Towards blind source separation with Boltzmann machines

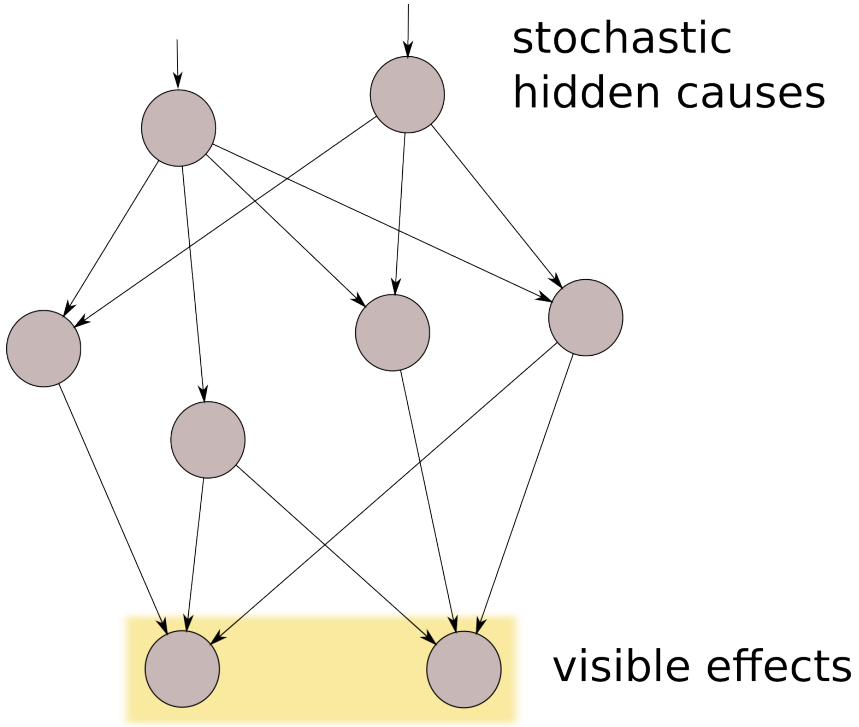
Marcus Frean

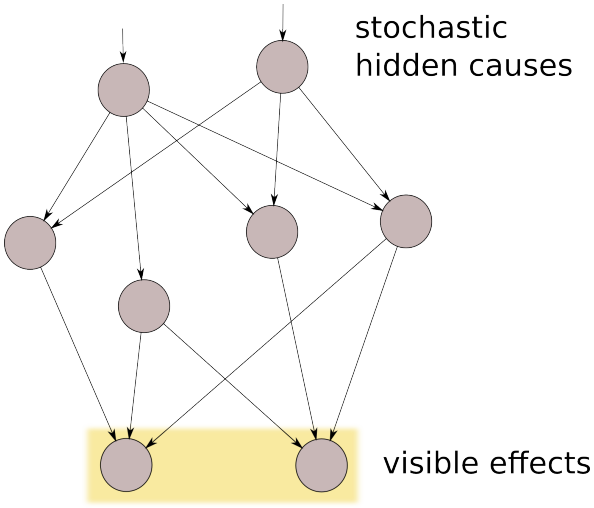
Victoria / *Te Punaha Matatini* (CoRE in complexity and networks)

This is work with Stephen Marsland and Max Godfrey

belief nets

- A belief net is a directed acyclic graph composed of stochastic variables
- We get to observe some of the variables and we would like to solve two problems:
- The inference problem: Infer the states of the unobserved variables.
- The learning problem: Adjust the interactions between variables to make the network more likely to generate the observed data.





SAMPLING FROM THE JOINT IS EASY

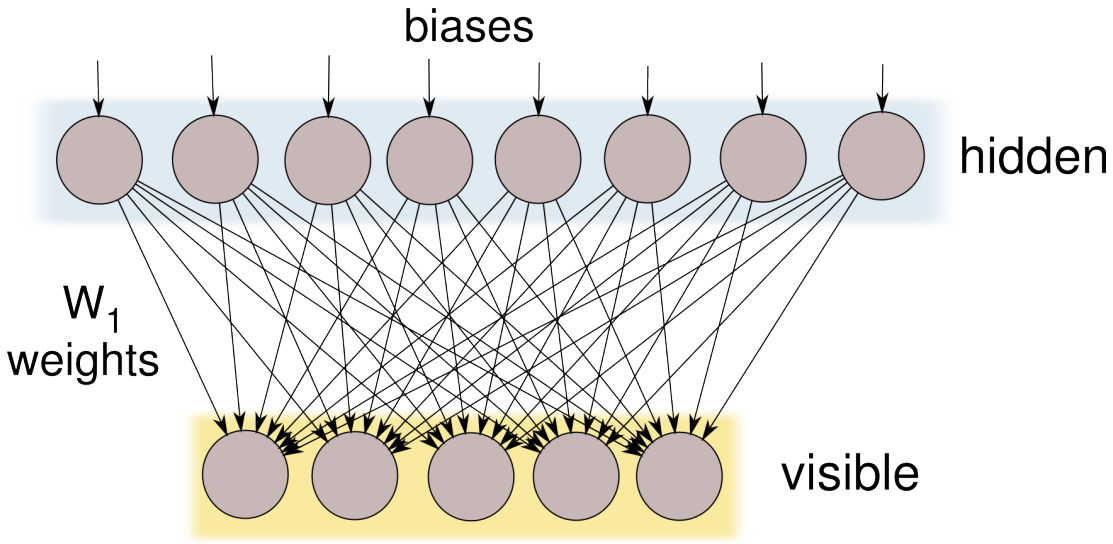
$$P(\mathbf{x}) = \prod_i P(x_i \mid \text{parents}_i)$$

we can just work through from start to finish ("down" the belief net from parents to children all the way), drawing each from its conditional distribution. And this gives us a bona fide sample from $\Pr(\mathbf{x})$.

Nomenclature:

- split $\mathbf{x}$ into $\mathbf{v}$ (for "visible"), and $\mathbf{h}$ (for "hidden").

"sigmoidal" belief nets



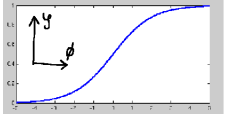
W are the params: "weights" in a "neural network"

This is a model of multiple (albeit simple) causes

$$P(v_i = 1 \mid h) = \sigma(\phi_i)$$

where $\phi_i = \sum_j W_{ji} h_j$ linear

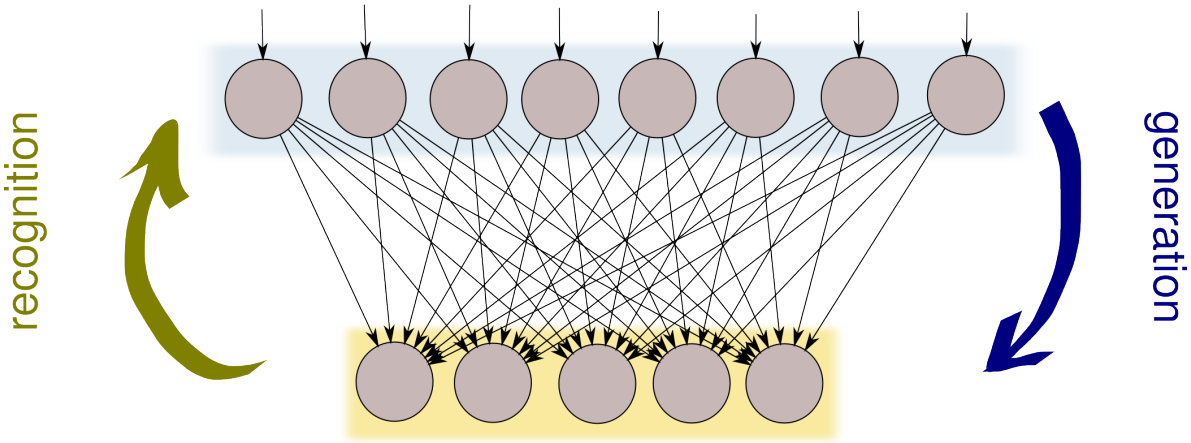
and $\sigma(\phi) = \frac{1}{1 + e^{-\phi}}$ sigmoid



hard

This is a generative model of $\mathbf{v}$ in terms of multiple causes.

Sampling from generative model: trivial.

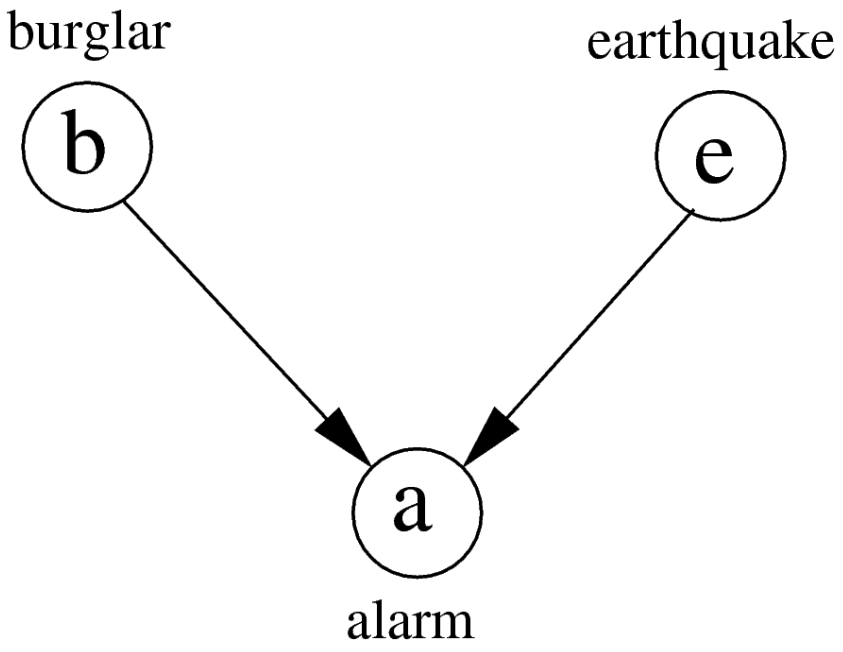


We want to "invert" it via Bayes theorem to get $P(\mathbf{h} \mid \mathbf{v})$.

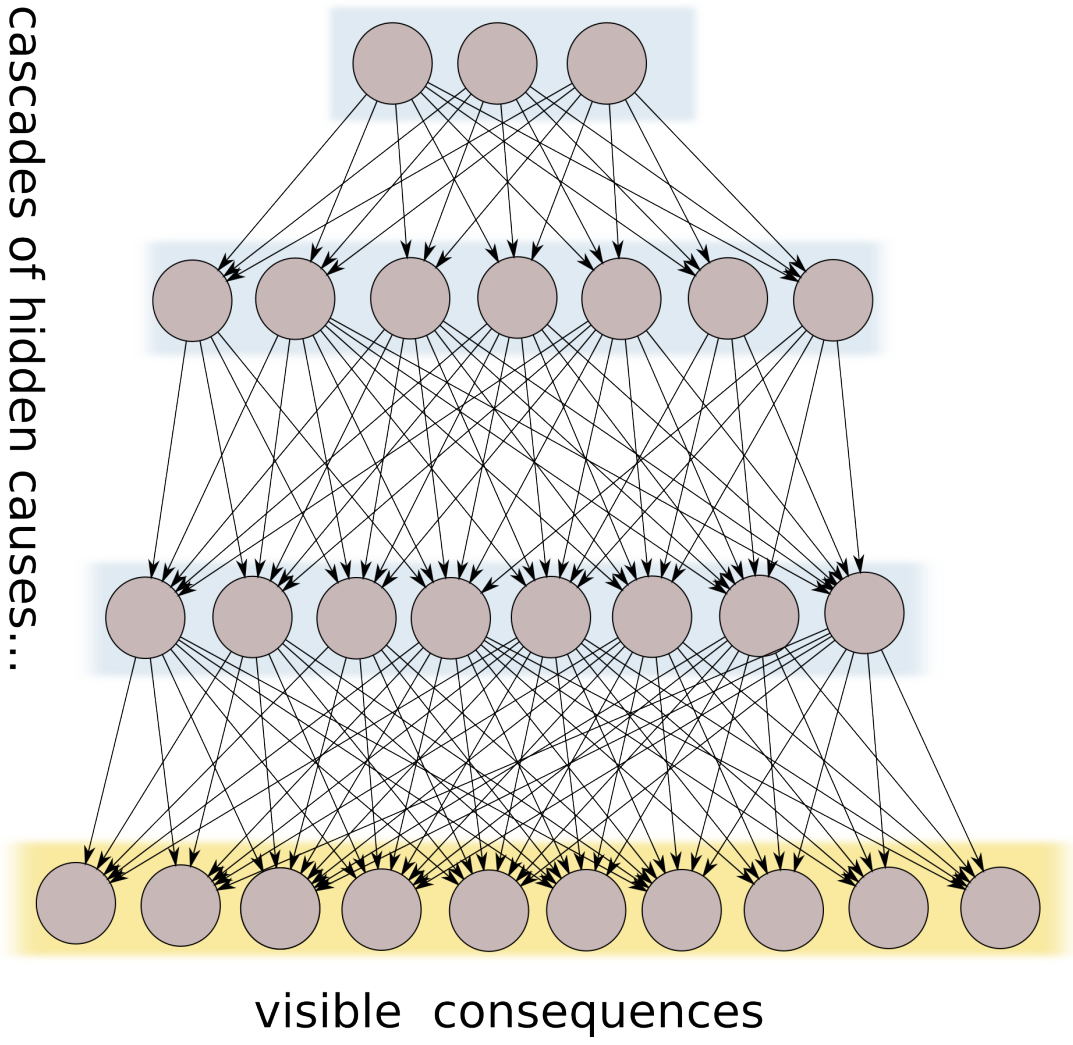
However $P(\mathbf{h} \mid \mathbf{v})$ doesn't factor...

$$P(\mathbf{h} \mid \mathbf{v}) \neq \prod_j P(h_j \mid \mathbf{v})$$

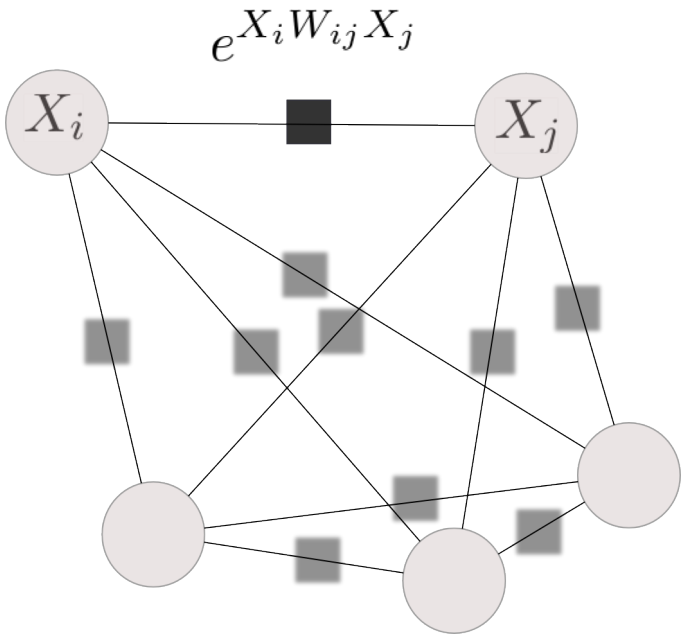
why it's hard: "explaining away"



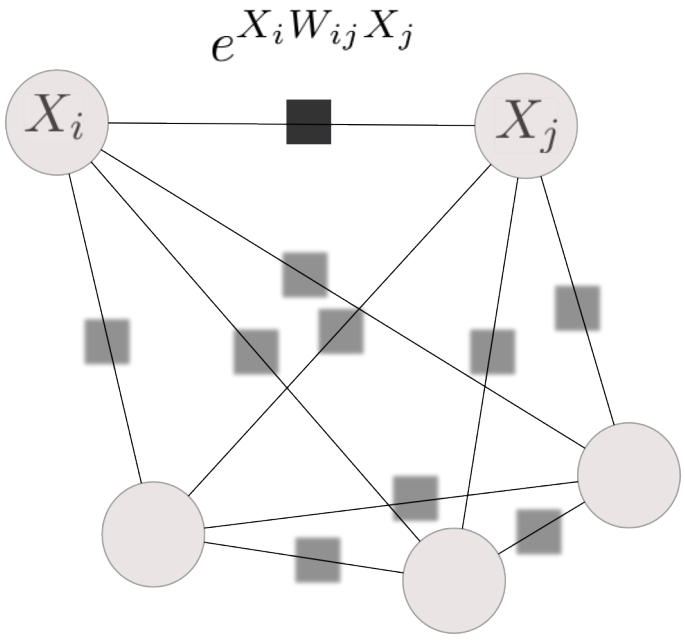
the real goal: deep belief nets

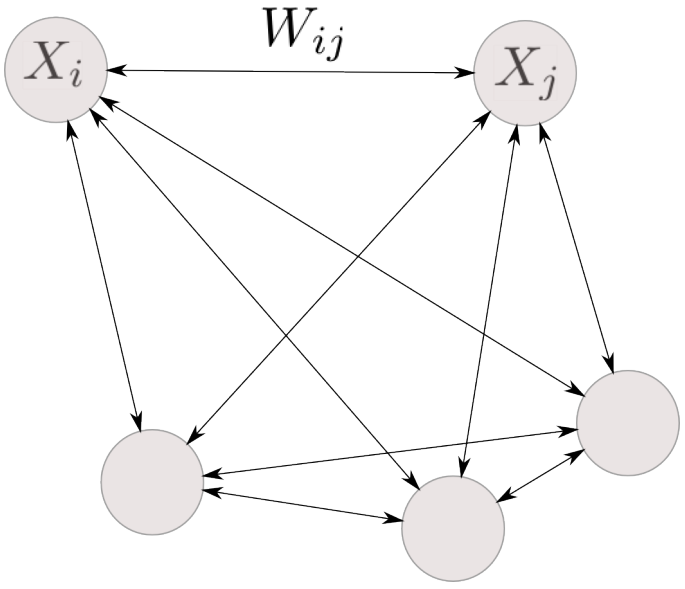


cf. the Boltzmann Machine

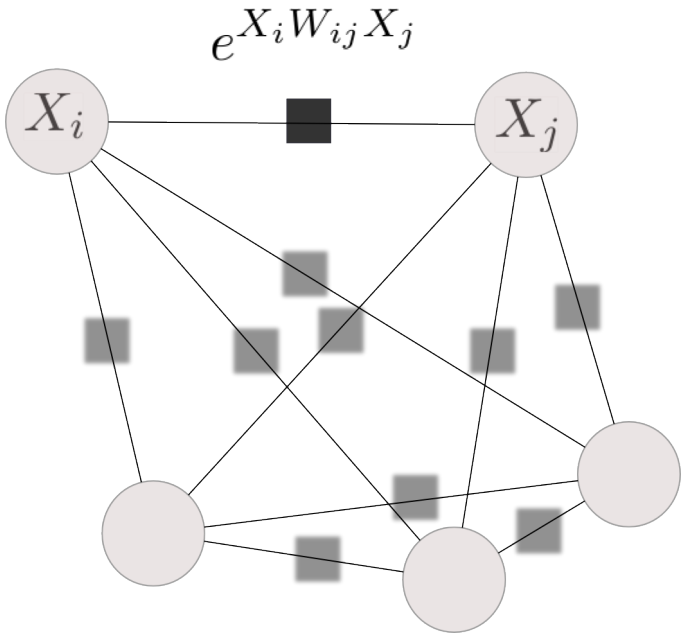


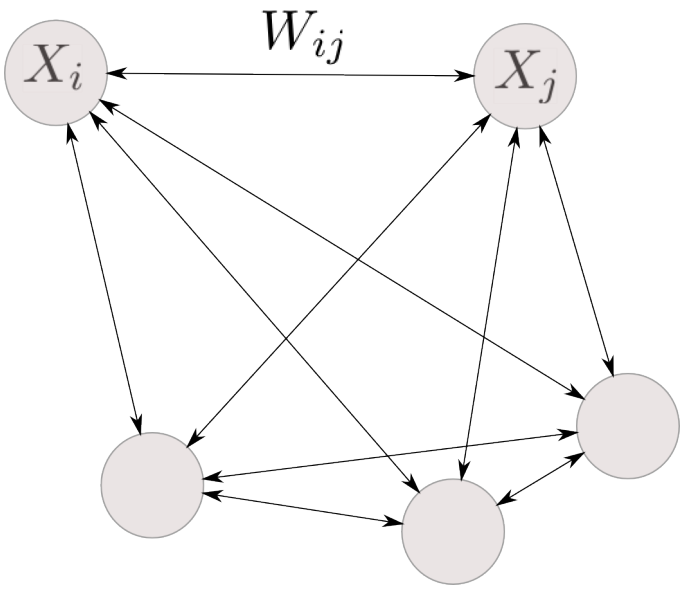
cf. the Boltzmann Machine





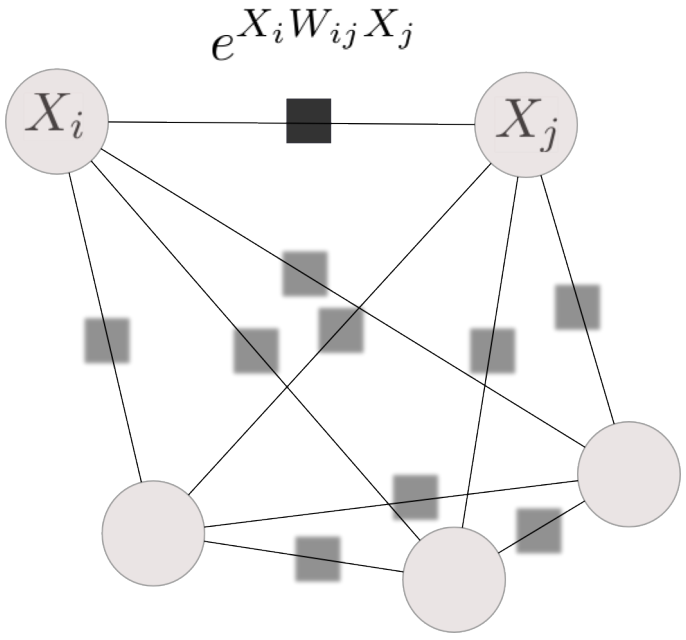
cf. the Boltzmann Machine

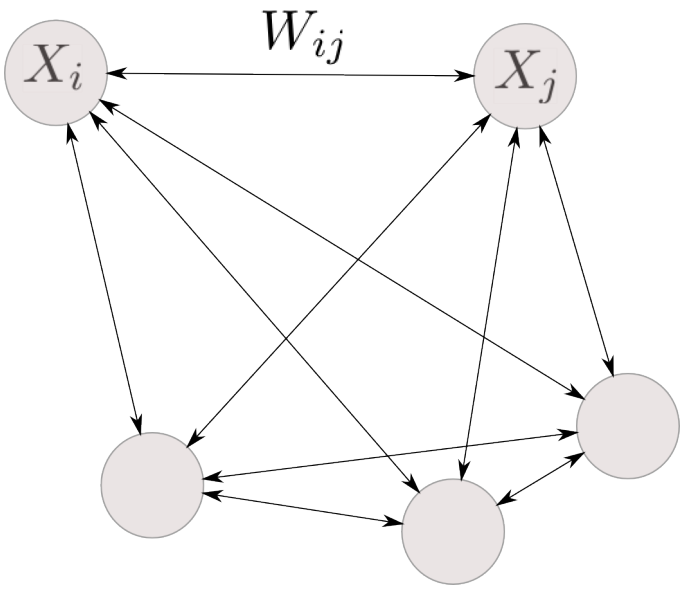




Joint is a product of all the factors: $P(\mathbf{x}) \propto \prod_i \prod_{j<i} e^{x_i W_{ji} x_j}$

cf. the Boltzmann Machine

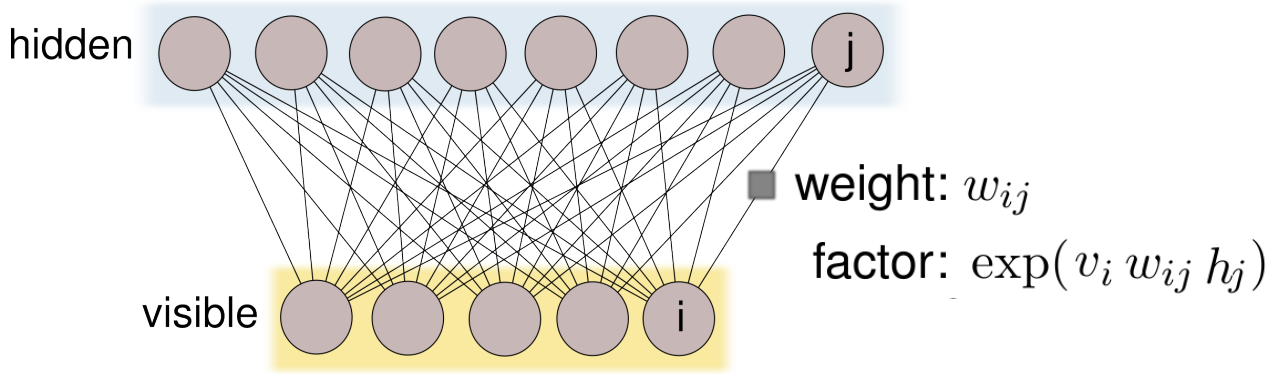




Joint is a product of all the factors: $P(\mathbf{x}) \propto \prod_i \prod_{j<i} e^{x_i W_{ji} x_j}$

Gibbs Sampling: $P(x_i \mid \mathbf{x}_{\setminus i}) = \sigma(\phi_i)$ with $\phi_i = \sum_{j \neq i} W_{ji} x_j$

Restricted Boltzmann machine (RBM)



$$P(\mathbf{h}, \mathbf{v}) \propto \prod_i \prod_j \exp(v_i h_j W_{ji})$$

$$P(\mathbf{h}|\mathbf{v}) \propto \prod_j P(h_j \mid \mathbf{v})$$ ie. it factors

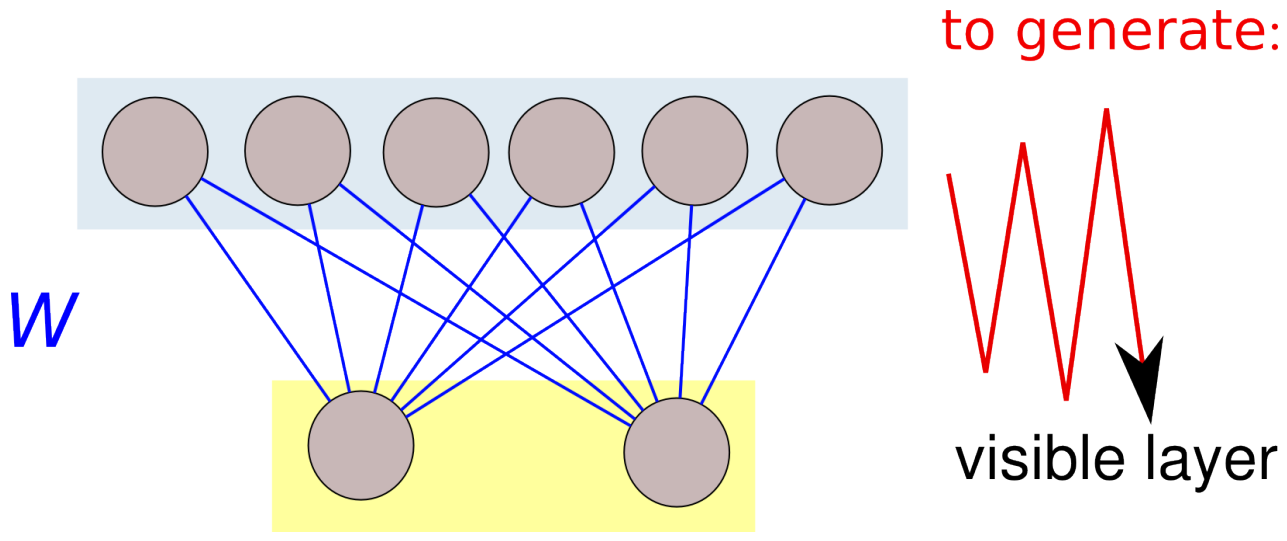
Moreover... $P(h_j = 1|\mathbf{v}) = \sigma(\psi_j)$

with $\psi_j = \sum_i W_{ji} v_i$

just like the visibles did.

Restricted Boltzmann machine (RBM)

NB. sampling from the generative model involves MCMC (**Gibbs Sampling**).



ie: The probability $p(h_j = 1|v)$ that the j-th hidden unit generates a 1 can be written as $\sigma(\psi_j)$ with $\sigma(x) = 1/(1 + e^{-x})$ and $\psi_j = \log P^*(h, v|h_j = 1) - \log P^*(h, v|h_j = 0)$. And from the $\log P^\star(h, v)$ given above, this is $\psi_j(v) = \sum_i W_{ji} v_i$

(And similarly, for the visible units)

Aside : the marginal distribution over visible patterns in an RBM

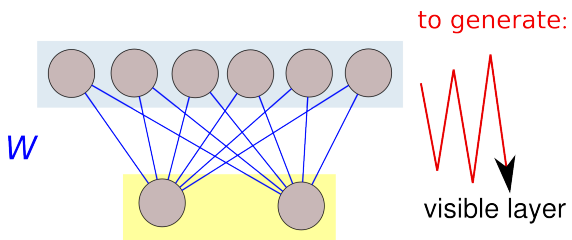
$$
\begin{aligned}
P^\star(\mathbf{v}) &= \sum_{h_1=0}^{1} \cdots \sum_{h_n=0}^{1} \exp\Big[\log P^\star(h, v)\Big] \\
&= \sum_{h_1=0}^{1} \cdots \sum_{h_n=0}^{1} \exp\Big[\sum_j \sum_j h_j W_{ji} v_i \Big] \\
&= \sum_{h_1=0}^{1} \cdots \sum_{h_n=0}^{1} \exp\Big[\sum_j h_j \psi_j(\mathbf{v}) \Big] \\
&= \sum_{h_1=0}^{1} \cdots \sum_{h_n=0}^{1} \prod_j \exp\Big[h_j \psi_j(\mathbf{v})\Big] \\
&= \prod_i \Big(1 + e^{\psi_j(\mathbf{v})}\Big)
\end{aligned}
$$

and so $\log P^\star(\mathbf{v}) = \sum_j \log \Big(1 + e^{\psi_j(\mathbf{v})}\Big)$

(a sum of rectified linear functions: “hinges”)

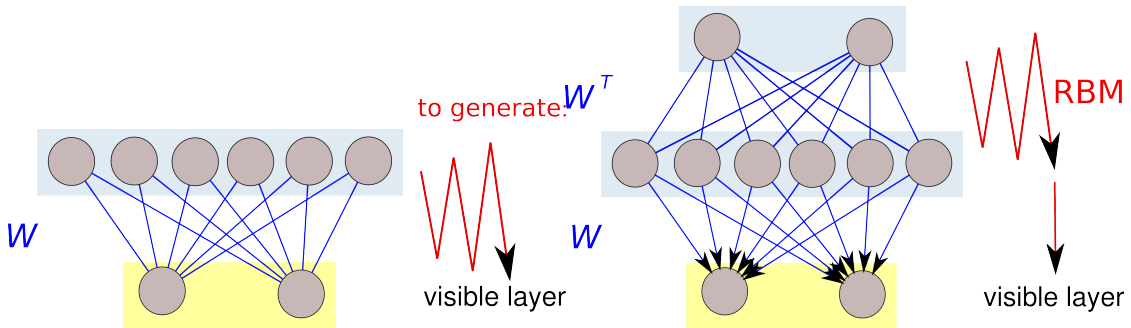
Aside 2: RBMs *are* deep Sigmoid belief nets

Sampling from an RBM is $\equiv$ sampling from an RBM with a Sigmoidal Belief Net at the last step. The two layers have *shared weights*.



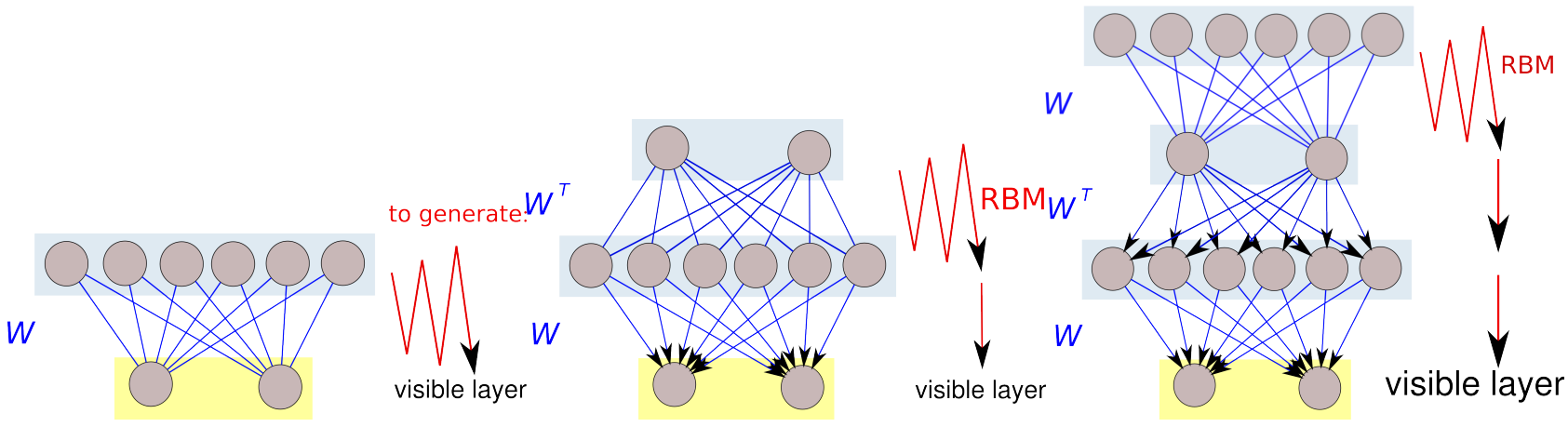
Aside 2: RBMs *are* deep Sigmoid belief nets

Sampling from an RBM is $\equiv$ sampling from an RBM with a Sigmoidal Belief Net at the last step. The two layers have *shared weights*.

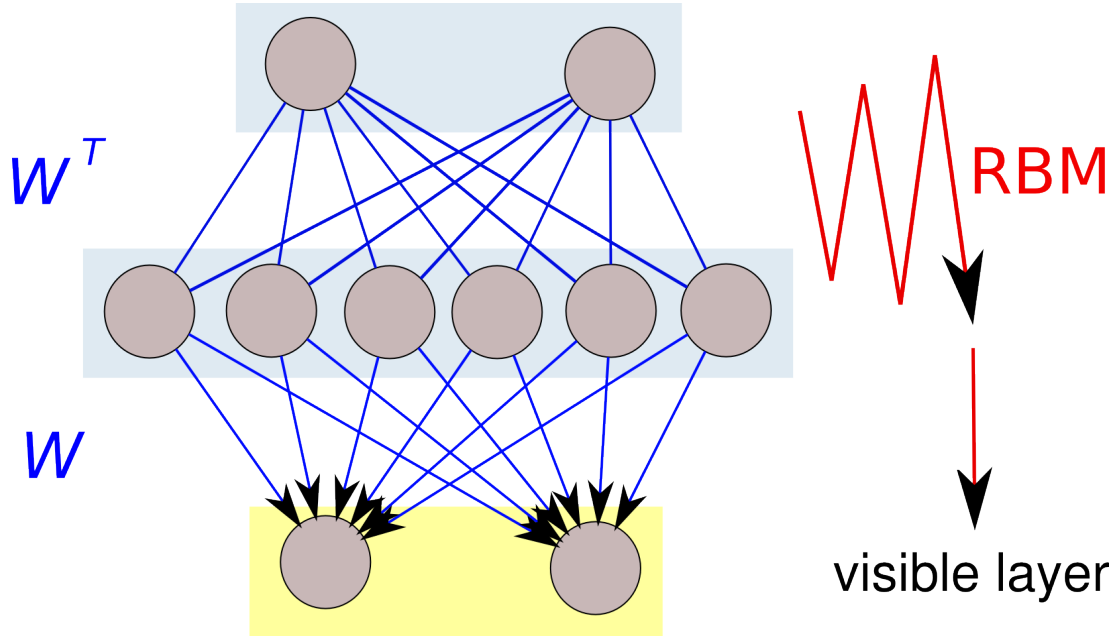


Aside 2: RBMs *are* deep Sigmoid belief nets

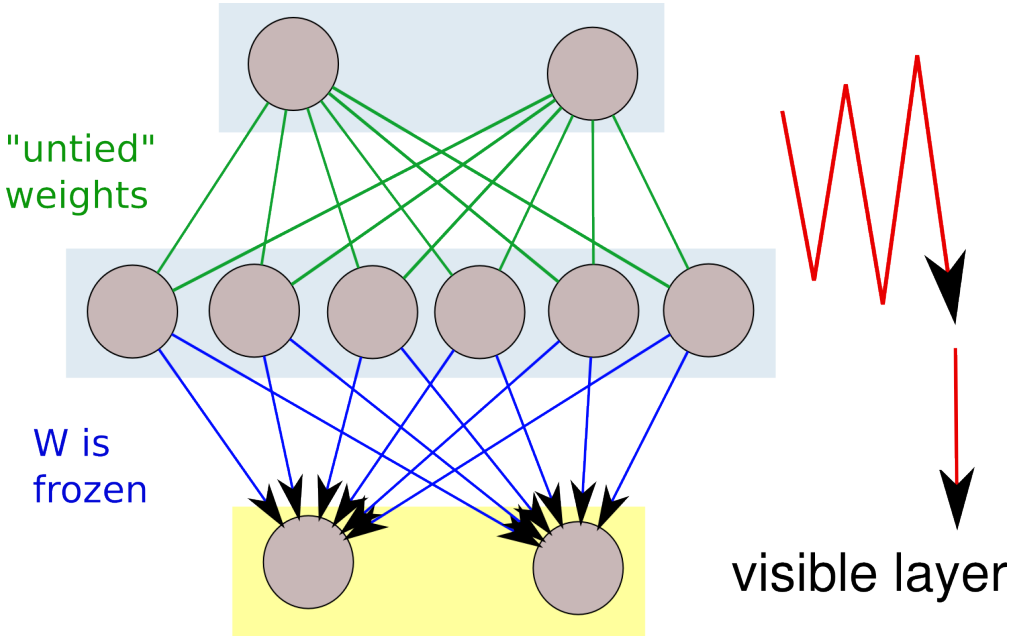
Sampling from an RBM is $\equiv$ sampling from an RBM with a Sigmoidal Belief Net at the last step. The two layers have *shared weights*.



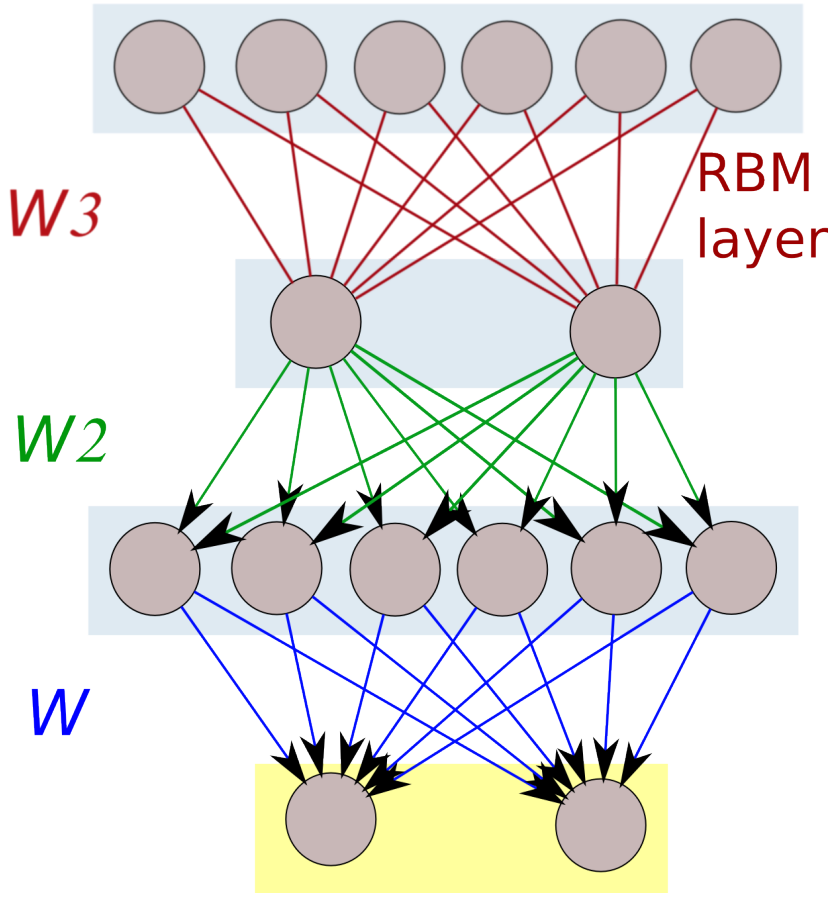
Aside 3: learning deep nets from RBMs



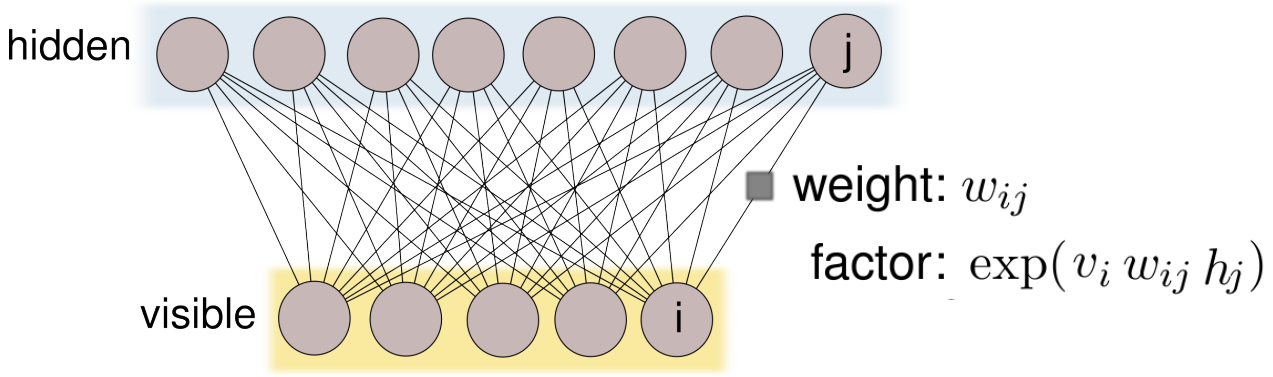
Aside 3: learning deep nets from RBMs



Aside 3: learning deep nets from RBMs



Great. But baby, bathwater?



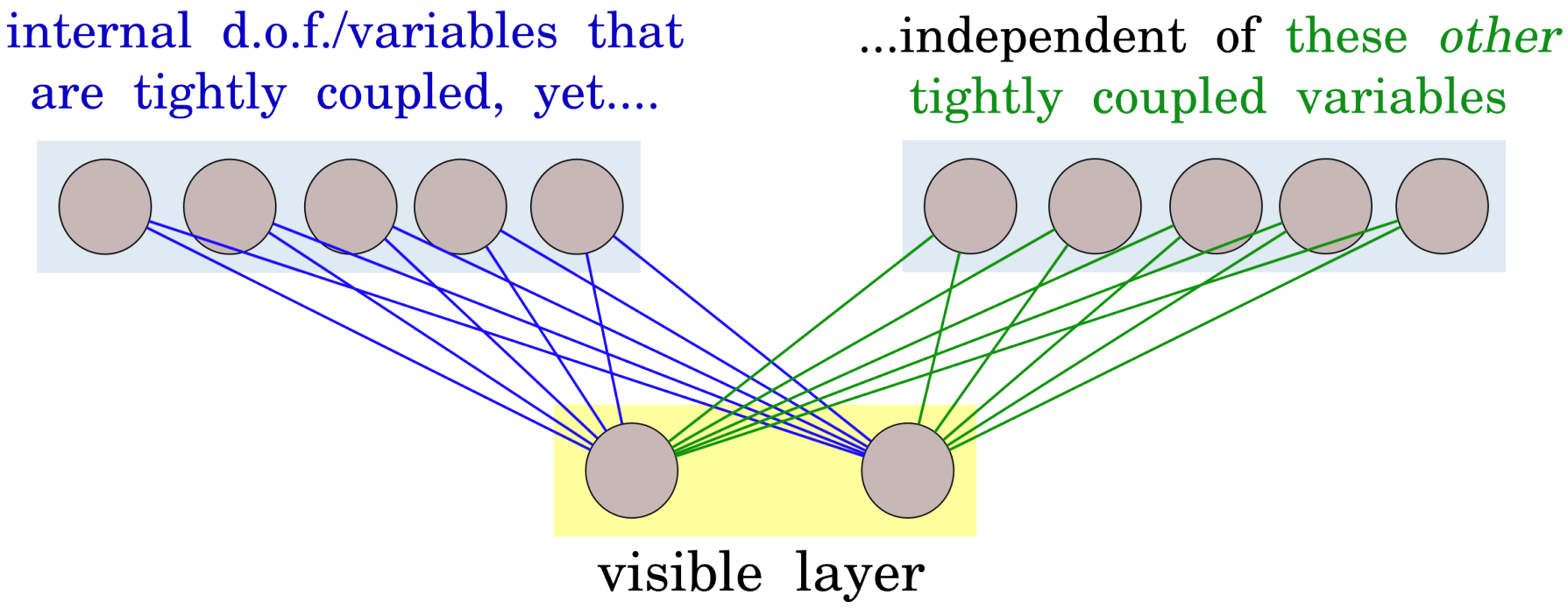
ie. unlike the Sigmoidal Belief Net, in an RBM the prior $P(\mathbf{h})$ *doesn't factor*.

So is not a model of multiple causes.

It's a single (albeit complex) cause.

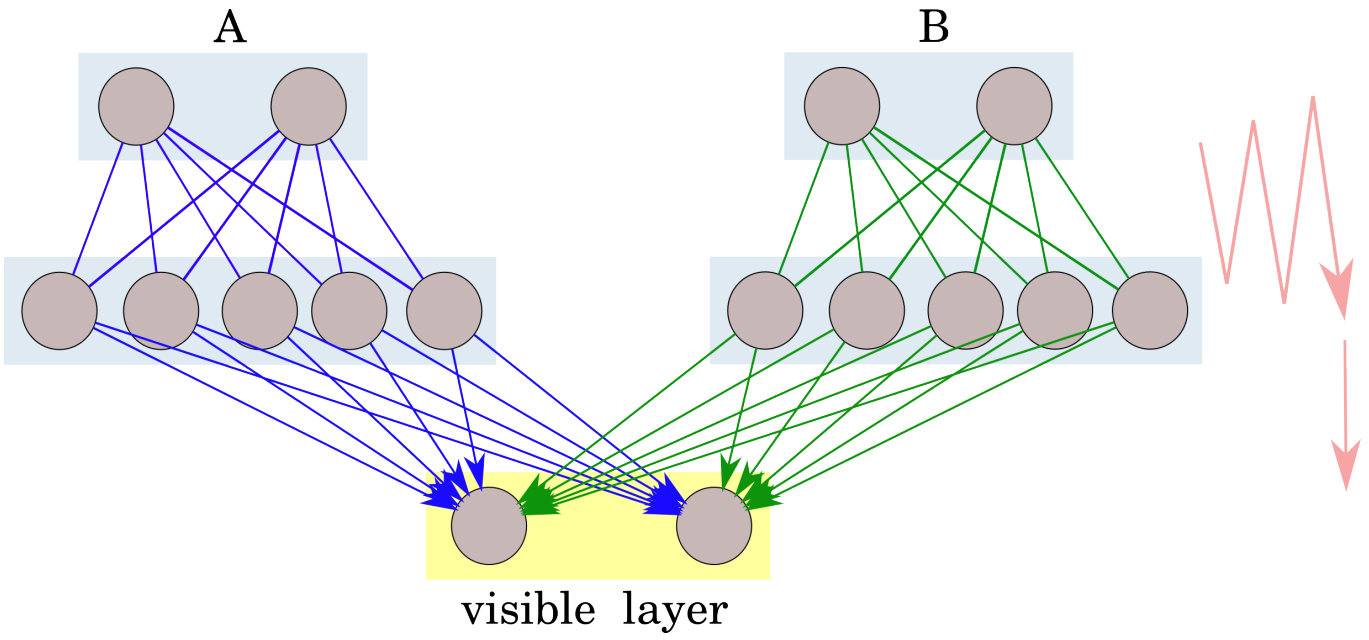
(cf. Sigmoidal: multiple (albeit simple) causes).

a “theory of things”



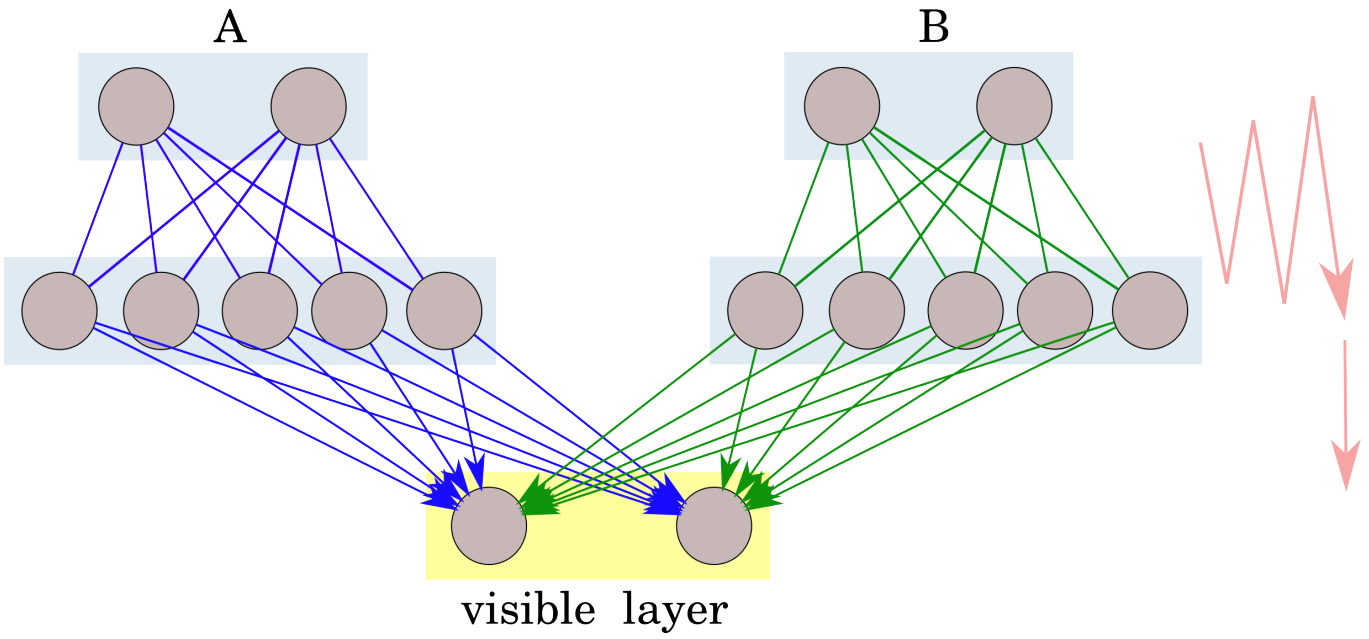
How to do it, and keep it learnable?

The ORBM



- the top layers are independed RBMs
- the bottom layer is a Sigmoid belief net, with input from both RBMs
- *roughly*, the visible layer computes $\approx$ logical OR (ish)

The ORBM



- the top layers are independed RBMs
- the bottom layer is a Sigmoid belief net, with input from both RBMs
- *roughly*, the visible layer computes $\approx$ logical OR (ish)

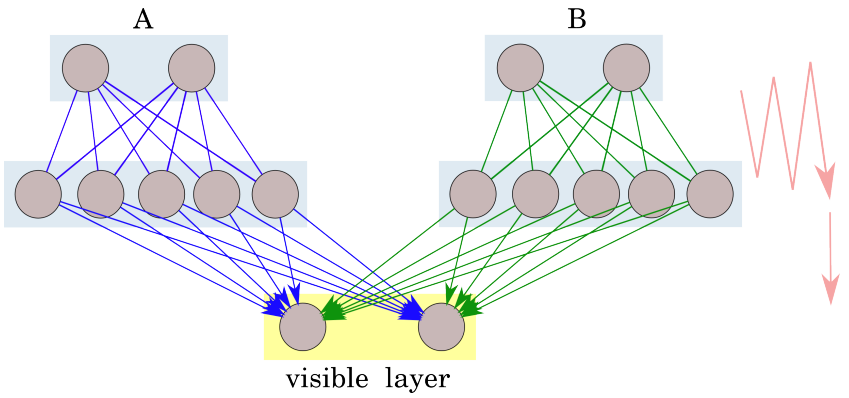
Sampling from generative model is easy. But what we really need is samples from $P(\mathbf{h}^A, \mathbf{h}^B \mid \mathbf{v})$ - how about them?

Gibbs in the orbium

Regular RBM:

$$P(h_j = 1|\mathbf{v}, \mathbf{h}_{\setminus j}) = \sigma(\psi_j)$$

$$\psi_j = \sum_i W_{ji} v_i$$



ORBM is the same but with

$$\psi_j = \sum_i W_{ji} v_i + C_{ji}$$

with $C_{ji} = \log\sigma(\ldots) + \log\sigma(\ldots) - \log\sigma(\ldots) - \log\sigma(\ldots)$

The $\log\sigma(\ldots)$ are $\approx$ “hinges”

(optional details)

The Gibbs Sampler's Bernoulli probability for the j-th hidden unit being 1 comes out to be:

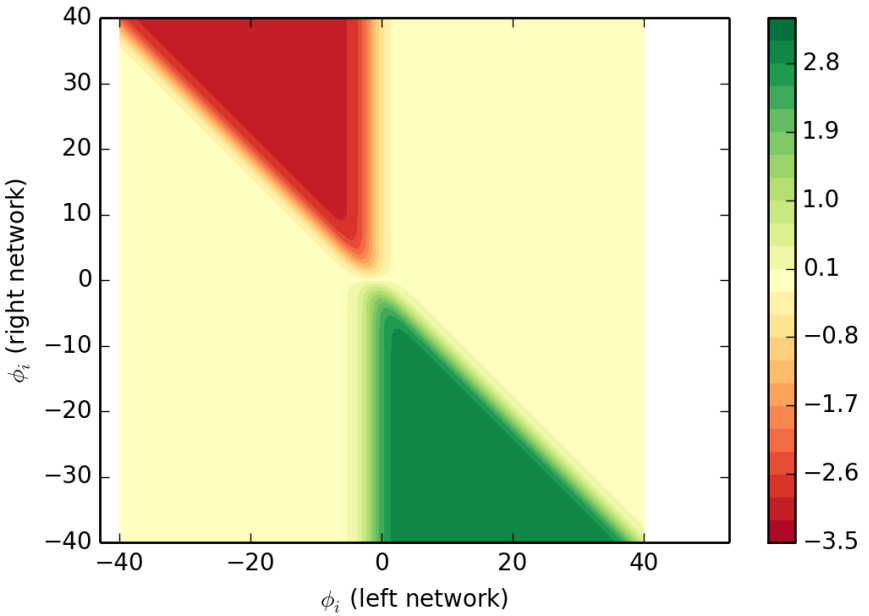
$$p_j = \sigma(\psi_j)$$

$$\psi_j = \sum_i W_{ji} v_i + C_{ji}$$

$$C_{ji} = \log\left[\frac{\sigma(\phi_i^0)}{\sigma(\phi_i^0 + W_{ji})} \cdot \frac{\sigma(\phi_i^0 + W_{ji} + \epsilon_i)}{\sigma(\phi_i^0 + \epsilon_i)}\right]$$

$$= \log\sigma(\phi_i^0) + \log\sigma(\phi_i^0 + W_{ji} + \epsilon_i) - \log\sigma(\phi_i^0 + W_{ji}) - \log\sigma(\phi_i^0 +$$

when the correction kicks in

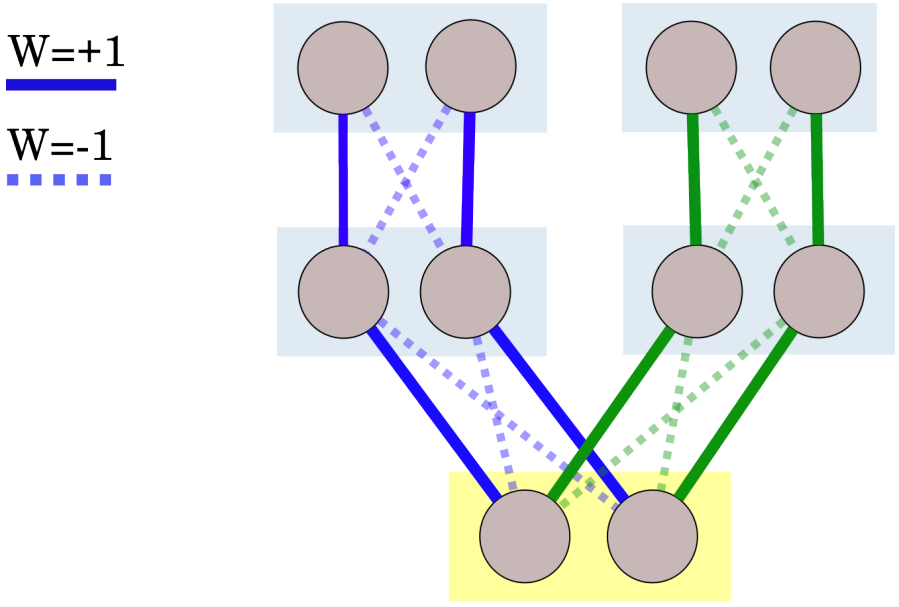


$$\implies \quad \psi_j \approx \sum_i W_{ji} \underbrace{\left(v_i + \sigma_i^A - \sigma_i^{AB}\right)}_{\text{“modified visibles”}}$$

Implements “explaining away”

NB: unlike in a simple RBM, sampling $P(\mathbf{h} \mid \mathbf{v})$ requires MCMC.
How bad?

demo on 2 bits

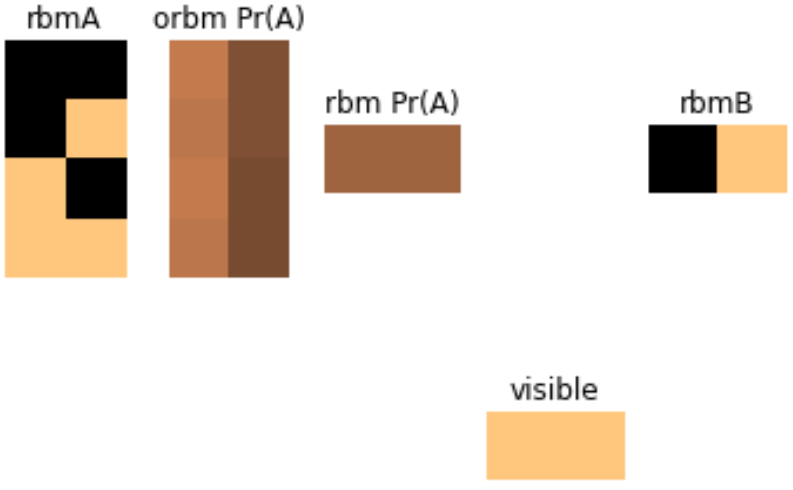


- two identical RBMS
- both “like” making visible patterns `01` or `10`, but not `00` or `11`
- given `11`, such RBMs will be hopelessly confused...
- the ORBM should “segment” the “image” `11`: each RBM should grab just one of the bits.

demo on 2 bits

And that's what happens...

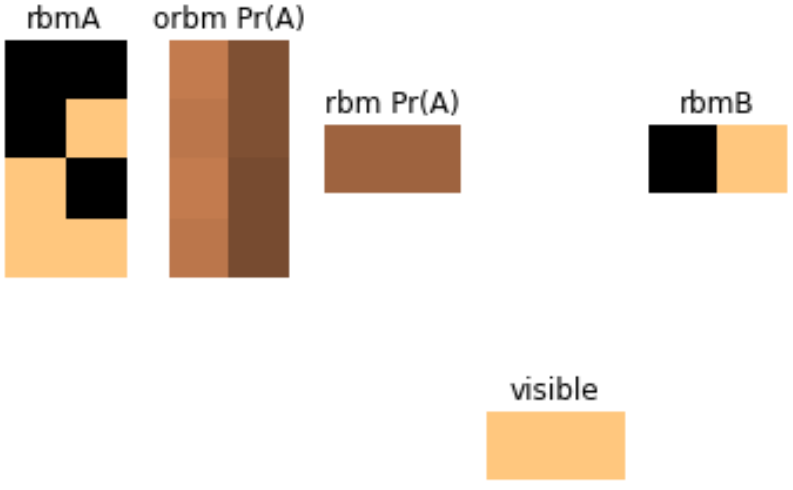
Using the exact calculation:



demo on 2 bits

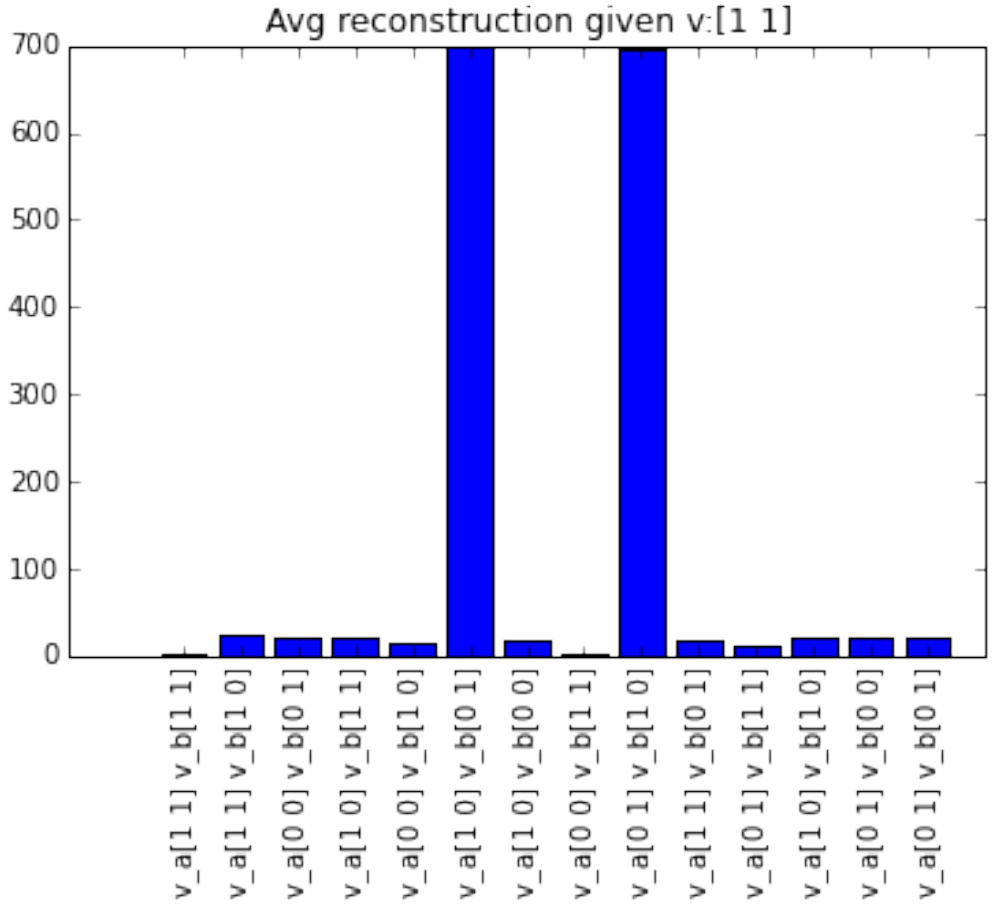
And that's what happens...

Using the approximation:

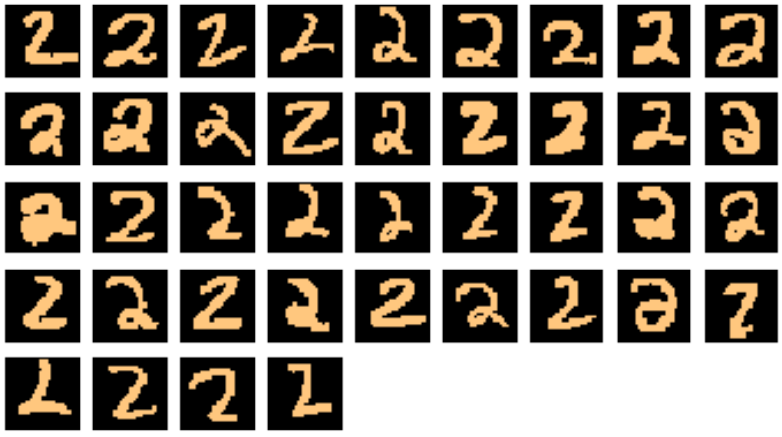


demo on 2 bits

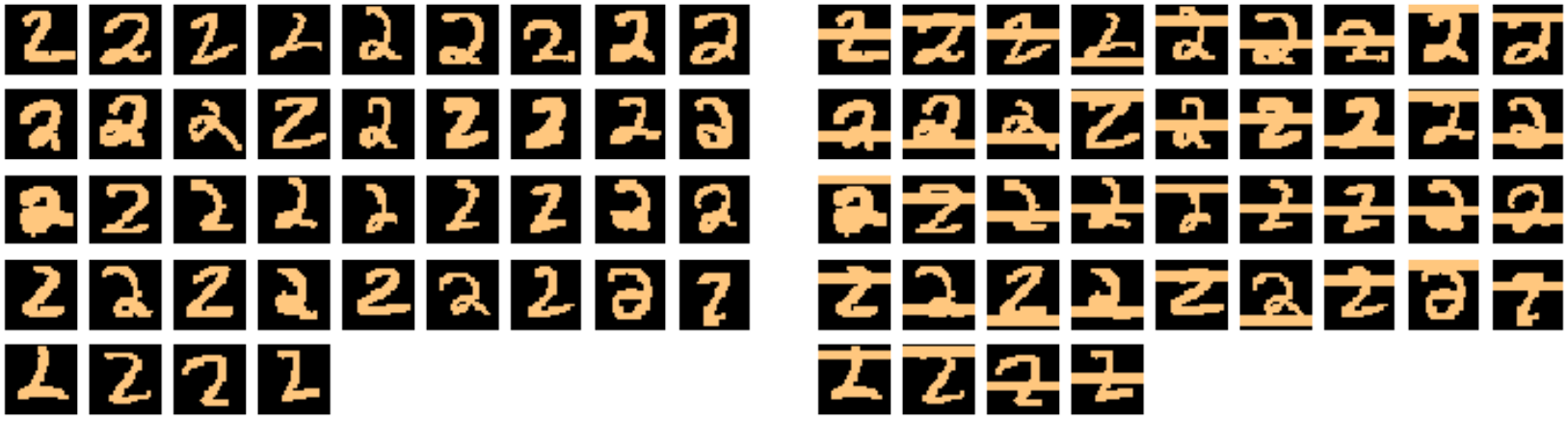
And that's what happens...



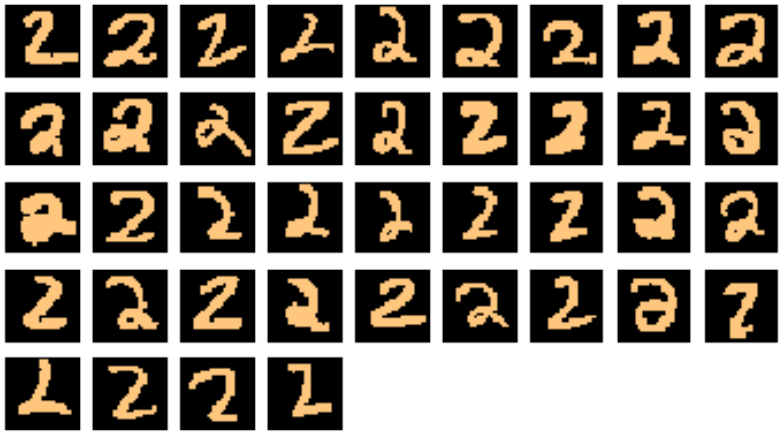
demo on 400 bits

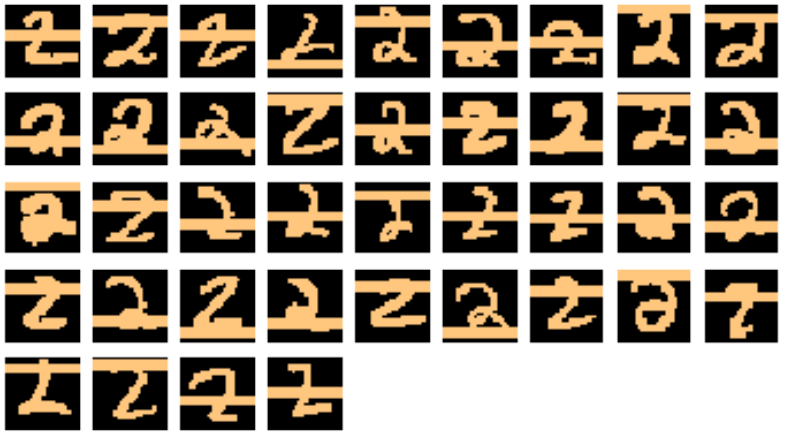


demo on 400 bits

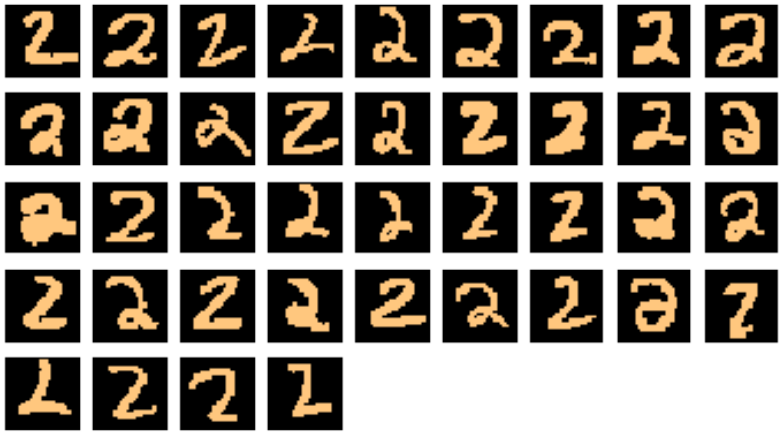


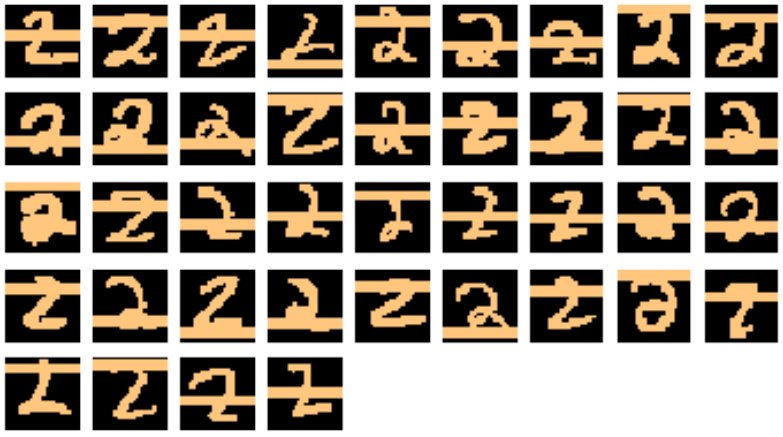
demo on 400 bits





demo on 400 bits





Lots to do...

Next steps:

- convincing large-scale demonstrations

Lots to do...

Next steps:

- convincing large-scale demonstrations
- the learning algorithm $\longrightarrow$ blind source separation

Lots to do...

Next steps:

- convincing large-scale demonstrations
- the learning algorithm $\longrightarrow$ blind source separation
- will it learn features that are a more useful foundation for a deep learner...?

Lots to do...

Next steps:

- convincing large-scale demonstrations
- the learning algorithm $\longrightarrow$ blind source separation
- will it learn features that are a more useful foundation for a deep learner...?

thanks

learning algorithm

WAKE PHASE

use our Gibbs chain to get samples from h for each $v \in \mathcal{D}$, and do

$$\Delta W_{ji}^{A} \quad \propto \quad \underbrace{\sigma(\phi_i^A) h_j^A}_{\text{mean field Hebbian}} \quad + \quad \underbrace{(v_i - \sigma(\phi_i^{\mathsf{AB}})) h_j^A}_{\text{Perceptron learning rule}}$$

and similarly for the other RBM. Note that the ϕ's are not the same in the two terms: the first only sums input from the A side, but the second sums from both hidden layers.

Another way to say the same thing would be

$$\Delta W_{ji}^{A} \quad \propto \quad \left[v_i + \sigma(\phi_i^A) - \sigma(\phi_i^{\mathsf{AB}}) \right] h_j^A$$

which is like a Hebbian update but with a modified visible activation, in effect.

learning algorithm

SLEEP PHASE

use ”regular” Gibbs sampling *in each model separately* to sample independently from the two RBMs, and do

$$\Delta W^A_{ml} \quad \propto \quad -v_l \; h^A_m$$

and similarly for the other RBM.

autoencoding