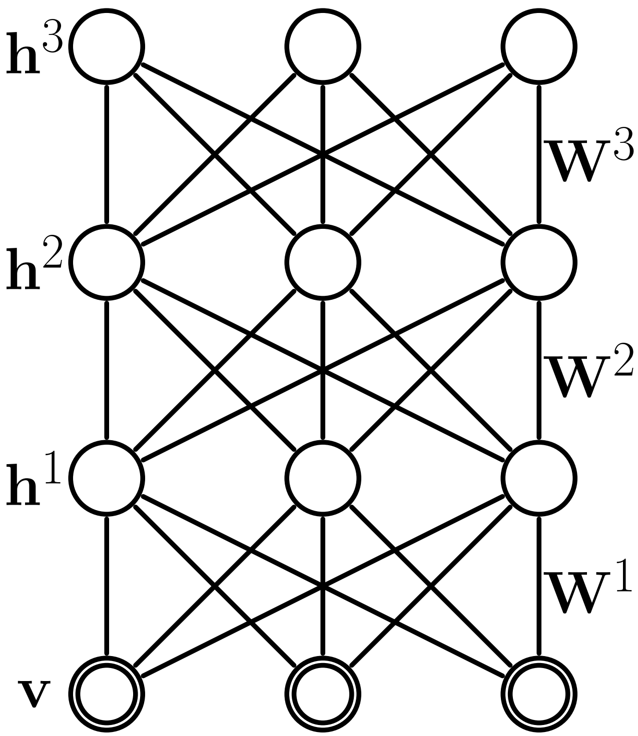
Deep Boltzmann Machines

Ruslan Salakhutdinov

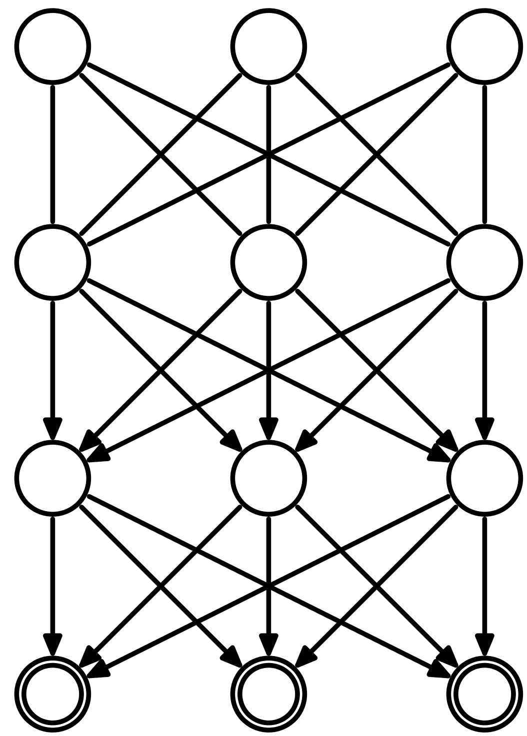
Work with **Geoffrey Hinton**

Dept. of Computer Science, University of Toronto

DBM's vs. DBN's



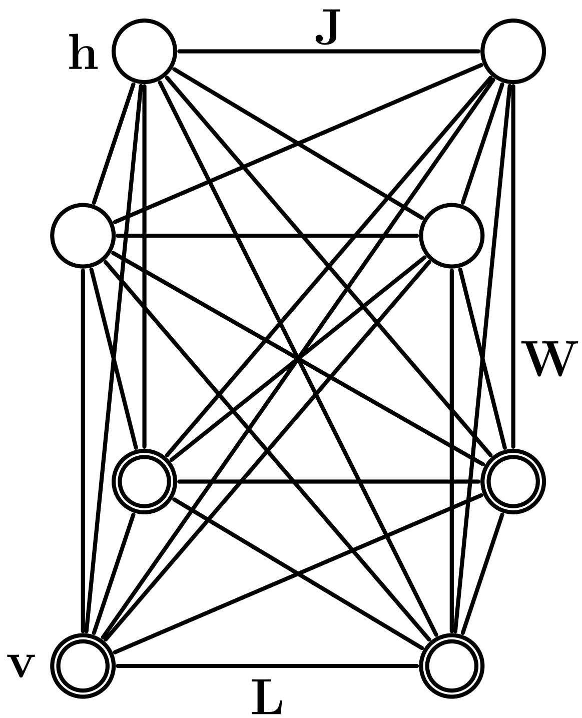
Deep Boltzmann Machine



Deep Belief Network

General Boltzmann Machines

$$P(\mathbf{v},\mathbf{h};\theta) = \frac{1}{\mathcal{Z}(\theta)} \exp\left[\mathbf{v}^\top W \mathbf{h} + \tfrac{1}{2}\mathbf{v}^\top L \mathbf{v} + \tfrac{1}{2}\mathbf{h}^\top J \mathbf{h}\right].$$



$$P(\mathbf{v};\theta) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h};\theta).$$

Set of visible $\mathbf{v}$ and hidden $\mathbf{h}$ binary stochastic units.

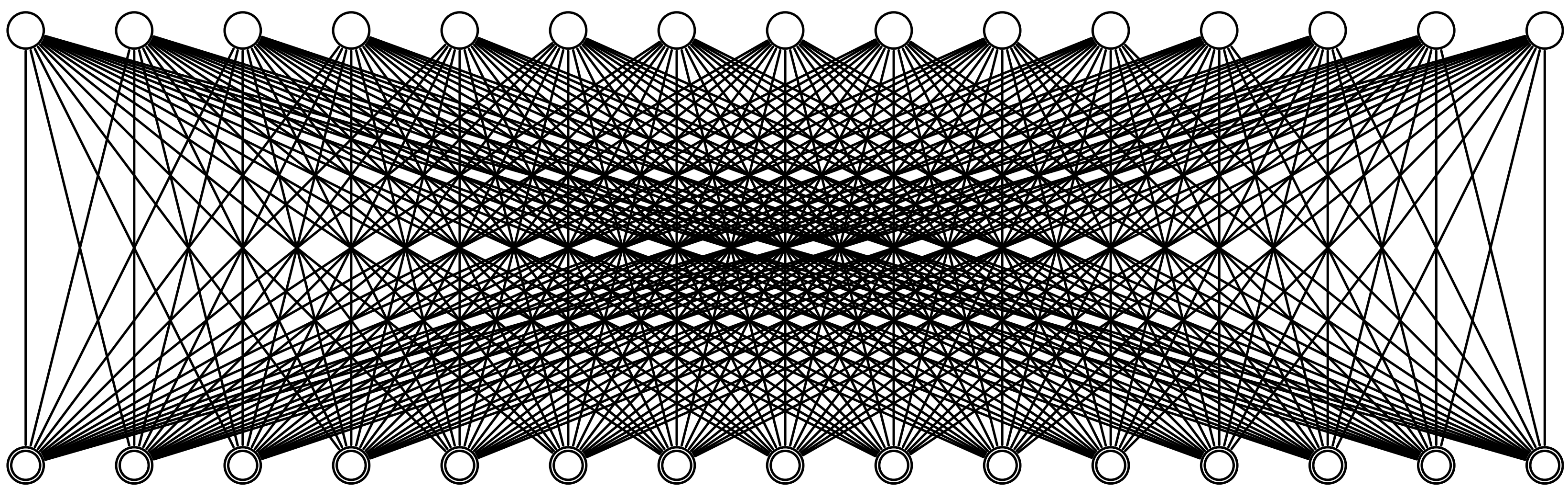
$\theta = \{W, L, J\}$ are model parameters.

Inference and maximum likelihood learning are hard.

This talk: Learning θ.

Restricted Boltzmann Machines

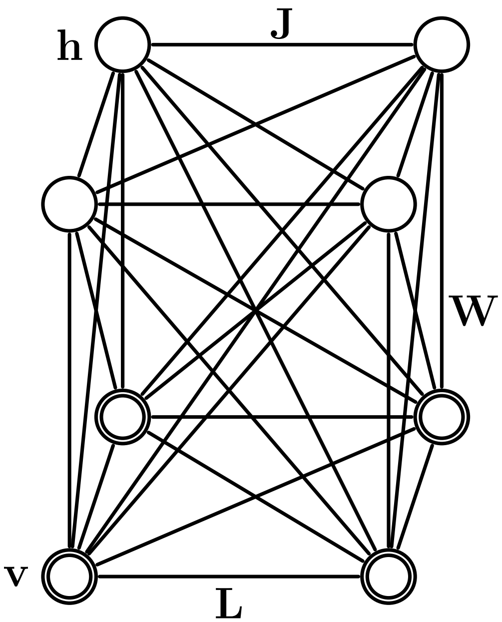
$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp\left[\mathbf{v}^\top W \mathbf{h}\right].$$



Computing $P(\mathbf{h}|\mathbf{v})$ is easy. Maximum likelihood learning is hard.

General BM's: Learning

$$P_{\text{model}}(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}} \exp\left[\mathbf{v}^\top W \mathbf{h} + \tfrac{1}{2}\mathbf{v}^\top L \mathbf{v} + \tfrac{1}{2}\mathbf{h}^\top J \mathbf{h}\right].$$



Maximum Likelihood Learning:

$$\frac{\partial \ln P(\mathbf{v})}{\partial W} = \mathrm{E}_{\mathrm{P}_{\text{data}}}[\mathbf{v}\mathbf{h}^\top] - \mathrm{E}_{\mathrm{P}_{\text{model}}}[\mathbf{v}\mathbf{h}^\top].$$

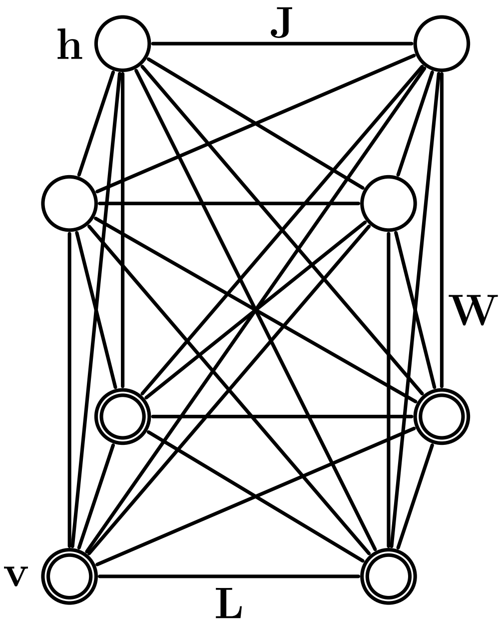
$$P_{\text{data}}(\mathbf{h}, \mathbf{v}) = P(\mathbf{h}|\mathbf{v}) P_{\text{data}}(\mathbf{v}).$$

Previous Approach: For each iteration of learning:

- A separate Markov chain is run for every data point to approximate $\mathrm{E}_{P_{\text{data}}}[\cdot]$.
- An additional chain is run to approximate $\mathrm{E}_{P_{\text{model}}}[\cdot]$.

General BM's: Learning

$$P_{\mathrm{model}}(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}} \exp\left[\mathbf{v}^\top W \mathbf{h} + \tfrac{1}{2}\mathbf{v}^\top L \mathbf{v} + \tfrac{1}{2}\mathbf{h}^\top J \mathbf{h}\right].$$



Maximum Likelihood Learning:

$$\frac{\partial \ln P(\mathbf{v})}{\partial W} = \mathrm{E}_{\mathrm{P}_{\mathrm{data}}}[\mathbf{v}\mathbf{h}^\top] - \mathrm{E}_{\mathrm{P}_{\mathrm{model}}}[\mathbf{v}\mathbf{h}^\top].$$

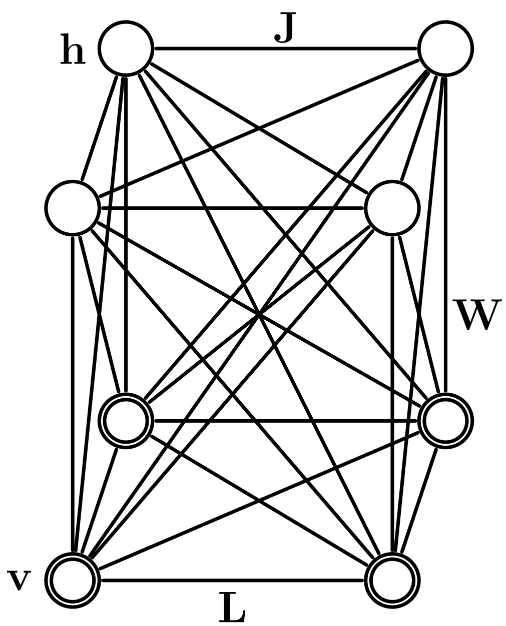
$$P_{\mathrm{data}}(\mathbf{h},\mathbf{v}) = P(\mathbf{h}|\mathbf{v}) P_{\mathrm{data}}(\mathbf{v}).$$

Key Idea:

- Variational Inference: Approximate $\mathrm{E}_{\mathrm{P}_{\mathrm{data}}}[\cdot]$.
- Persistent MCMC: Approximate $\mathrm{E}_{\mathrm{P}_{\mathrm{model}}}[\cdot]$.

General BM's: Learning

$$P_{\text{model}}(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}} \exp\left[\mathbf{v}^\top W \mathbf{h} + \tfrac{1}{2}\mathbf{v}^\top L \mathbf{v} + \tfrac{1}{2}\mathbf{h}^\top J \mathbf{h}\right].$$



Maximum Likelihood Learning:

$$\frac{\partial \ln P(\mathbf{v})}{\partial W} = \mathrm{E}_{\mathrm{P}_{\text{data}}}[\mathbf{v}\mathbf{h}^\top] - \mathrm{E}_{\mathrm{P}_{\text{model}}}[\mathbf{v}\mathbf{h}^\top].$$

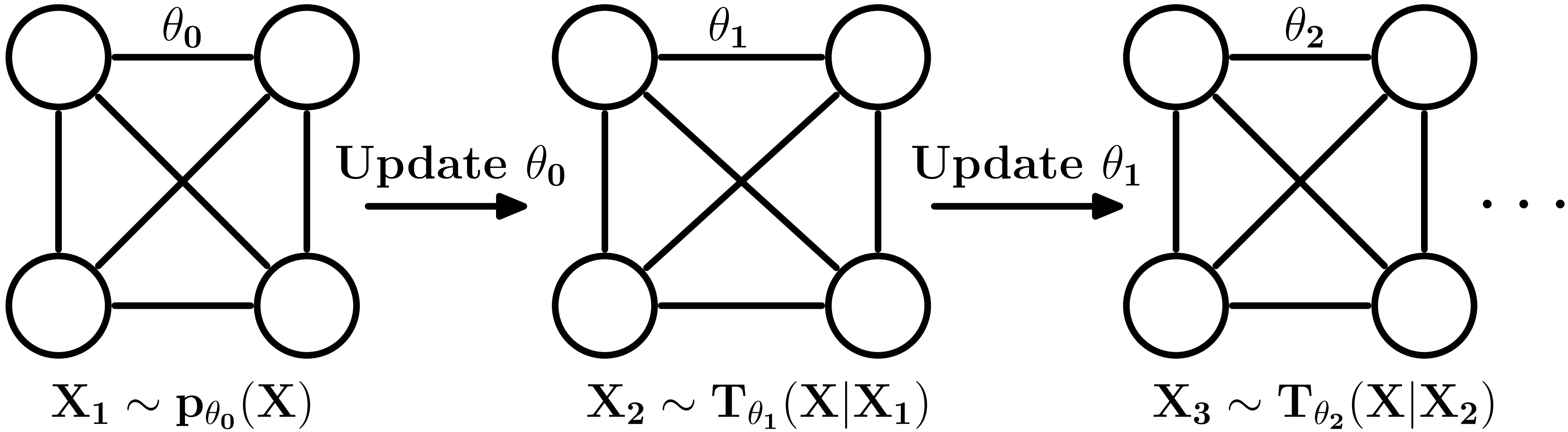
$$P_{\text{data}}(\mathbf{h}, \mathbf{v}) = P(\mathbf{h}|\mathbf{v}) P_{\text{data}}(\mathbf{v}).$$

Key Idea:

- Variational Inference: Approximate $\mathrm{E}_{\mathrm{P}_{\text{data}}}[\cdot]$.
- Persistent MCMC: Approximate $\underbrace{\mathrm{E}_{\mathrm{P}_{\text{model}}}[\cdot]}_{\text{most difficult}}$.

Stochastic Approximation

Stochastic approximation procedure: estimate $\mathrm{E}_{\mathrm{P}_{\mathrm{model}}}[\cdot]$.



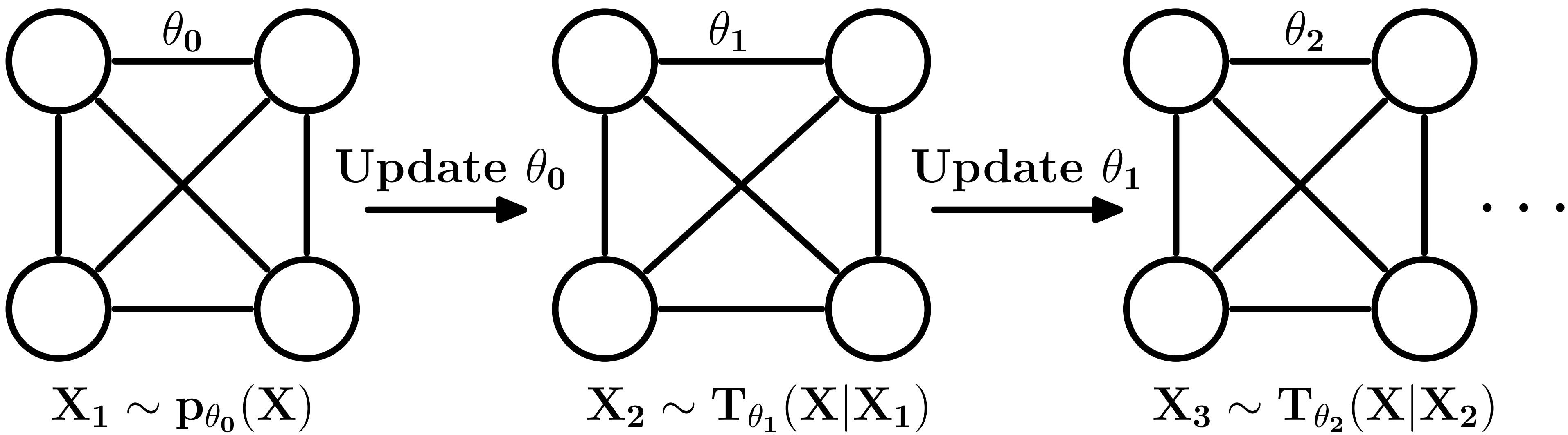
Update X_t and θ_t sequentially:

- Sample $X_{t+1} \sim T_{\theta_t}(X|X_t)$ that leaves p_{θ_t} invariant.
- Update θ_t by replacing $\mathrm{E}_{P_{\mathrm{model}}}[\cdot]$ by the expectation w.r.t. X_{t+1}.

Almost sure convergence guarantees.

Stochastic Approximation

Stochastic approximation procedure: estimate $\mathrm{E}_{\mathrm{P}_{\mathrm{model}}}[\cdot]$.



Let $S(\theta) = \frac{\partial \log p(\mathbf{x};\theta)}{\partial \theta}$, then:

$$\underbrace{\theta_t = \theta_{t-1} + \alpha_t S(\theta_t)}_{\text{ODE } \dot{\theta} = S(\theta)} + \alpha_t \underbrace{\left(\frac{1}{M} \sum_m \tilde{S}(X_t^m, \theta_t) - S(\theta_t) \right)}_{\text{noise term } \epsilon_t}.$$

Learning BM's

$$\log P(\mathbf{v};\theta) \geq$$

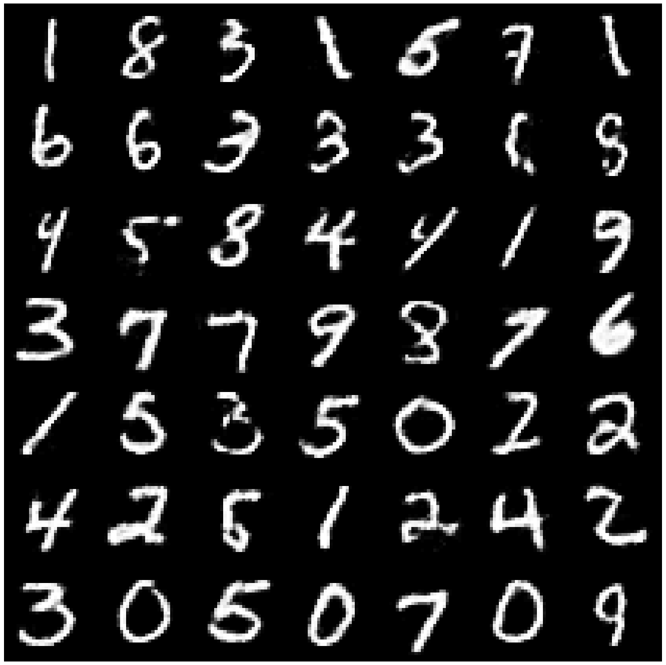
$$\sum_{\mathbf{h}} Q(\mathbf{h}|\mathbf{v}) \underbrace{\log P^*(\mathbf{v},\mathbf{h};\theta)}_{\mathbf{v}^\top W\mathbf{h} + \frac{1}{2}\mathbf{v}^\top L\mathbf{v} + \frac{1}{2}\mathbf{h}^\top J\mathbf{h}} - \log \mathcal{Z}(\theta) + \mathcal{H}[Q(\mathbf{h}|\mathbf{v})].$$

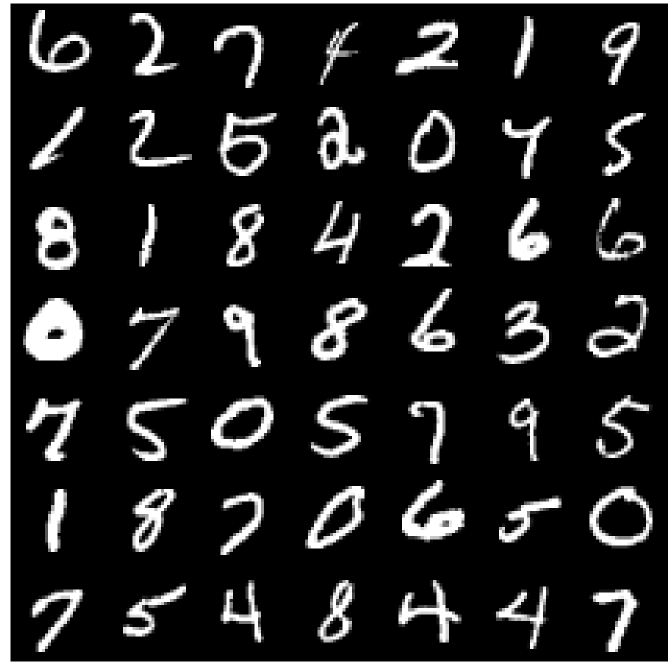
For each iteration of learning:

1. **Variational Inference:** Maximize the lower bound w.r.t. variational parameters for fixed θ.

2. **MCMC:** Apply stochastic approximation procedure to update the model parameters θ.

MNIST



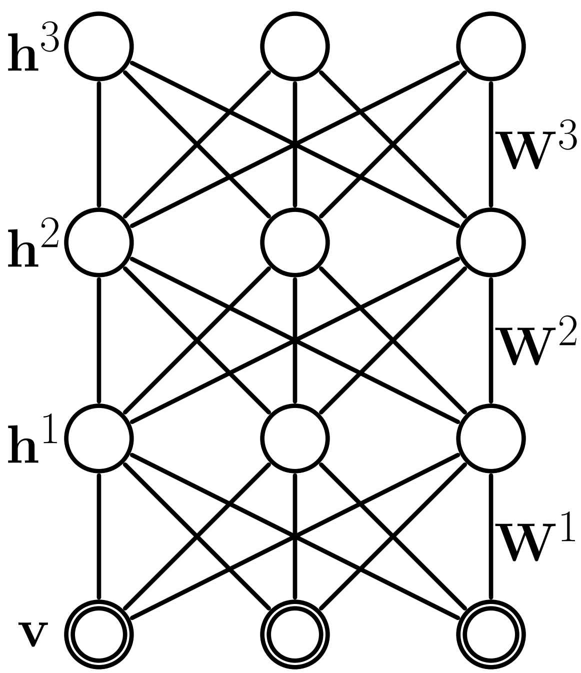


500 hidden and 784 visible units (820,000 parameters).

Samples were generated by running the Gibbs sampler for 100,000 steps.

Deep Boltzmann Machines

$$P(\mathbf{v}) = \sum_{\mathbf{h}^1,\mathbf{h}^2,\mathbf{h}^3} \frac{1}{\mathcal{Z}} \exp\left[\mathbf{v}^\top W^1 \mathbf{h} + {\mathbf{h}^1}^\top W^2 \mathbf{h}^2 + {\mathbf{h}^2}^\top W^3 \mathbf{h}^3\right].$$



Complex representations.

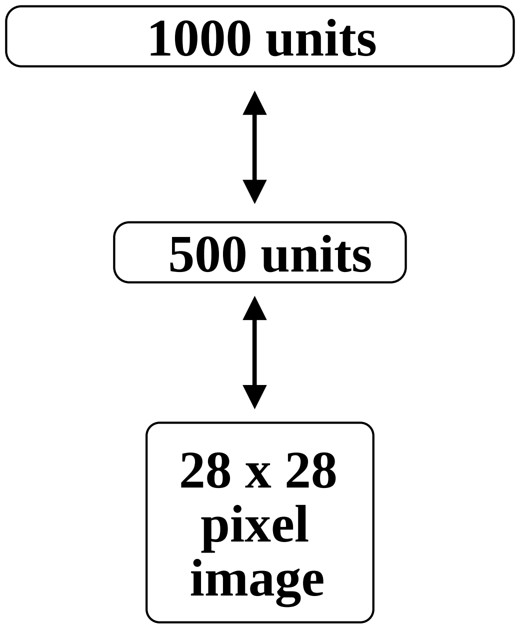
Fast greedy initialization.

Bottom-up + Top-down.

High-level representations are built from unlabeled inputs.

Labeled data is used to only slightly fine-tune the model.

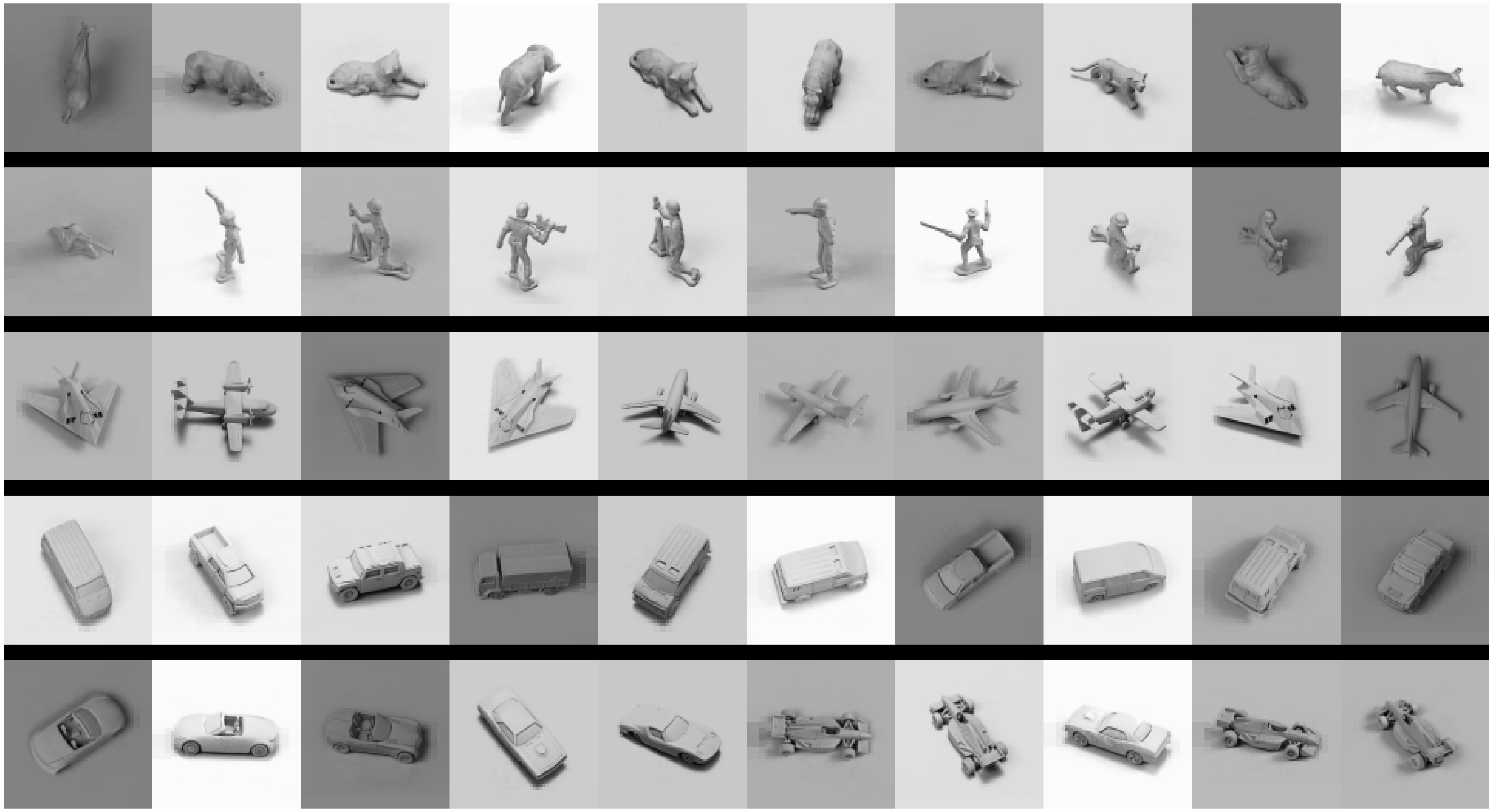
MNIST



0.9 million parameters,
60,000 training and 10,000 test examples.

Test error: 0.95%.
DBN's get 1.2%, SVM's get 1.4%, backprop gets 1.6%.

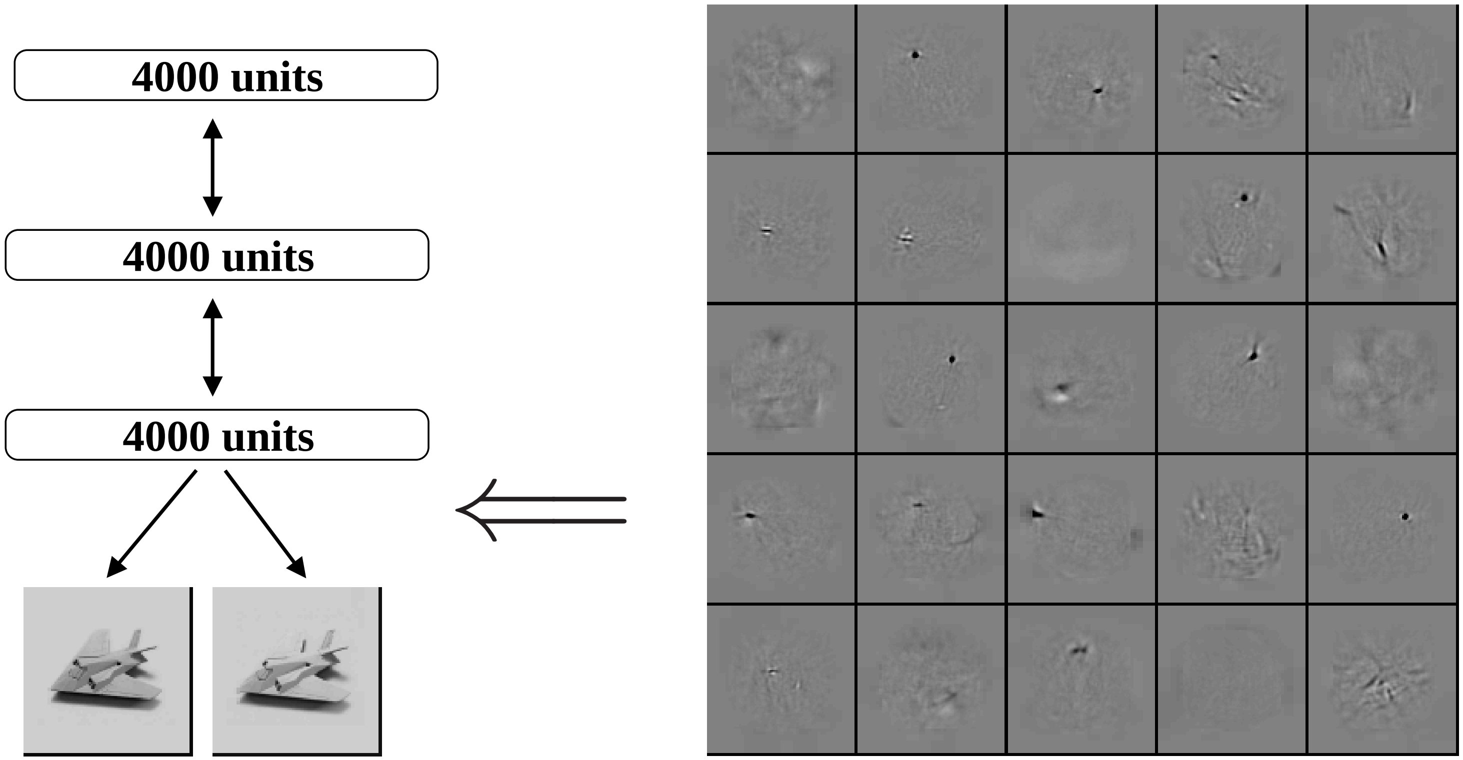
NORB data



5 object categories, 5 different objects within each category, 6 lighting conditions, 9 elevations, 18 azimuth.

24,300 training and 24,300 test cases.

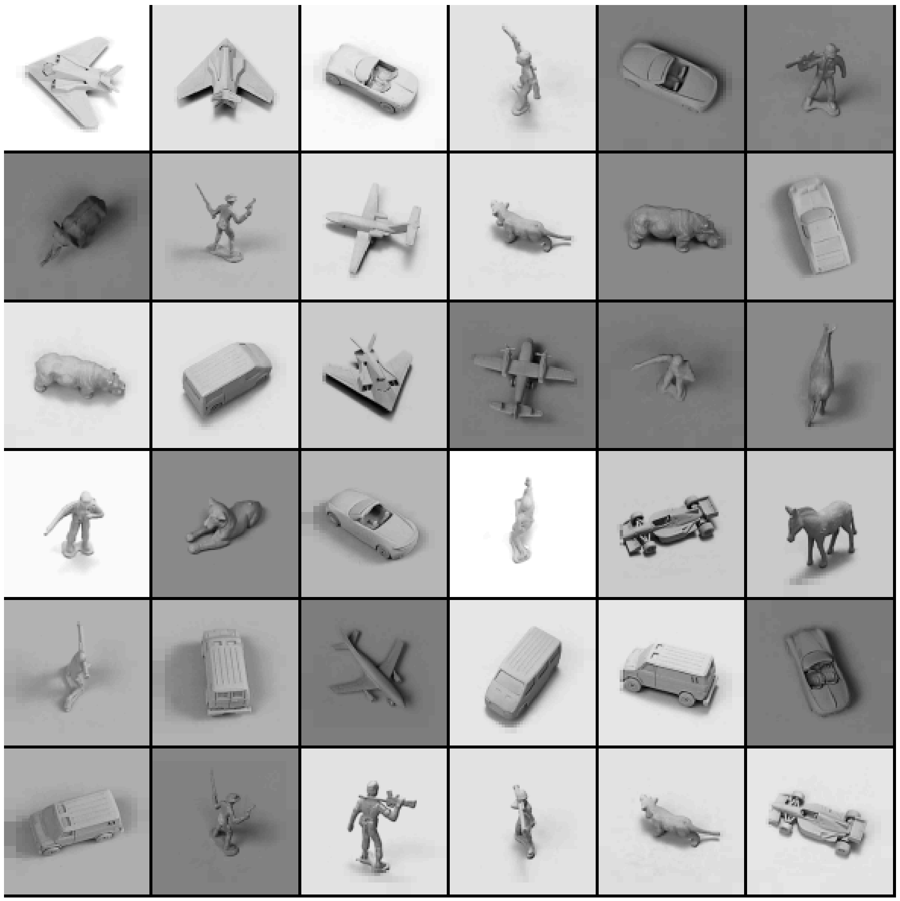
Deep Boltzmann Machines

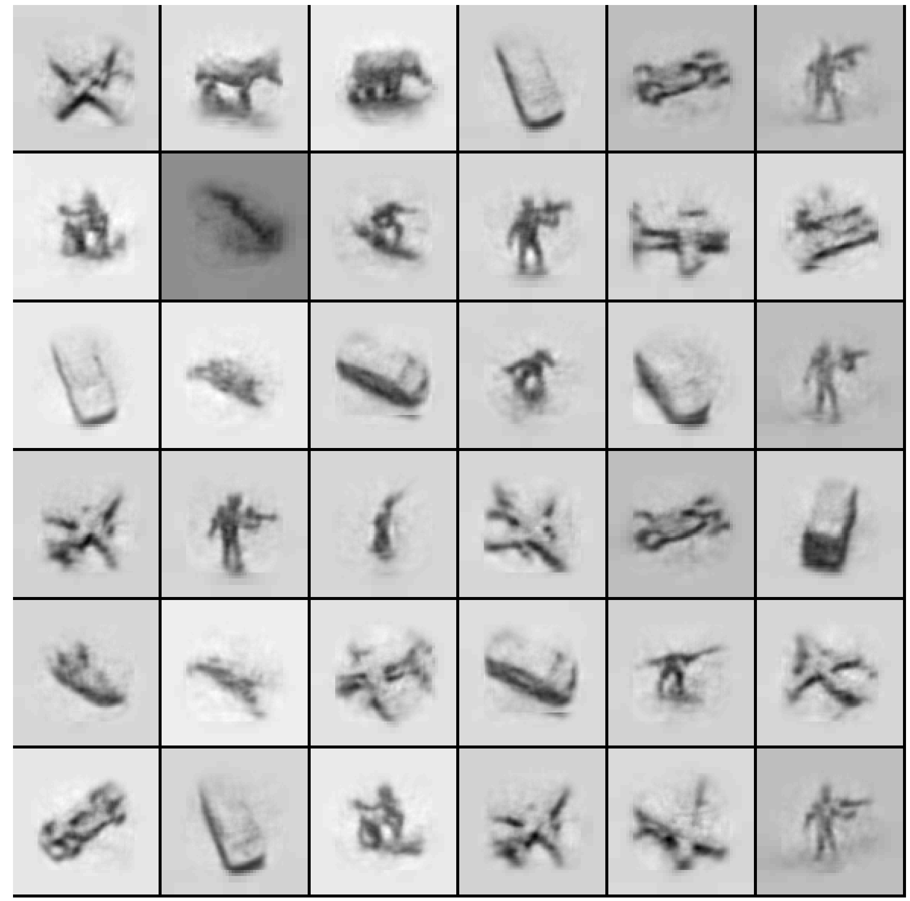


Stereo pair

About 68 million parameters.

Model Samples





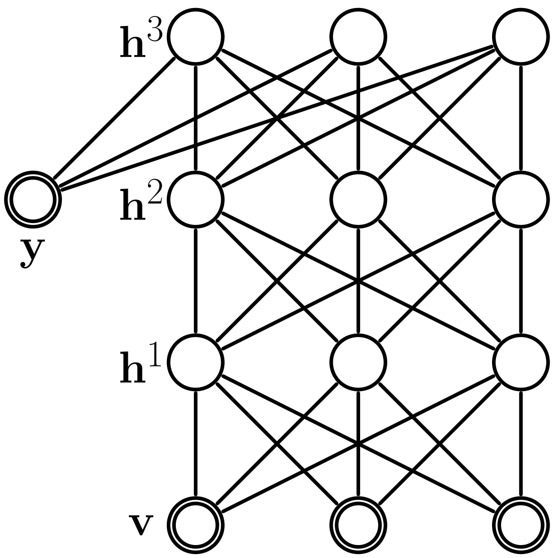
Discriminative fine-tuning: test error of 7.2%.

SVM's get 11.6%, logistic regression gets 22.5%.

Semi-supervised Learning

Variational

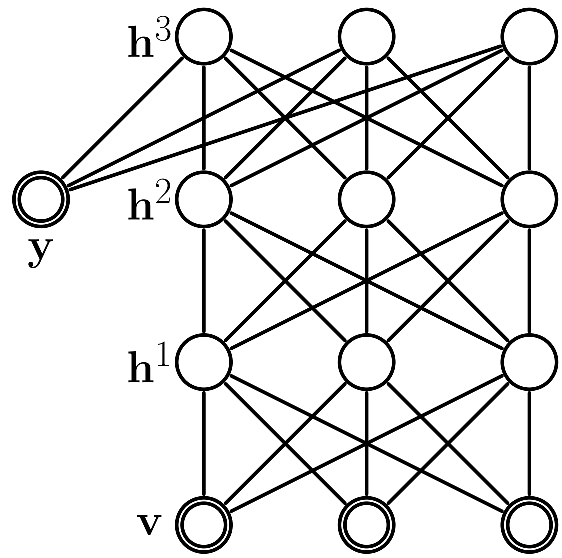
Find approx. posterior $Q(\mathbf{h}|\mathbf{v})$



$\mathbf{E}_{P_{\mathrm{data}}}[\cdot]$

MCMC

Sample binary state $\{\mathbf{v}, \mathbf{h}\ \mathbf{y}\}$



$\mathbf{E}_{P_{\mathrm{model}}}[\cdot]$

If $\mathbf{y}$ is missing: use variational inference to effectively fill in the missing label.

Thank you.