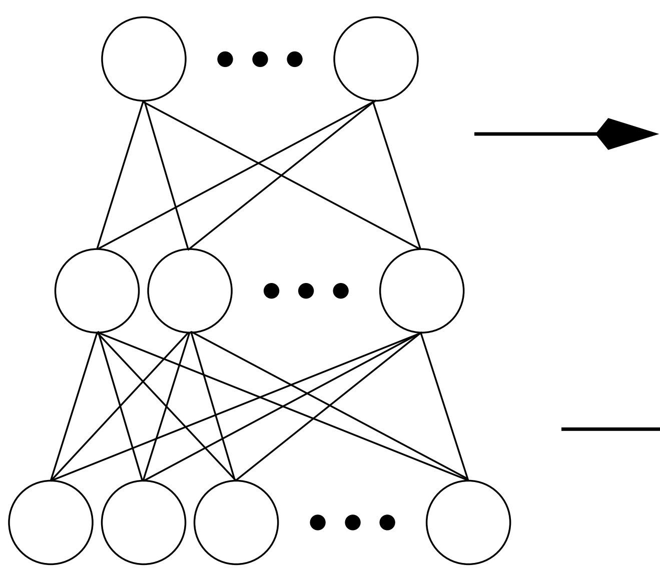
Patterns sampled from the model

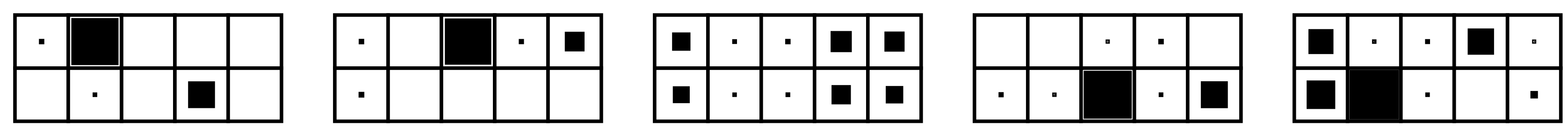
*Can we infer the structure of the network given only the patterns?*

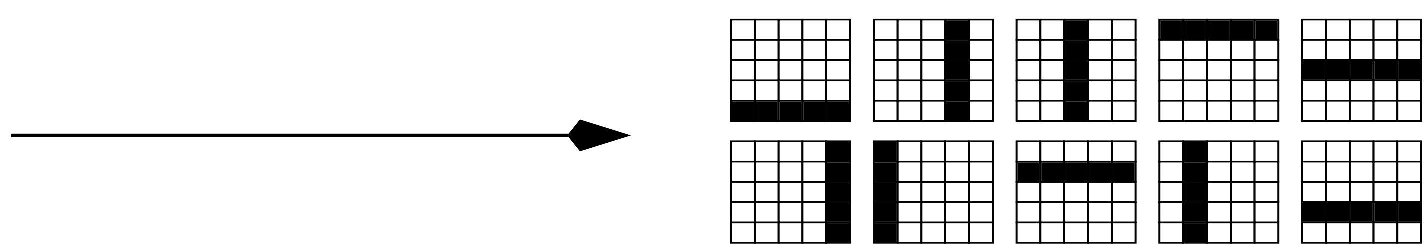
# Weights in a 25-10-5 belief network after learning



The second layer learns combinations of the first layer features

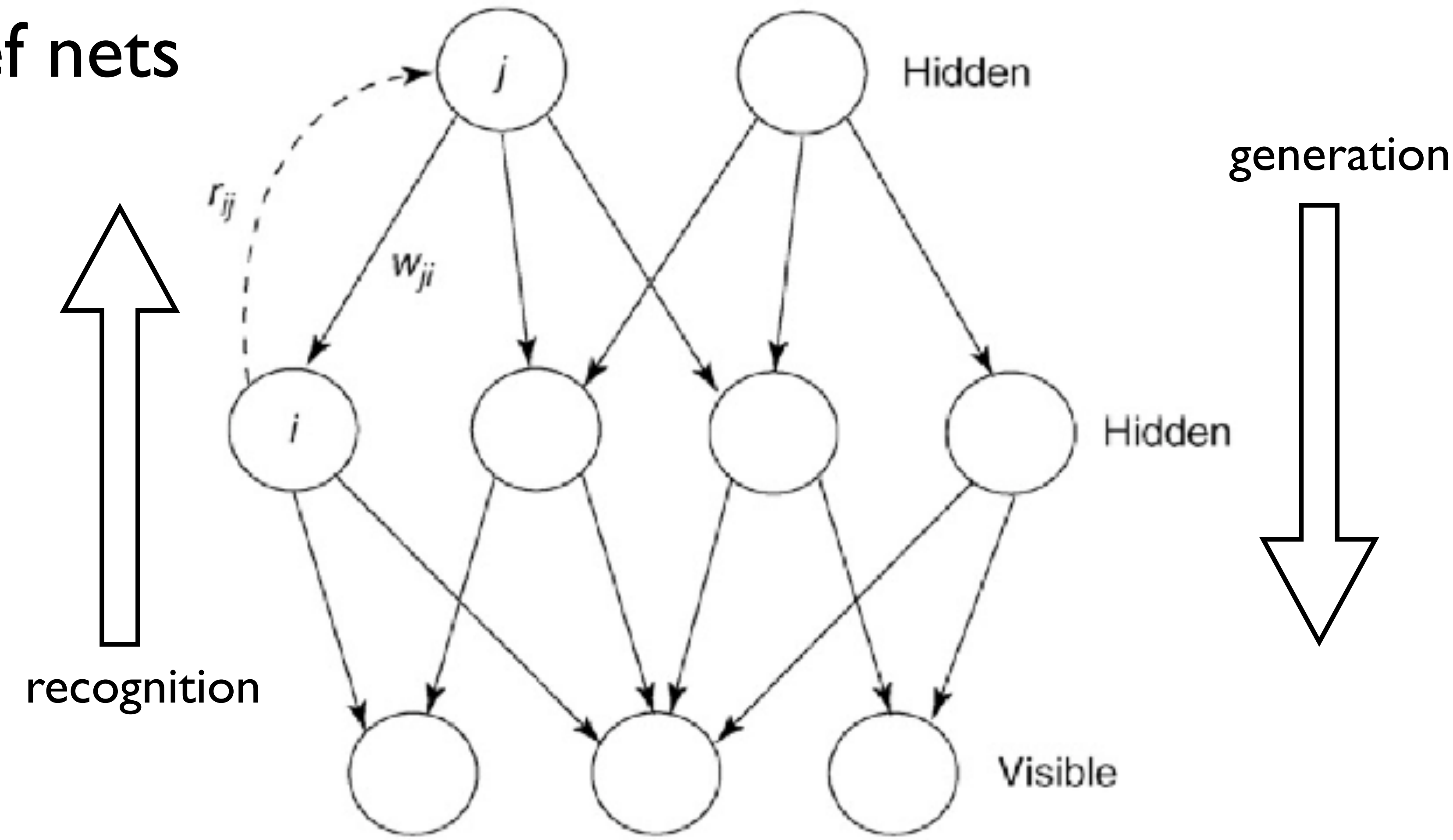






The first layer of weights learn that patterns are combinations of lines.

# Logistic belief nets




- generative model:

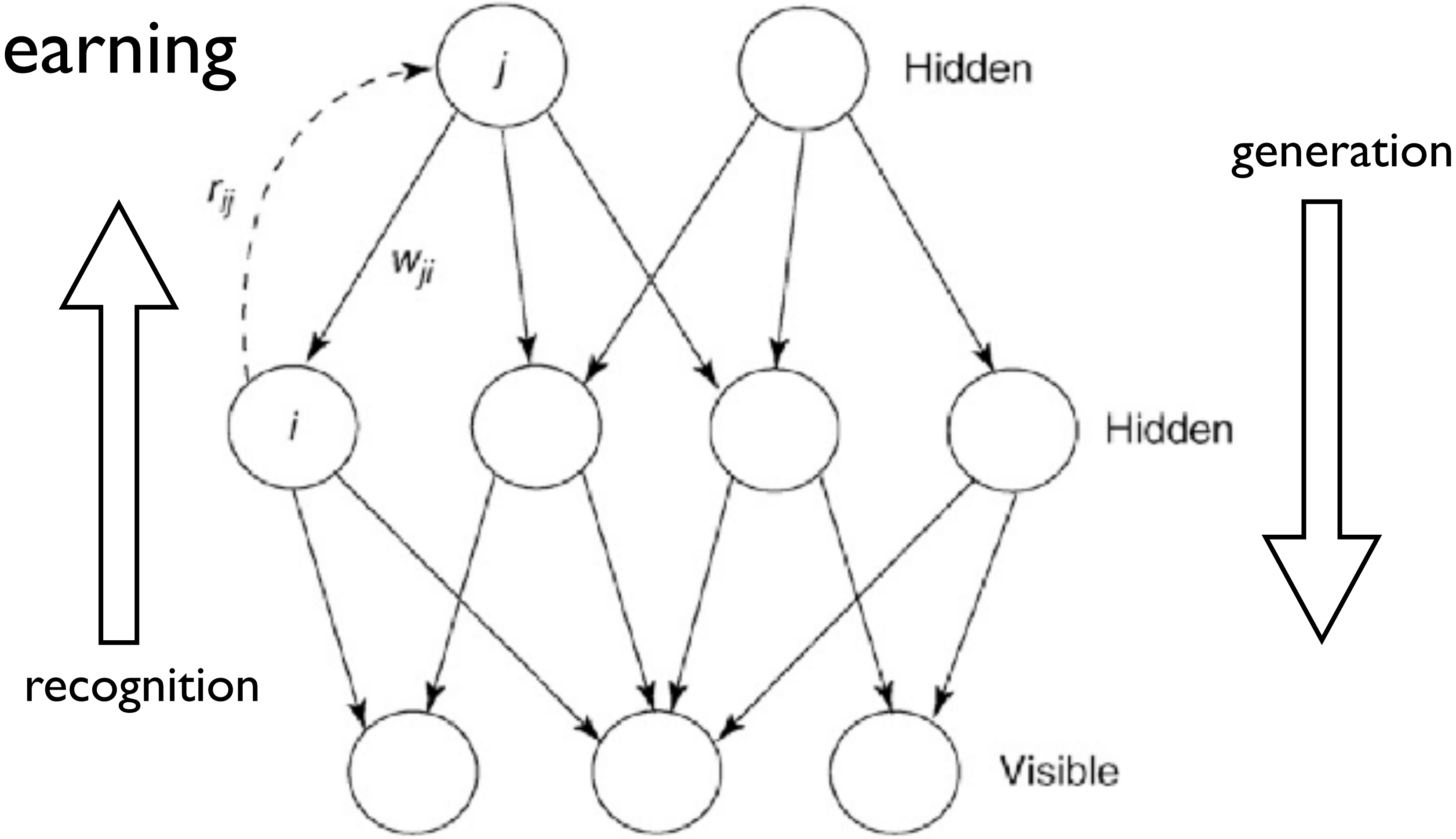
$$p(v_i = 1) = \sigma(b_i + \sum_j h_j w_{ij})$$

- $\sigma(x)$ is the logistic function:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

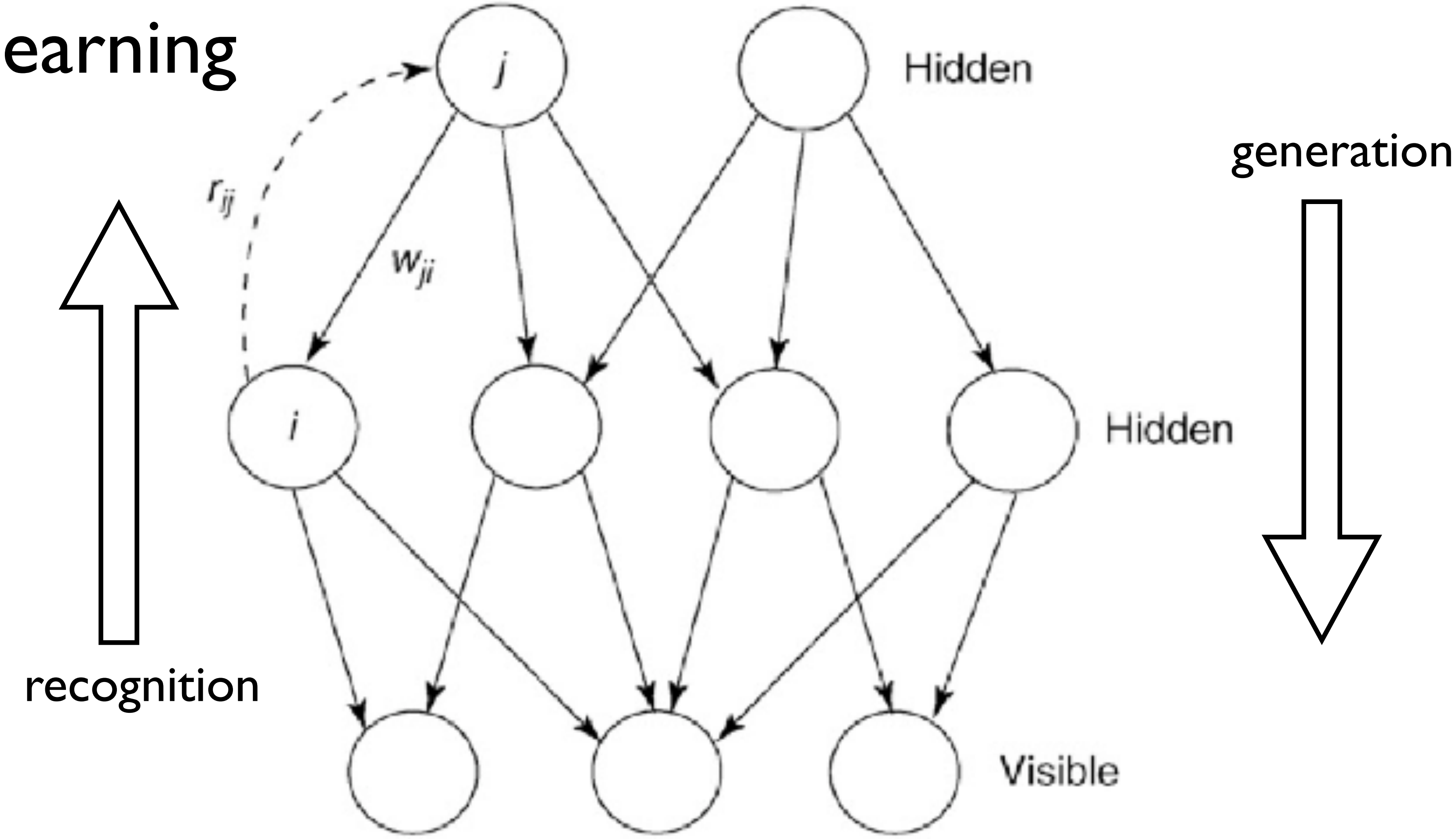
- with deep networks, it is possible to learn complex joint probability distributions

# Wake-sleep learning




- For probabilistic models
  - top-down weights generate patterns from model distribution
  - bottom-up weights convey distribution of data-vectors
  - ideally the two distributions should match
- The "wake-sleep" algorithm adjusts the weights so that the distribution from recognition (wake) matches the distribution from generation (sleep)

# Wake-sleep learning




- For each digit in training set
  - bottom-up pass: use recognition weights to stochastically set hidden states

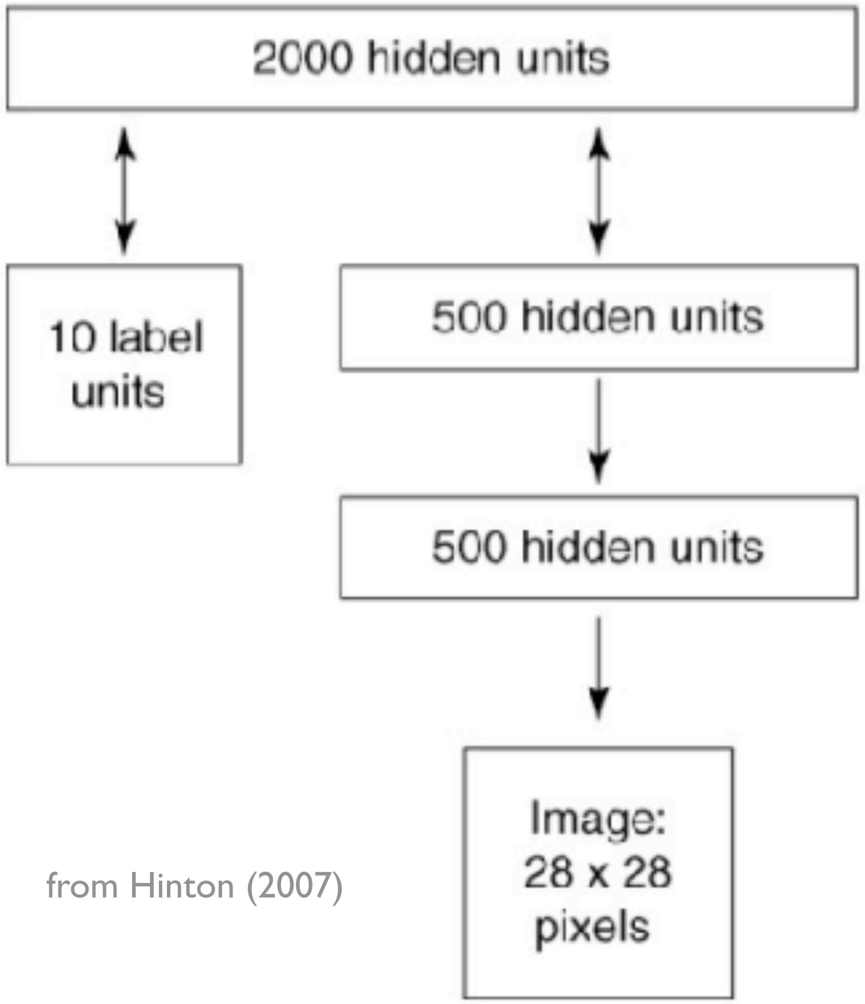
    $$p(h_j = 1) = \sigma(b_j + \sum_i v_i w_{ij})$$

  - adjust generative weights to improve how model generates training data:

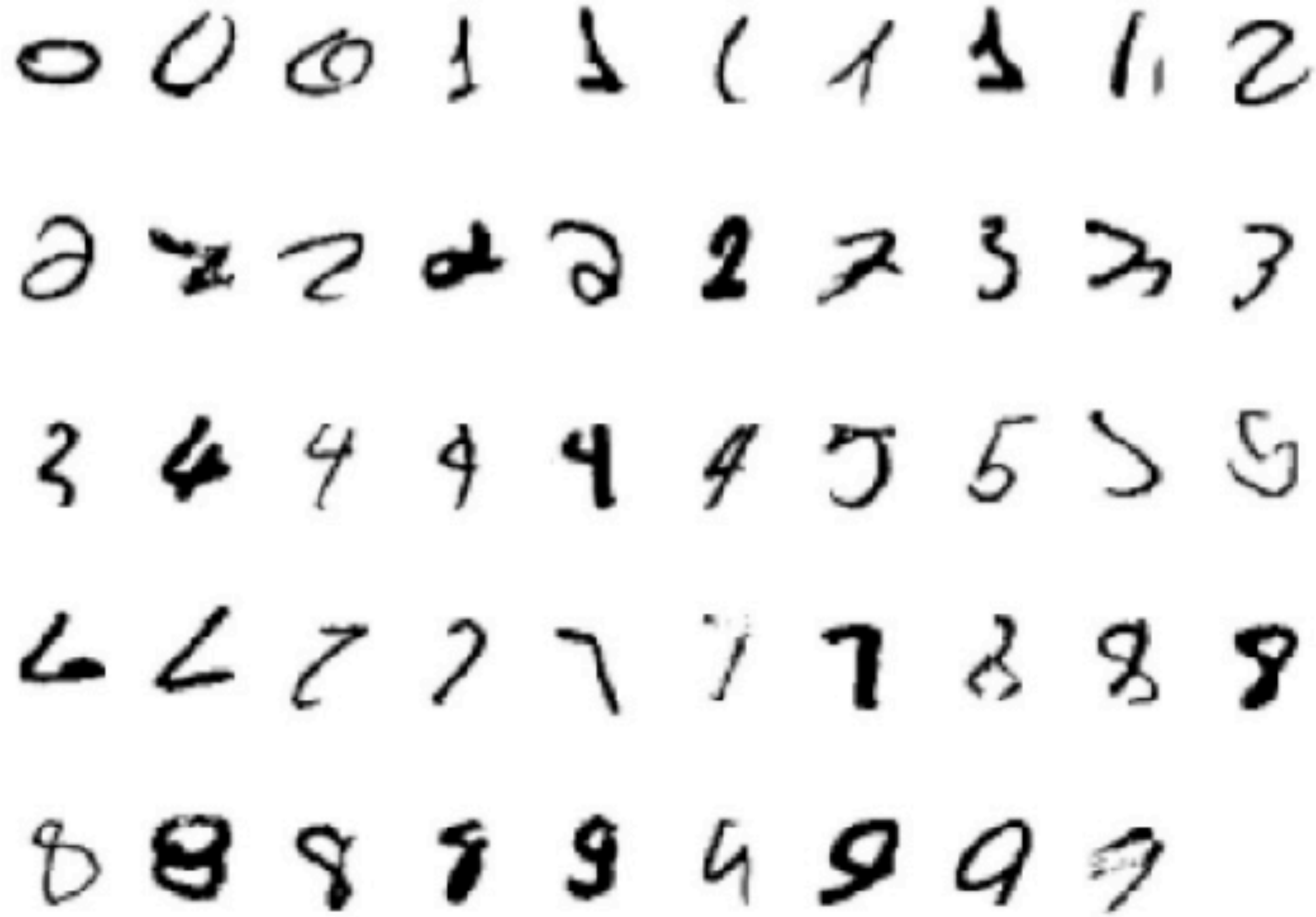
    $$\Delta w_{ji} \propto h_j (h_i - \hat{h}_i)$$

  - $\hat{h}_i$ is the probability of activating state i given inferred states $h_j$

# Generative model for hand-written digits

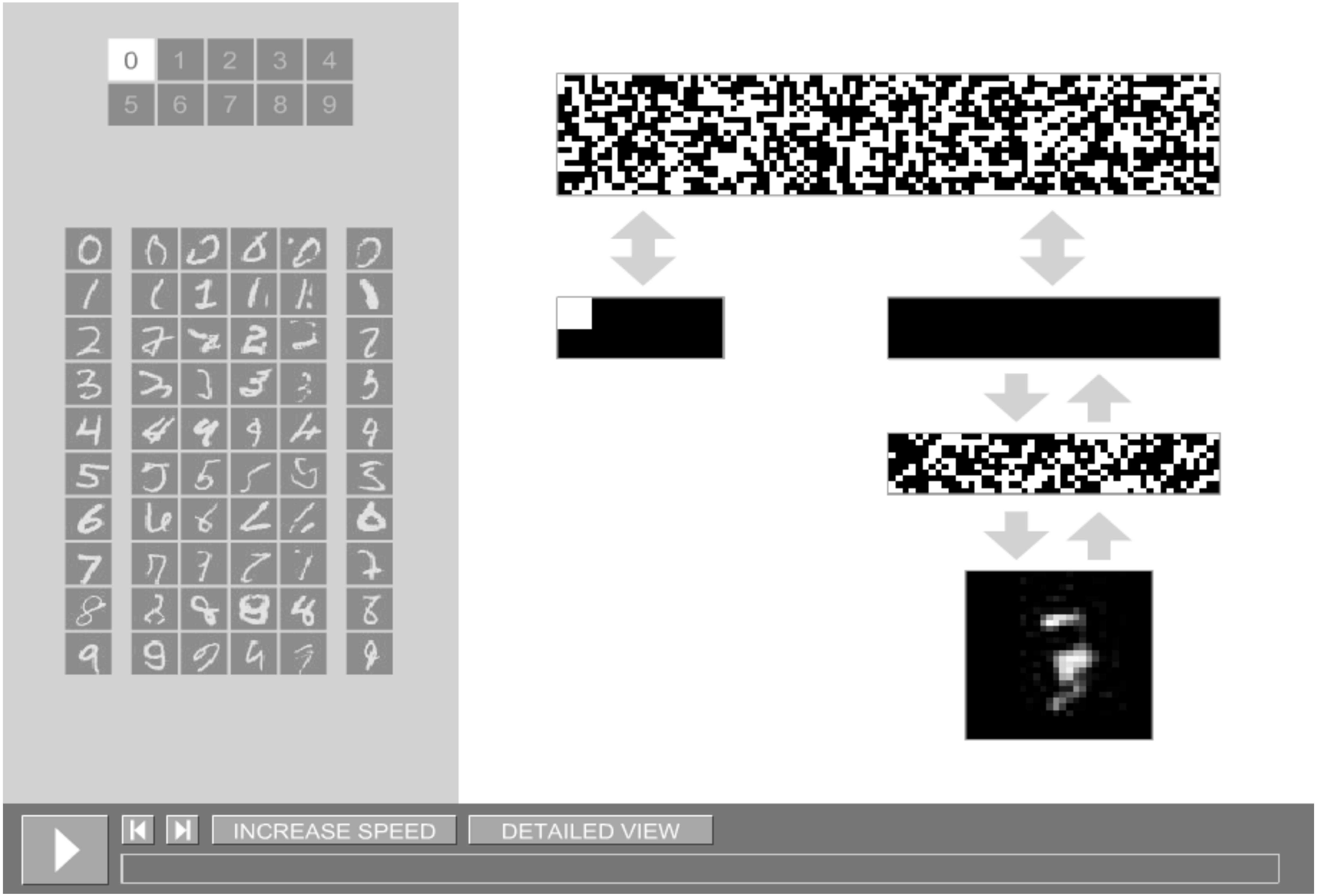



from Hinton (2007)



- Generation:
  - use alternating Gibbs sampling from top-level assoc. memory
  - use directed weights to stochastically generate pixel probs. from sampled binary of 500 hidden units
- Recognition:
  - Use bottom-up weights to produce binary activities in two lower layers
  - use alternating Gibbs sampling in the top two layers

# Demo: deep-belief network model (Hinton)




www.cs.toronto.edu/~hinton/adi/index.htm