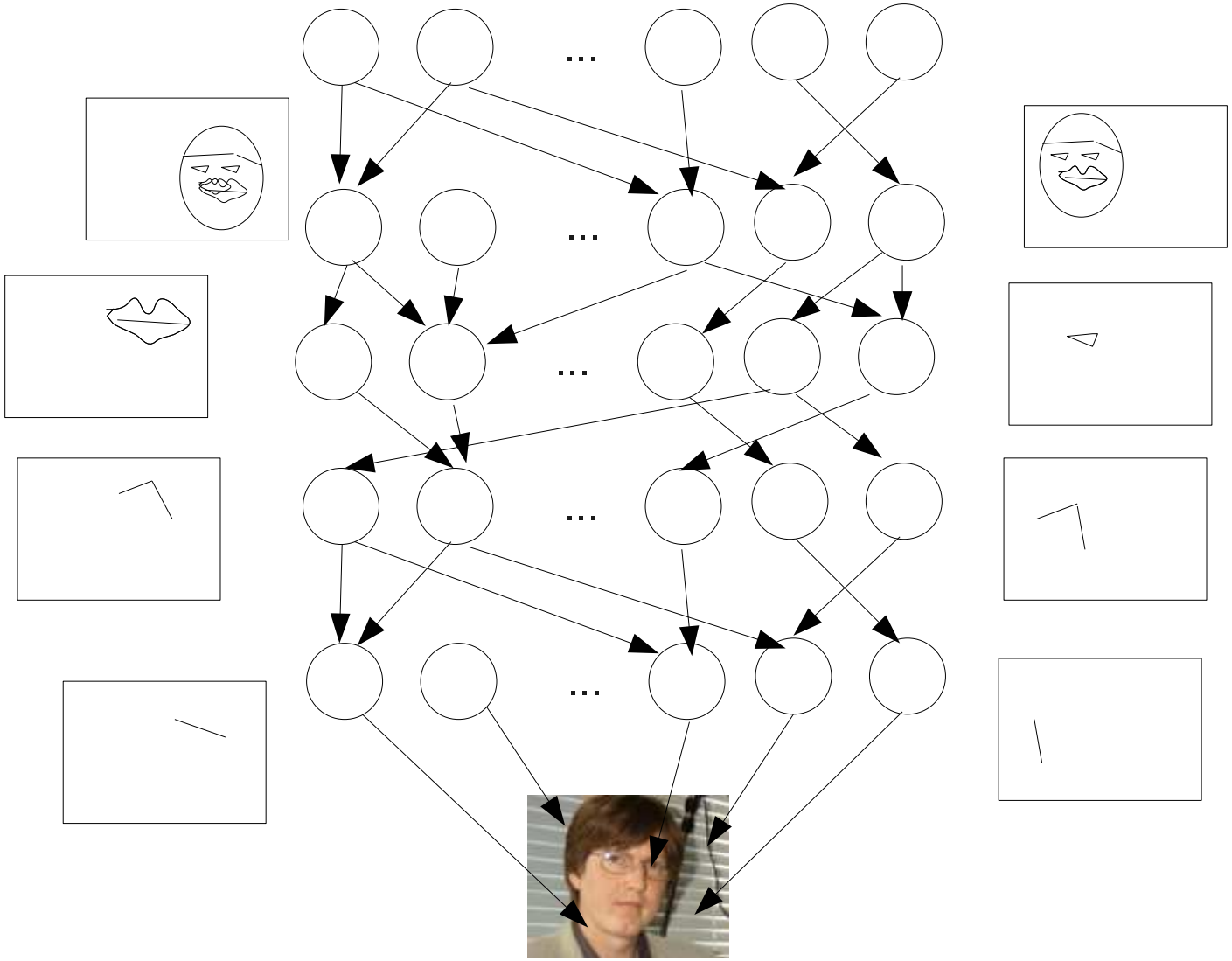
Probabilistic Neural Computation

9. Two stories about cortex

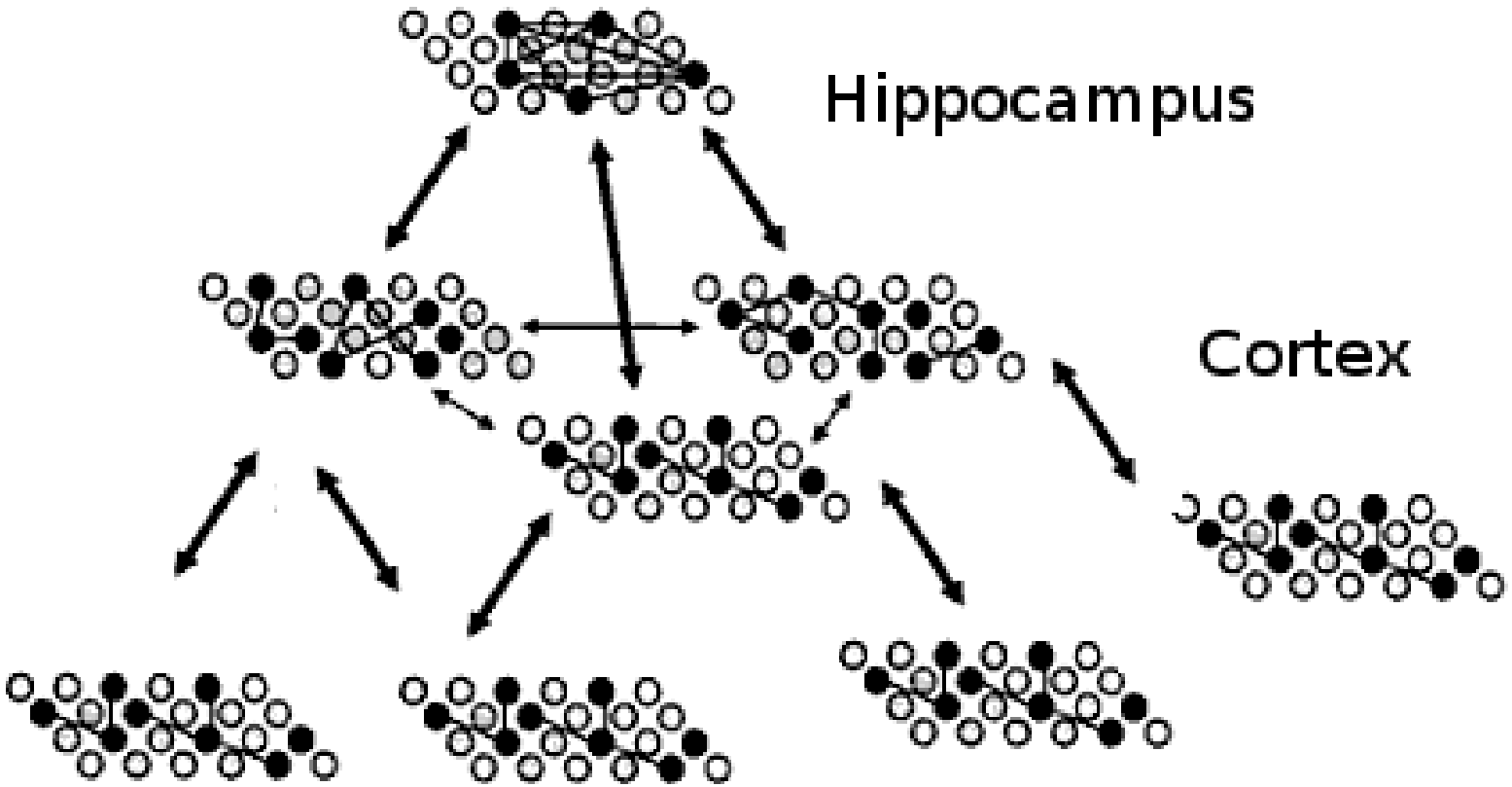
Charles Fox

March 19, 2010

Perceptual hierarchies



Marr hierarchies



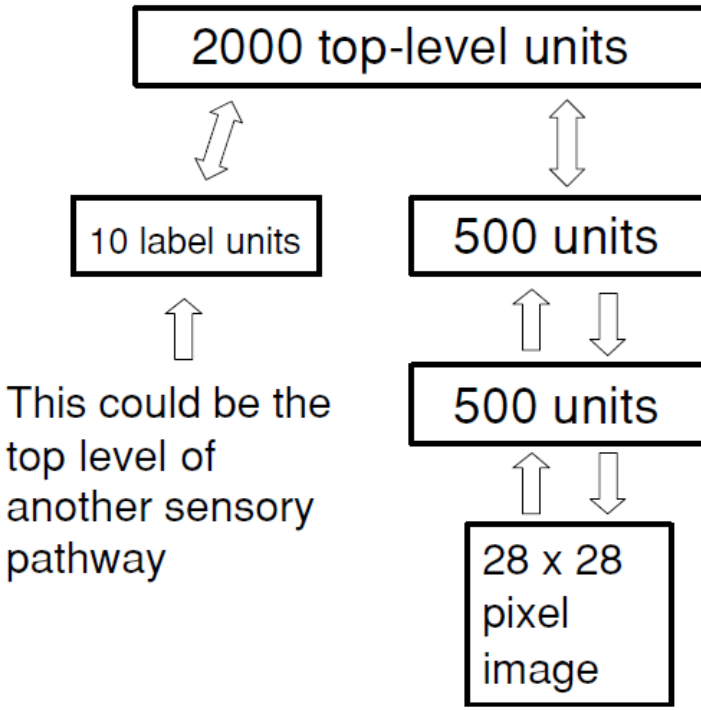
Deep Belief Networks (Hinton et al. 07)

Aim:
- Training a directed Bayesian network

Hard because:
- Explaining away
- Many local minima
- Credit assignment problem

Attempts:
- Backpropagation
- Variational inference

DBN approach:
- greedy layerwise training
- based on Resticted Boltzmann Machines
- followed by a variational layerwise retraining



Gibbs sampling recap

x is a network containing $M-1$ nodes, x_i,

$$P(x) = \frac{1}{Z'} \exp\left(\sum_{ij} x_i w_{ij} x_j + \sum_i b_i x_i\right)$$

We can simplify by assuming an extra node x_M which is always on. This lets us move the bias terms into the weight set:

$$P(x) = \frac{1}{Z'} \exp \sum_{ij} x_i w_{ij} x_j$$

Gibbs sampling recap

Probability for updating a single node,

$$P(x_i = 0) = \frac{1}{Z} \exp \sum_j 0 w_{ij} x_j = \frac{1}{Z} \exp 0$$

$$P(x_i = 1) = \frac{1}{Z} \exp \sum_j 1 w_{ij} x_j$$

$$\Rightarrow P(x_i = 1) = \frac{\exp \sum_j w_j x_j}{\exp \sum_j w_j x_j + \exp 0}$$

$$\Rightarrow P(x_i = 1) = \frac{1}{1 + \exp - \sum_j w_j x_j}$$

Is the sigmoid spiking probability!

Boltzmann training: wake-sleep

Partition into Hidden and eVidence nodes, $x = (h, v)$. Suppose we have N observed patterns of visible nodes.

$$P(\{v\}_n|w) = \prod_{n=1}^{N} \sum_h P(v, h|w)$$

$$= \prod_{n=1}^{N} \frac{1}{Z(w)} \sum_h \exp \sum_{ij} w_{ij} x_i x_j$$

Z will be important - it leads to the sleep term!

$$Z(w) = \sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j$$

Boltzmann training: wake-sleep

Let's work with log probabilities:

$$\log P(\{v\}_n|w) = \sum_n [\log \sum_h \exp \sum_{ij} w_{ij} x_i x_j - \log \sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j]$$

These will become the wake and sleep terms. We want the network to model the distribution of the data, so maximize the log prob by gradient descent:

$$\frac{d}{dw_{ij}} \log P(\{v\}_n|w)$$

$$= \sum_n \left(\frac{d}{dw_{ij}} [\log \sum_h \exp \sum_{ij} w_{ij} x_i x_j] - \frac{d}{dw_{ij}} [\log \sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j] \right)$$

Boltzmann training: wake-sleep

$$= \sum_n [\frac{d}{dw_{ij}} [\log \sum_h \exp \sum_{ij} w_{ij} x_i x_j] - \frac{d}{dw_{ij}} [\log \sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j]]$$

By the chain rule, and $\frac{d \log(x)}{dx} = \frac{1}{x}$

$$= \sum_n [\frac{\frac{d}{dw_{ij}} [\sum_h \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_h \exp \sum_{ij} w_{ij} x_i x_j} - \frac{\frac{d}{dw_{ij}} [\sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j}]$$

By chain rule on the exponentials:

$$= \sum_n [\frac{[\sum_h x_i x_j \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_h \exp \sum_{ij} w_{ij} x_i x_j} - \frac{[\sum_{v,h} x_i x_j \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j}]$$

Boltzmann training: wake-sleep

$$= \sum_n [\frac{[\sum_h x_i x_j \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_h \exp \sum_{ij} w_{ij} x_i x_j} - \frac{[\sum_{v,h} x_i x_j \exp \sum_{ij} w_{ij} x_i x_j]}{\sum_{v,h} \exp \sum_{ij} w_{ij} x_i x_j}]$$

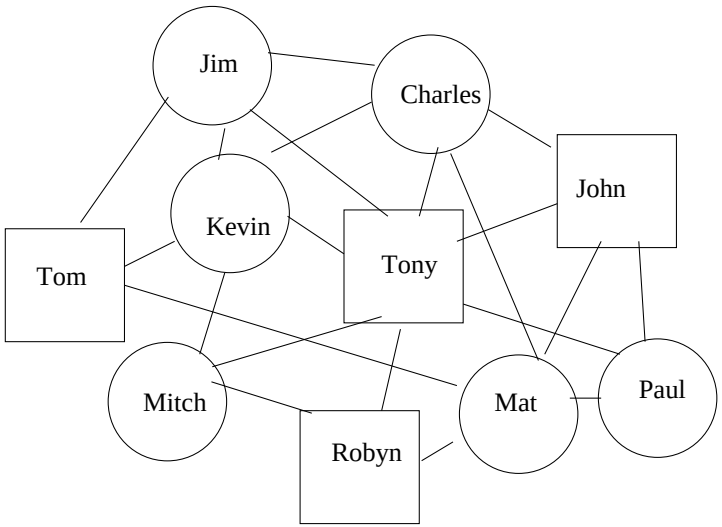
$$\sum_n [\sum_h x_i x_j P(v_n, h|w) - \sum_{v,h} x_i x_j P(v, h|w)]$$

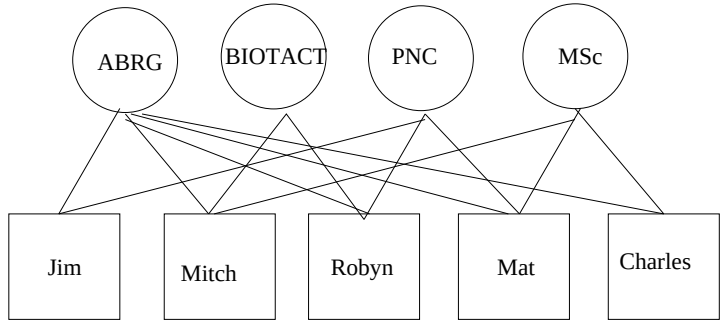
$$= \sum_n [\langle x_i x_j \rangle_{P(h|w,v_n)} - \langle x_i x_j \rangle_{P(v,h|w)}]$$

Algorithmically this says, for each pattern,

- Clamp the pattern, run the network and apply Hebbain learning
- Then unclamp it, run the network and apply anti-Hebbain learning

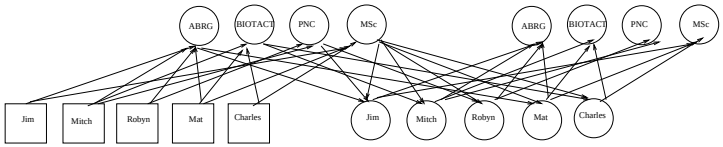
Restricted Boltzmann Machines



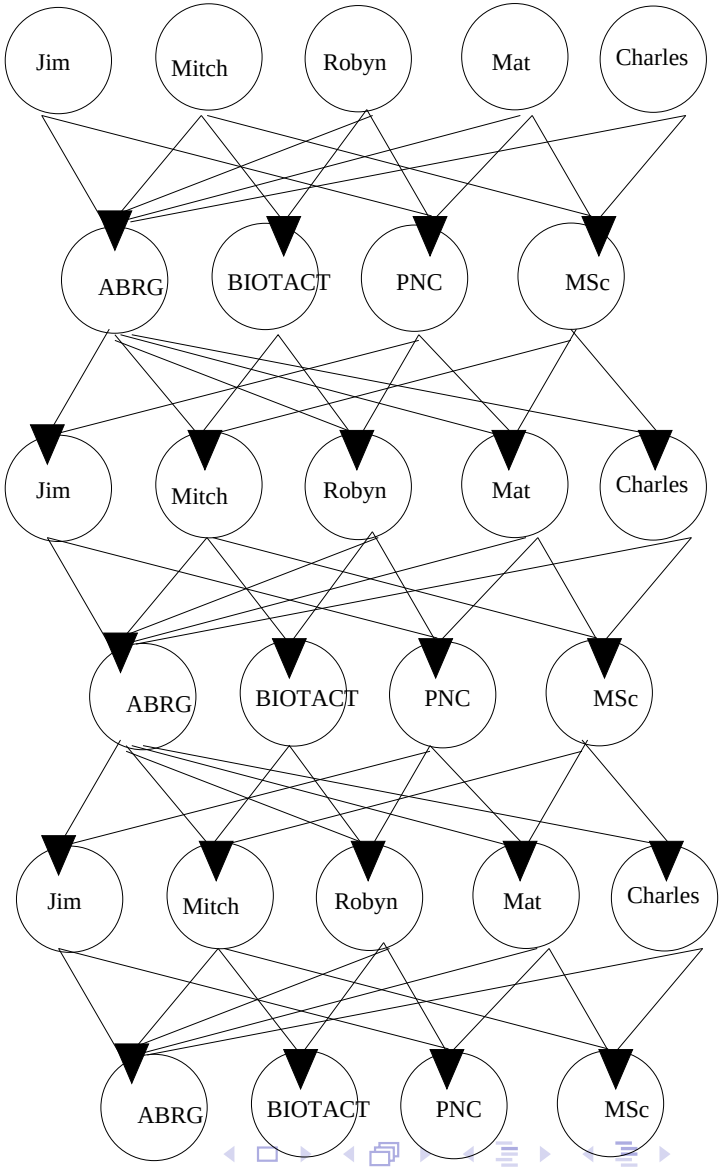


In normal BMs 'running the net' takes ages (infinite time!).
But RBM can wake-sample perfectly in one step.
Factor analysis; External field HC models
What about sleep sample?

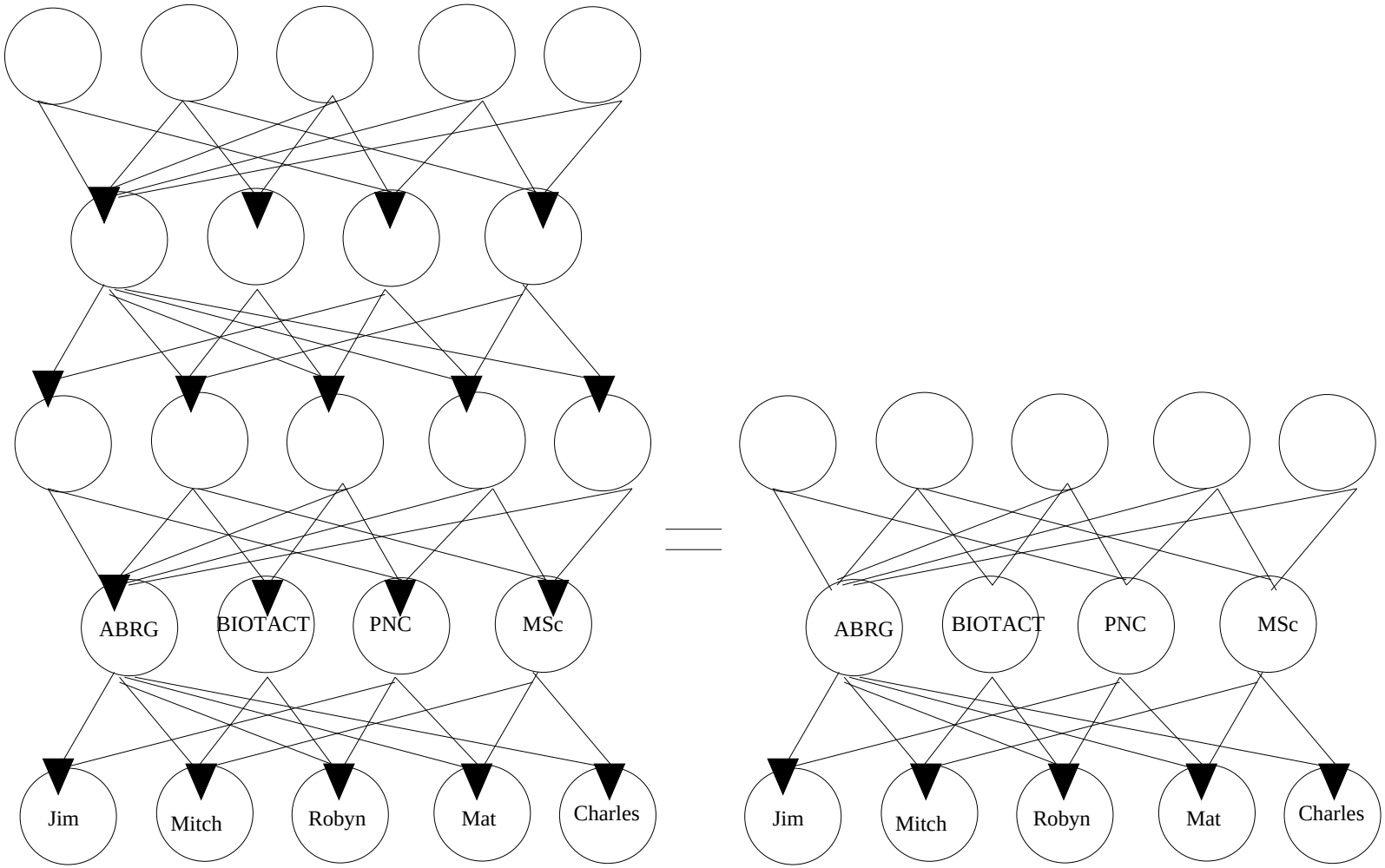
Contrastive Divergence learning in RBMs



Sleep sample for infinite time is equivilent to the prior in an infinite tied Bayes net (i.e. all weights are shared) So if we learn the RBM then we have learnt an infinite tied Bayes net! How could the Bayes net then be improved?

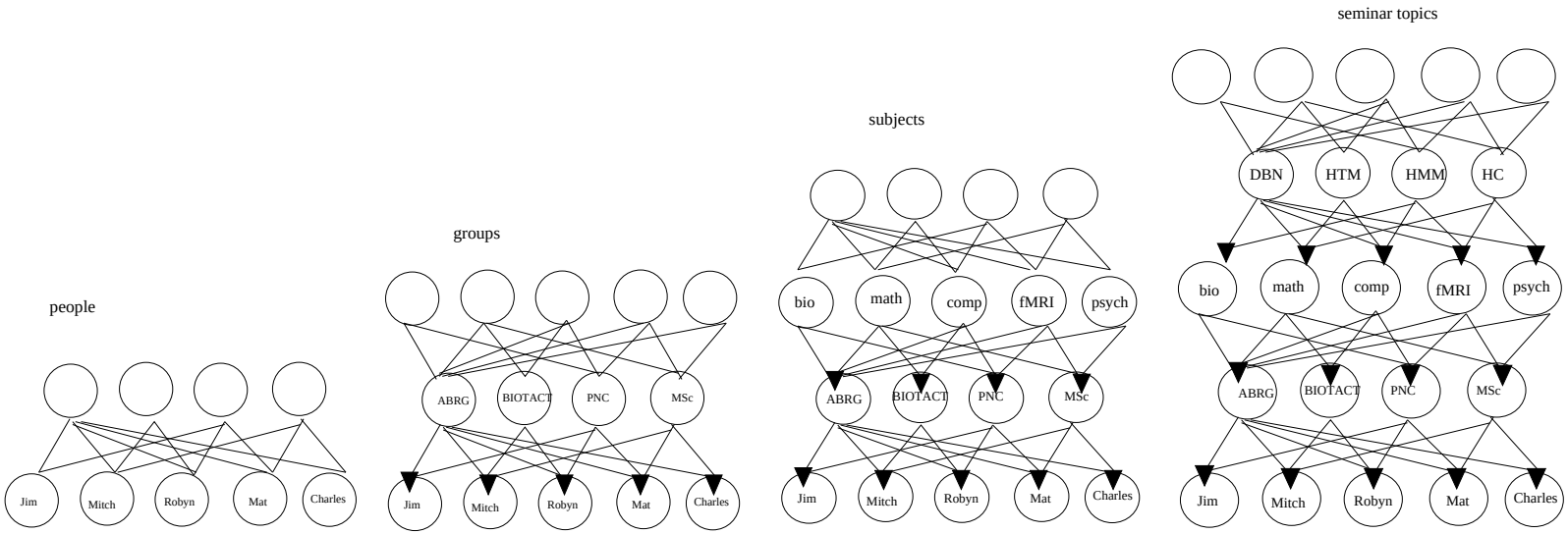


RBMs equivilent to infinite tied networks



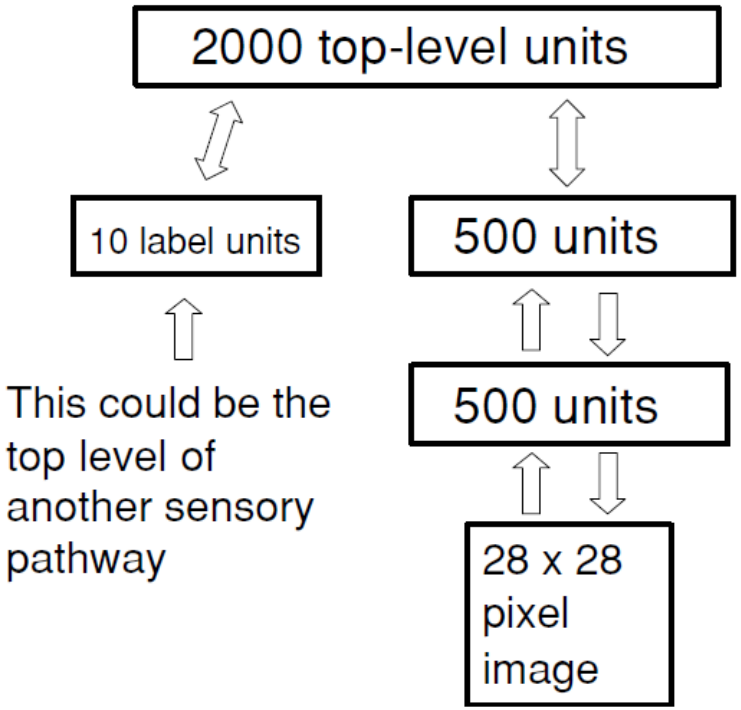
Comp prior, makes indep imagine infinite sampling chain

Greedy learning

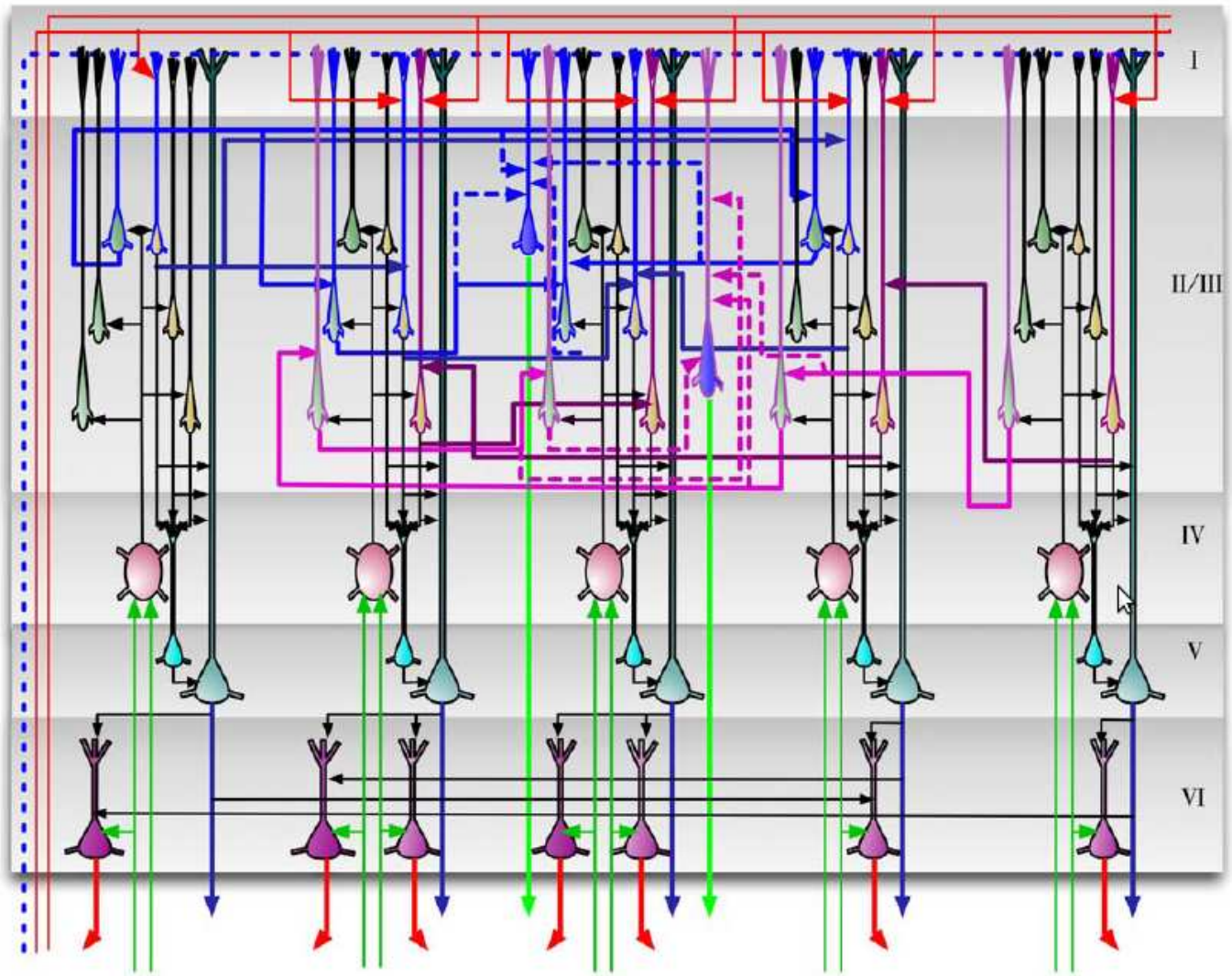


At each epoch, add a new layer and learn its weights.
Top two layers always form an RBM.
The network grows more 'cortical' layers between the input and 'hippocampus'.
(Hinton et al 2007) then uses the result to initialize a backprop-style network – this is just one application.

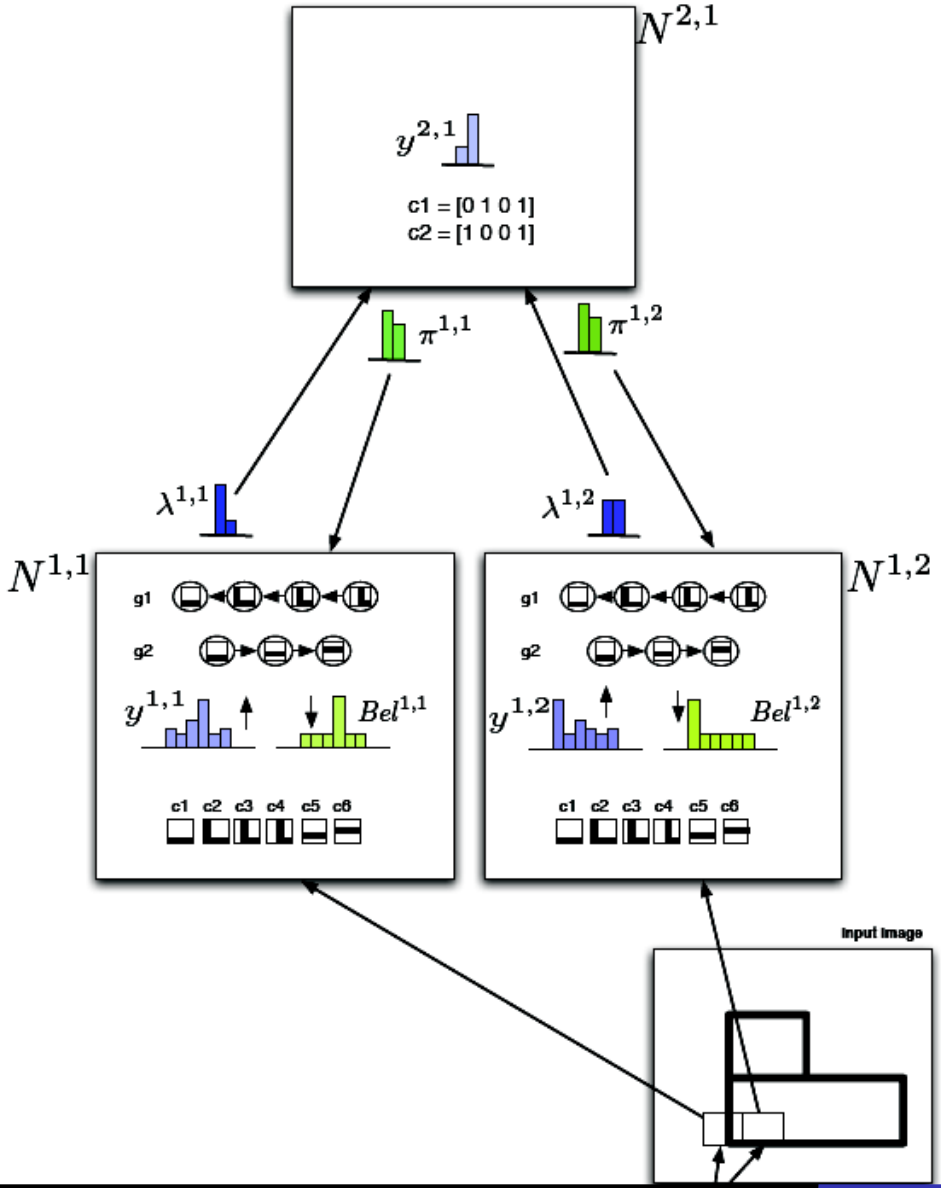
DBN character recognition



Hierarchical Temporal Memory (George&Hawkins 09)

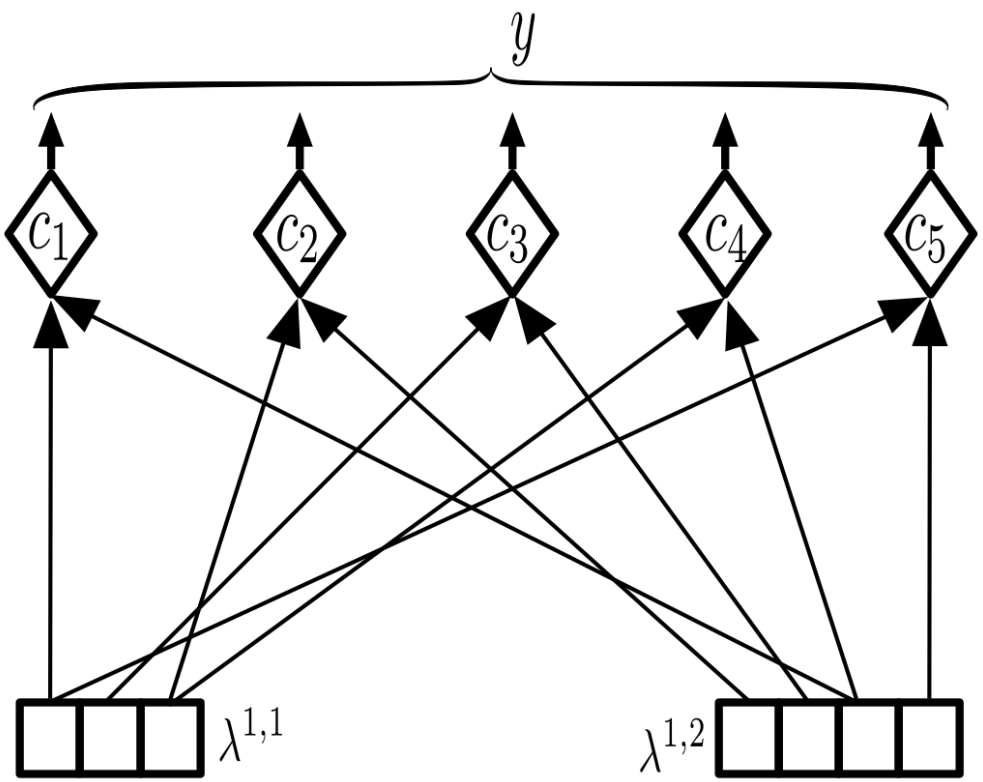


HTM toy example



Fusing data likelihoods

$$y_c(t) = P(^{-}e_t | c(t)) = \frac{1}{Z} \prod_j \lambda_t^j$$



c ranges over coincidence; y=coincidence likelihood

Fusing temporal likelihood - LIV

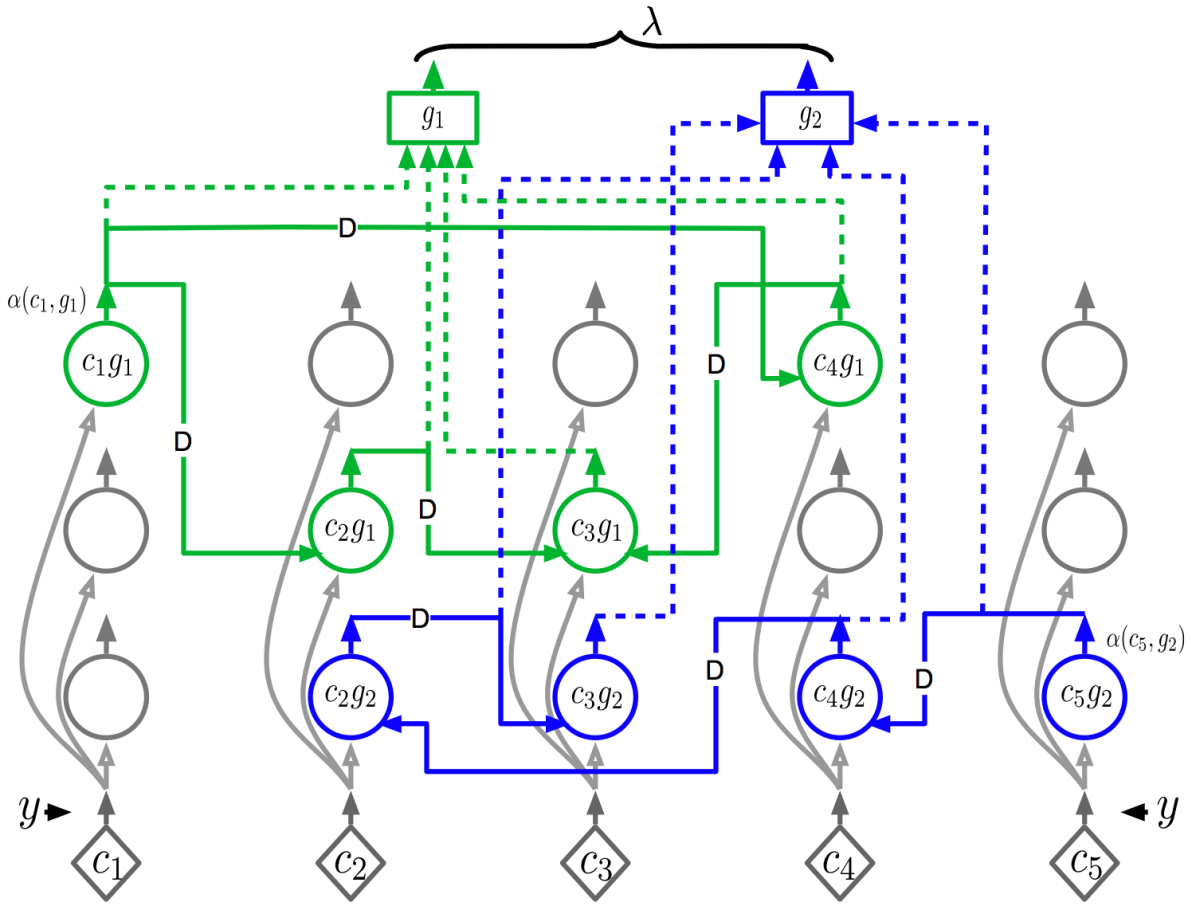
$$\lambda_t(g) = P(-e_0^t|g(t)) = \frac{1}{Z}\sum_c \alpha_t(c,g)$$

$$\alpha_t(c,g) = P(-e_0^t|c(t))\sum_{c'} P(c(t)|c'(t-1))\alpha_{t-1}(c,g)$$

$$\alpha_o(c,g) = P(-e_0|c(0))P(c(0)|g)$$

g ranges over Markov Chains
$\lambda(g)$ likelihood for a Markov Chain
$\alpha(c,g)$ dynamic programming variables

Fusing temporal likelihood - neural implementation



Like Rao – each $\alpha(c, g)$ cell computes

$\alpha_t(c, g) = P(-e_0^t|c(t)) \sum_{c'} P(c(t)|c'(t-1))$

Temporal posterior beliefs - LII/III and LV

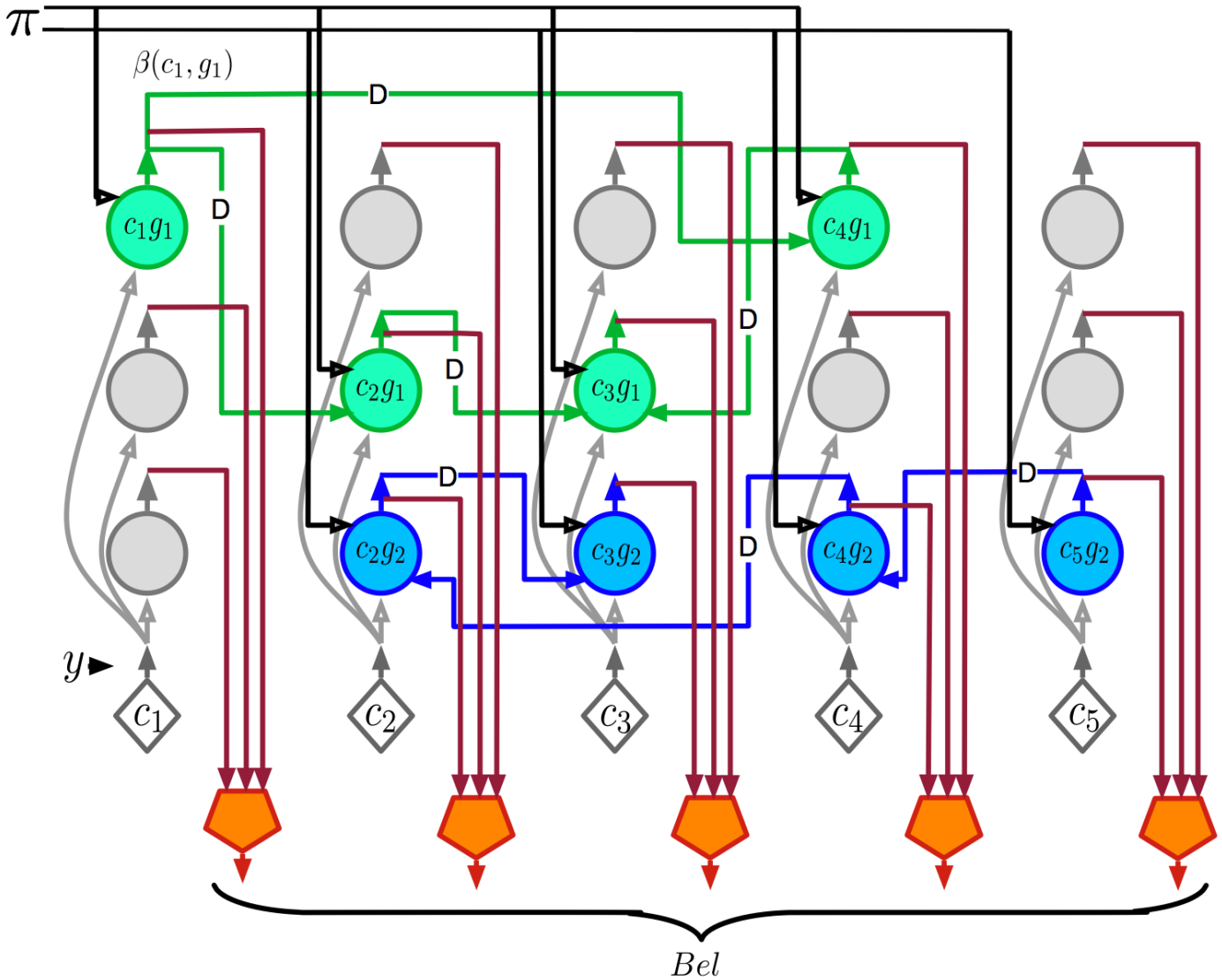
$$Bel_t(c) = P(-e_0^t|c(t))P(c(t)|+e_o(t)) = \frac{1}{Z}\sum_g \beta_t(c,g)$$

$$\beta_t(c,g) = P(-e_0^t|c(t))\sum_{c'} P(c(t)|c'(t-1))\beta_{t-1}(c,g)$$

$$\beta_o(c,g) = P(-e_0|c(0))P(c(0)|g)\pi_o(g)$$

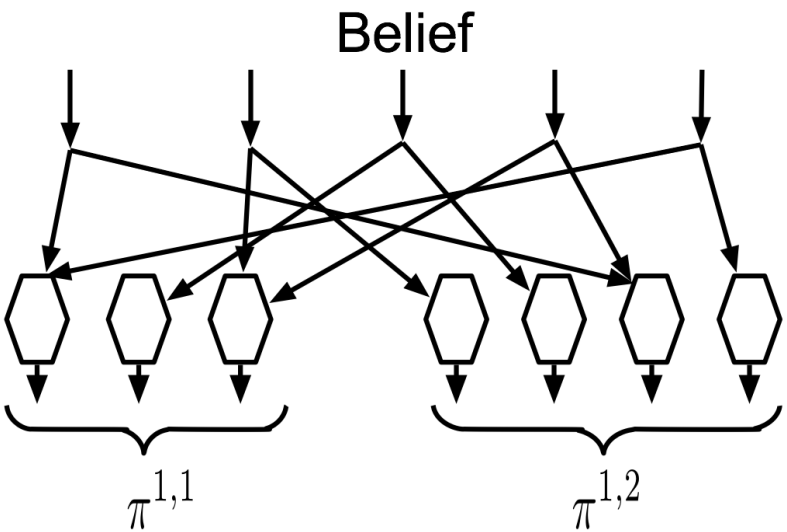
The $\beta(c,g)$ computation is identical to $\alpha(c,g)$ computation, except includes prior over Markov Chains, $\pi(g)$.
ERRATUM: in the lecture, I said that Bel was the posterior over Markov Chains, when it of course is the posterior over coincidences.

Temporal posterior beliefs - neural implementation



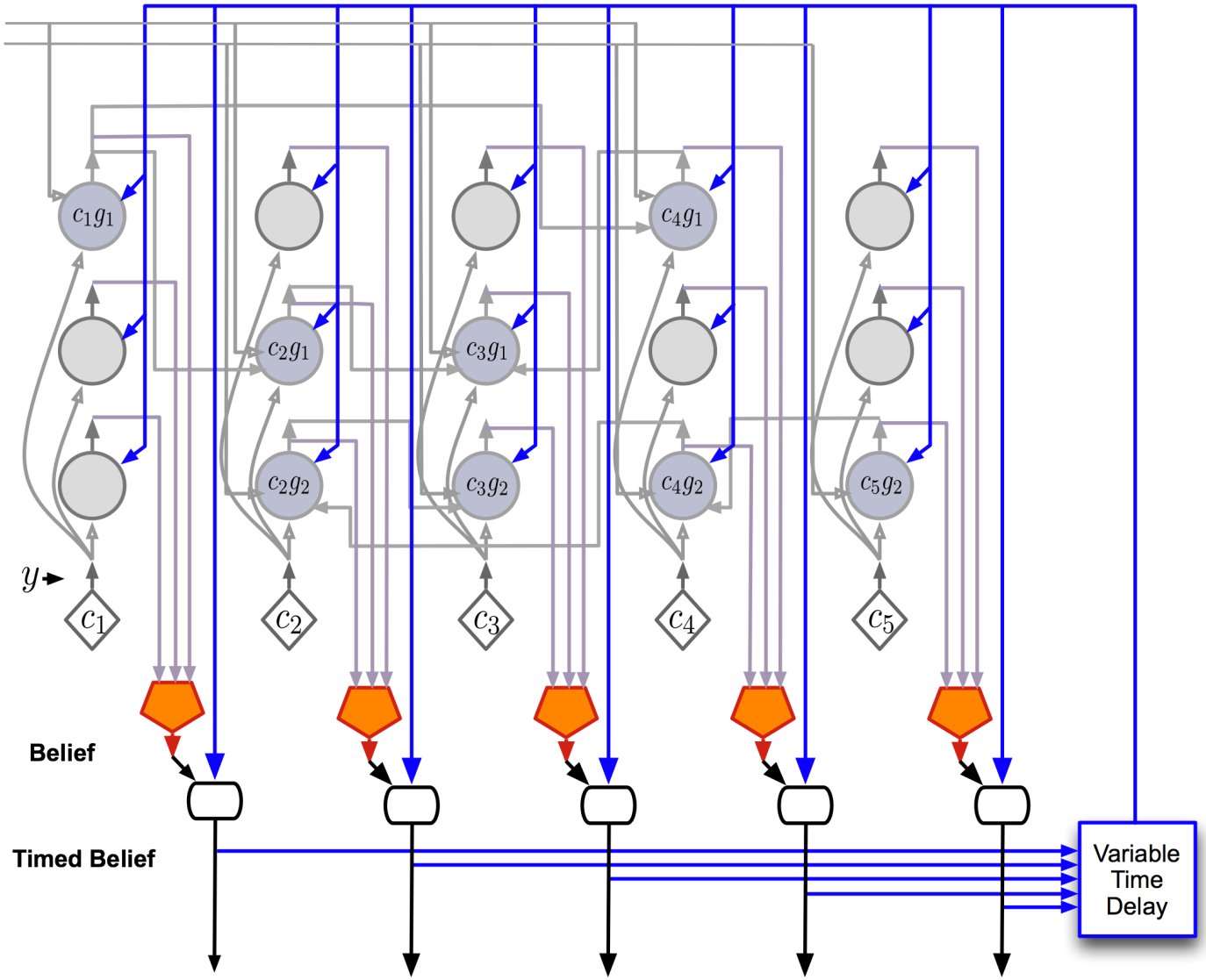
Computing prior message - Layer VI

$$\pi^m(g) = \sum_c I(c) Bel(c)$$

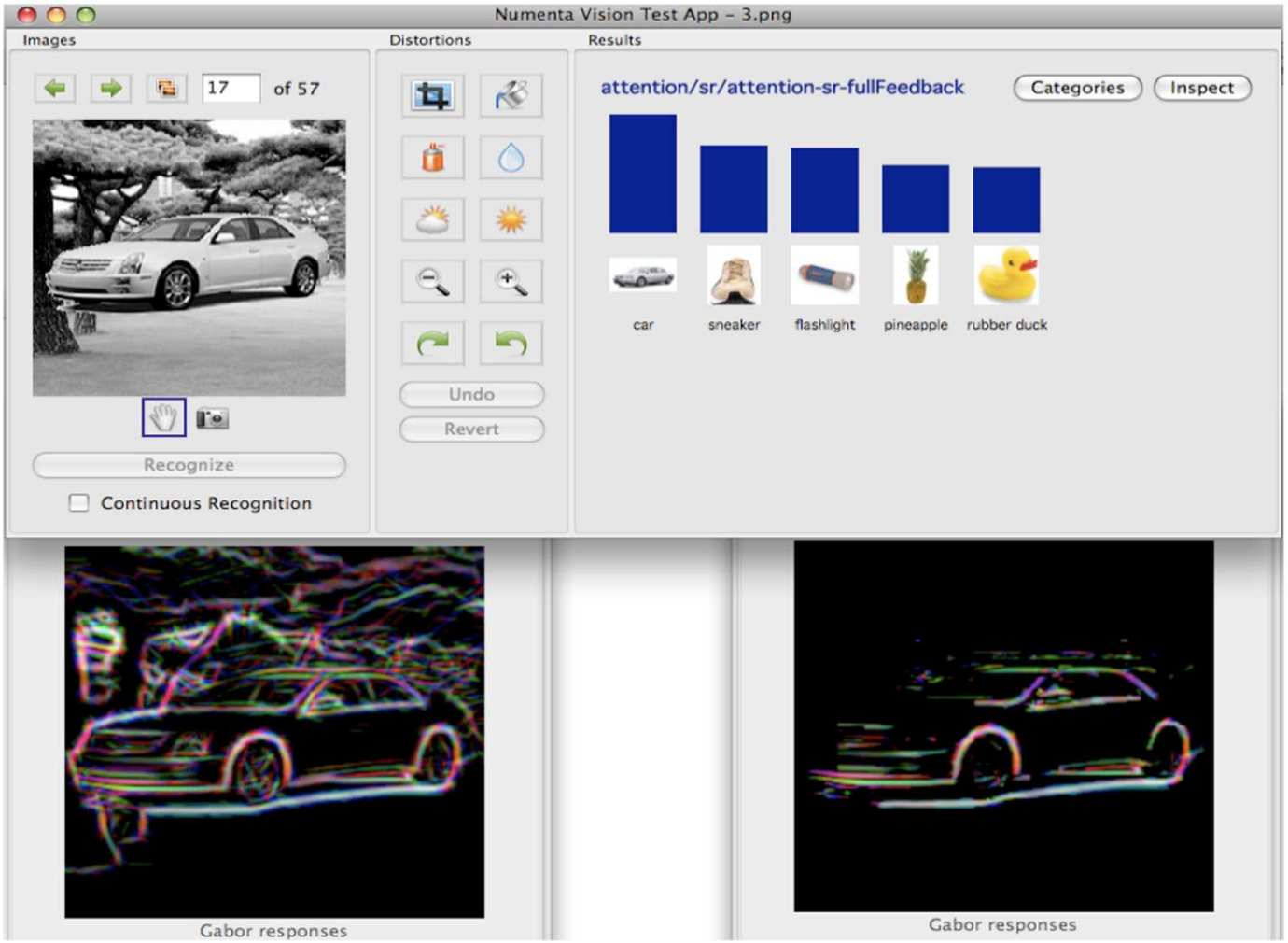


$I(c)$ is n Boolean indicator function, true if the mth child is in coincidence c.

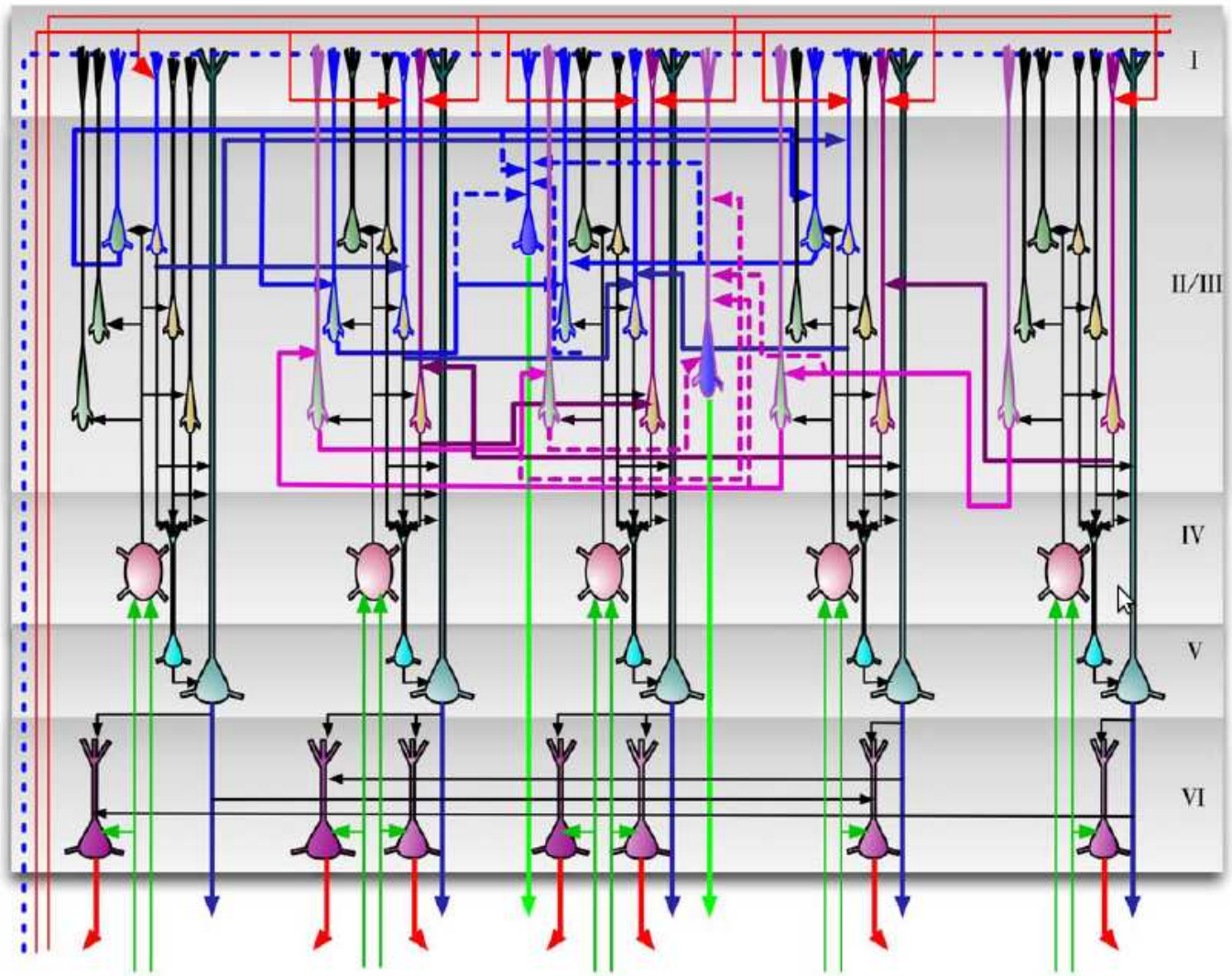
Timing circuit - Layer V



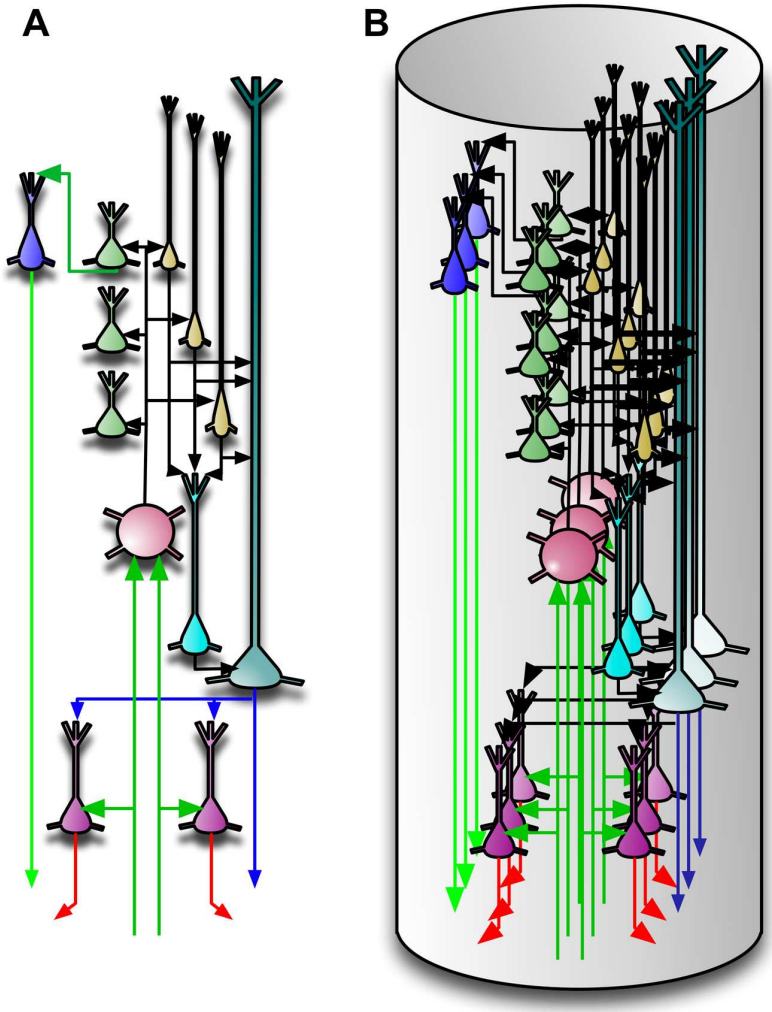
HTM vision example



Cortical mapping



Cortical column microcircuit



$$y_c(t) = P(^{-}e_t|c(t)) = \frac{1}{Z}\prod_j \lambda_t^j$$

$$\lambda_t(g) = P(-e_0^t|g(t)) = \frac{1}{Z}\sum_c \alpha_t(c,g)$$

$$\alpha_t(c,g) = P(-e_0^t|c(t))\sum_{c'} P(c(t)|c'(t-1))\alpha_{t-1}(c,g)$$

$$\alpha_o(c,g) = P(-e_0|c(0))P(c(0)|g)$$

$$Bel_t(g) = P(-e_0^t|g(t))P(g(t)|+e_o(t)) = \frac{1}{Z}\sum_c \beta_t(c,g)$$

$$\beta_t(c,g) = P(-e_0^t|c(t))\sum_{c'} P(c(t)|c'(t-1))\beta_{t-1}(c,g)$$

$$\beta_o(c,g) = P(-e_0|c(0))P(c(0)|g)\pi_o(g)$$

$$\pi^m(g) = \sum_c I(c)Bel(c)$$

PNC Course summary

Hour 1: Introduction to Bayesian networks
Hour 2: Maths practical
Hour 3: Pearl message passing
Hour 4: Matlab/BNT Bayes nets practical
Hour 5: Neural Hidden Markov Models
Hour 6: HMM Matlab practical
Hour 7: Hippocampus: HMM vs Boltzmann models
Hour 8: Hippocampus: Boltzmann machine practical
Hour 9: Cortex: Deep Belief Networks
Hour 10: Cortex: Hierarchical Temporal Memory

PNC Exam style

- To pass the exam – understand the course concepts, carry out hand calculations of Pearl, HMM, Boltzmann and Hopfield nets; give summaries of Rao, HC models, Hawkins, Hinton.
- For distinction: do your own reading and thinking. In particular, give novel criticisms of the Rao, Hippocampus and Cortex models, either/both from a biological or computing viewpoint. Read the original papers on the models and anatomy, find papers that cite them, debate them with your collegues. Do something interesting!