10707
Deep Learning

Russ Salakhutdinov

Machine Learning Department
rsalakhu@cs.cmu.edu

Restricted Boltzmann Machines

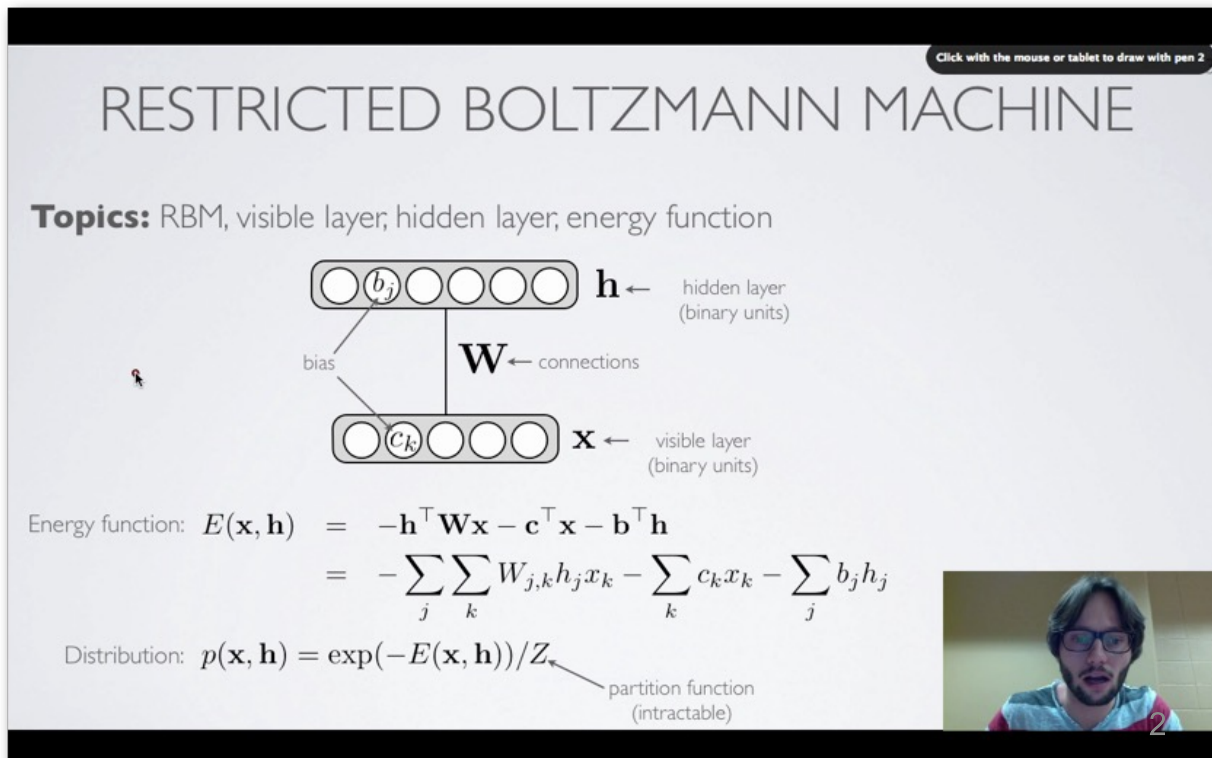
Neural Networks Online Course

- **Disclaimer**: Much of the material and slides for this lecture were borrowed from Hugo Larochelle's class on Neural Networks:

- Hugo's class covers many other topics: convolutional networks, neural language model, Boltzmann machines, autoencoders, sparse coding, etc.

- We will use his material for some of the other lectures.

http://info.usherbrooke.ca/hlarochelle/neural_networks

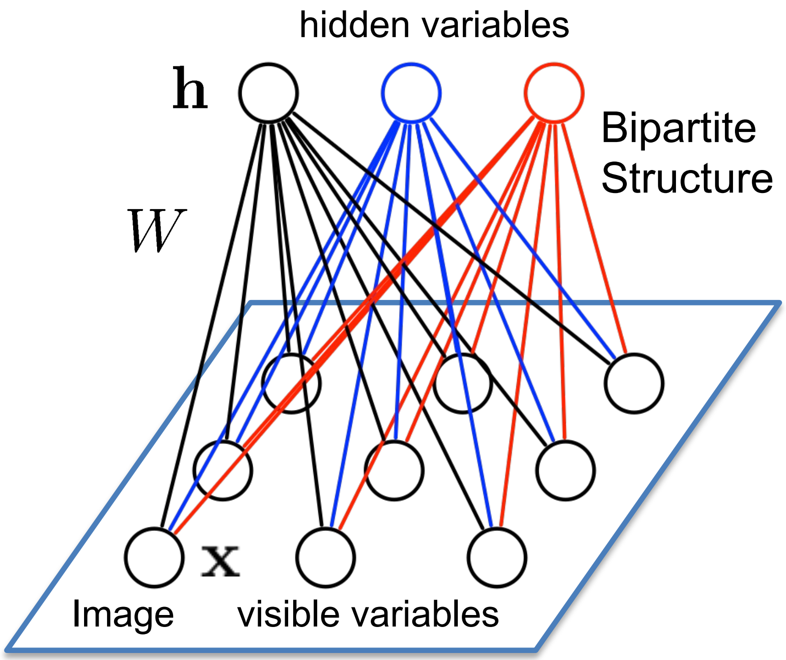


Unsupervised Learning

- Unsupervised learning: we only use the inputs $\mathbf{x}^{(t)}$ for learning
  - automatically extract meaningful features for your data
  - leverage the availability of unlabeled data
  - add a data-dependent regularizer to training ($-\log p(\mathbf{x}^{(t)})$)

- We will consider 3 models for unsupervised learning that will form the basic building blocks for deeper models:
  - Restricted Boltzmann Machines
  - Autoencoders
  - Sparse coding models

Restricted Boltzmann Machines



- Undirected bipartite graphical model
- Stochastic binary visible variables:

$$\mathbf{x} \in \{\mathbf{0}, \mathbf{1}\}^{\mathbf{D}}$$

- Stochastic binary hidden variables:

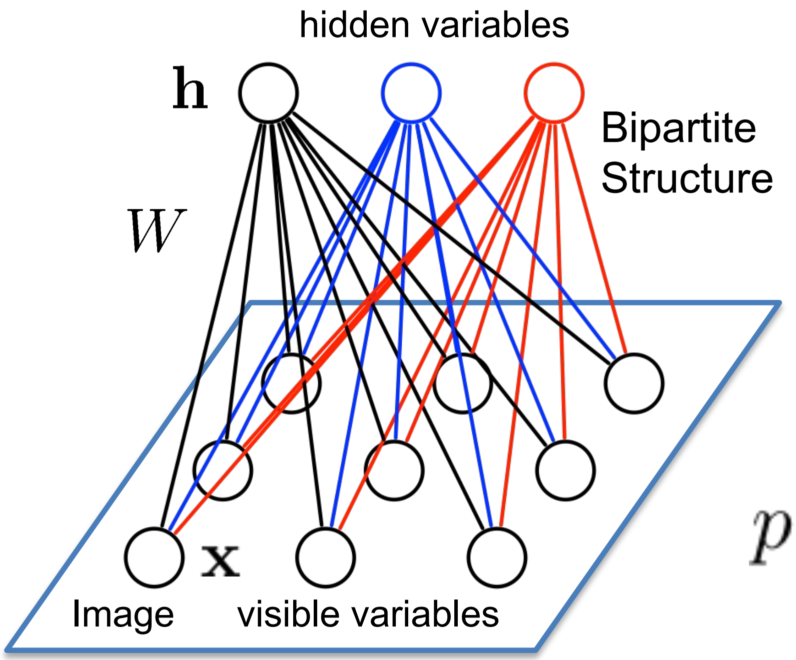
$$\mathbf{h} \in \{0, 1\}^F$$

- The energy of the joint configuration:

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}) &= -\mathbf{h}^\top \mathbf{W}\mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h} \\ &= -\sum_j \sum_k W_{j,k} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j \end{aligned}$$

Markov random fields, Boltzmann machines, log-linear models.

Restricted Boltzmann Machines



- Probability of the joint configuration is given by the Boltzmann distribution:

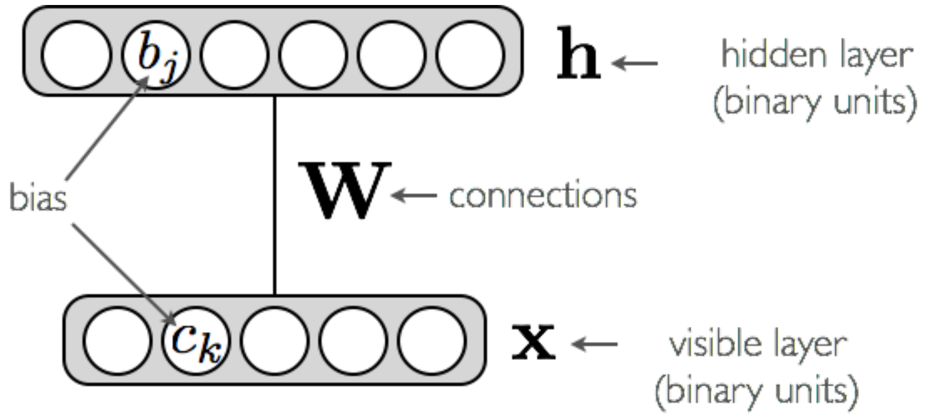
$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

Partition function (intractable)

$$Z = \sum_{\mathbf{x},\mathbf{h}} \exp\left(-E(\mathbf{x}, \mathbf{h})\right)$$

Markov random fields, Boltzmann machines, log-linear models.

Restricted Boltzmann Machines



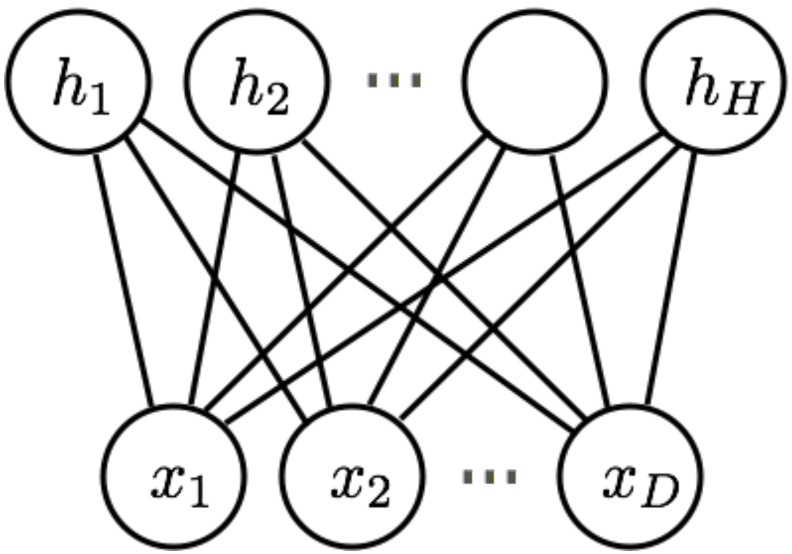
$$\begin{aligned} p(\mathbf{x}, \mathbf{h}) &= \exp(-E(\mathbf{x}, \mathbf{h}))/Z \\ &= \exp(\mathbf{h}^\top \mathbf{W} \mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h})/Z \\ &= \underbrace{\exp(\mathbf{h}^\top \mathbf{W} \mathbf{x}) \exp(\mathbf{c}^\top \mathbf{x}) \exp(\mathbf{b}^\top \mathbf{h})}_{\text{Factors}}/Z \end{aligned}$$

• The notation based on an energy function is simply an alternative to the representation as the product of factors

Restricted Boltzmann Machines

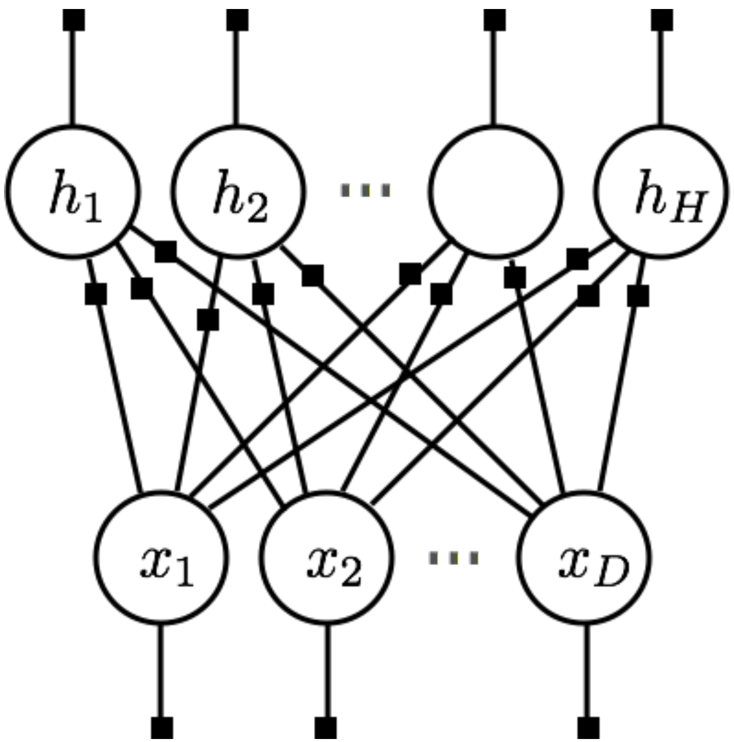
$$p(\mathbf{x},\mathbf{h}) = \frac{1}{Z}\overbrace{\prod_j \prod_k \exp(W_{j,k} h_j x_k)}^{\text{Pair-wise factors}}$$

$$\left.\begin{array}{l}\displaystyle\prod_k \exp(c_k x_k) \\ \displaystyle\prod_j \exp(b_j h_j)\end{array}\right\}\ \text{Unary factors}$$



- The scalar visualization is more informative of the structure within the vectors.

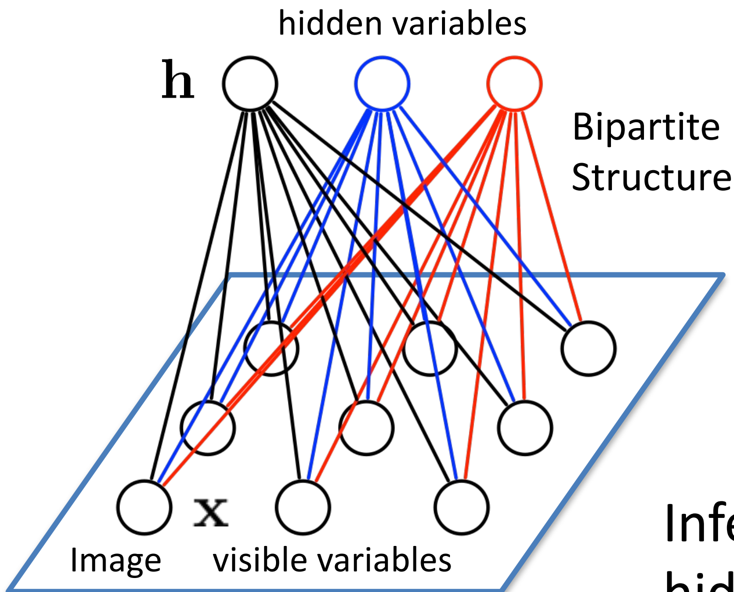
Factor Graph View



$$p(\mathbf{x}, \mathbf{h}) = \overbrace{\frac{1}{Z} \prod_j \prod_k \exp(W_{j,k} h_j x_k)}^{\text{Pair-wise factors}}$$

$$\left.\begin{array}{l} \displaystyle\prod_k \exp(c_k x_k) \\ \displaystyle\prod_j \exp(b_j h_j) \end{array}\right\} \text{Unary factors}$$

Inference



Restricted: No interaction between hidden variables

Inferring the distribution over the hidden variables is easy:

$$p(\mathbf{h}|\mathbf{x}) = \underbrace{\prod_j p(h_j|\mathbf{x})}_{\text{Factorizes: Easy to compute}}$$

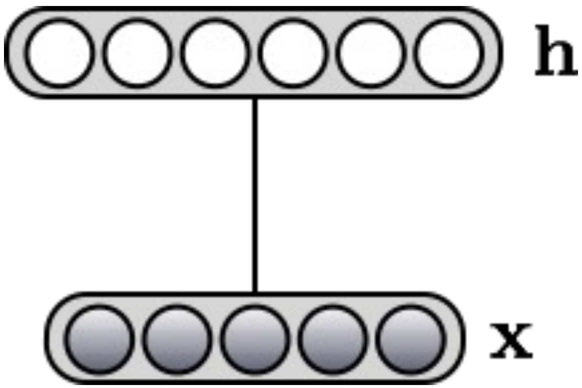
Similarly:

$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

Markov random fields, Boltzmann machines, log-linear models.

Inference

- Conditional Distributions:

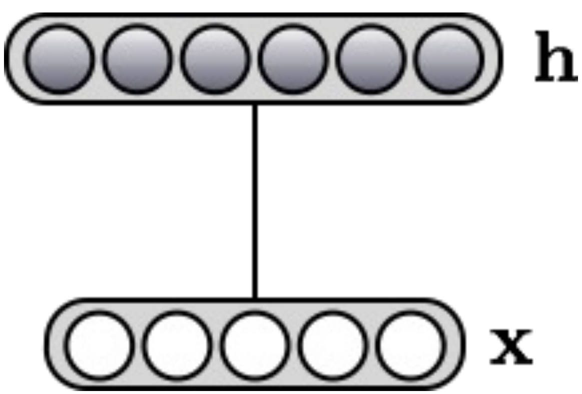


$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

$$p(h_j = 1|\mathbf{x}) = \frac{1}{1+\exp(-(b_j + \mathbf{W}_{j\cdot}\mathbf{x}))}$$

$$= \mathrm{sigm}(b_j + \mathbf{W}_{j\cdot}\mathbf{x})$$

j^{th} row of W



$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

$$p(x_k = 1|\mathbf{h}) = \frac{1}{1+\exp(-(c_k + \mathbf{h}^\top \mathbf{W}_{\cdot k}))}$$

$$= \mathrm{sigm}(c_k + \mathbf{h}^\top \mathbf{W}_{\cdot k})$$

k^{th} column of W

Local Markov Property

- In general, we have the following property:

$$
\begin{aligned}
p(z_i|z_1,\ldots,z_V) &= p(z_i|\mathrm{Ne}(z_i)) \\
&= \frac{p(z_i,\mathrm{Ne}(z_i))}{\sum_{z_i'} p(z_i',\mathrm{Ne}(z_i))} \\
&= \frac{\prod_{\substack{f \text{ involving } z_i \\ \text{and any } \mathrm{Ne}(z_i)}} \Psi_f(z_i,\mathrm{Ne}(z_i))}{\sum_{z_i'} \prod_{\substack{f \text{ involving } z_i \\ \text{and any } \mathrm{Ne}(z_i)}} \Psi_f(z_i',\mathrm{Ne}(z_i))}
\end{aligned}
$$

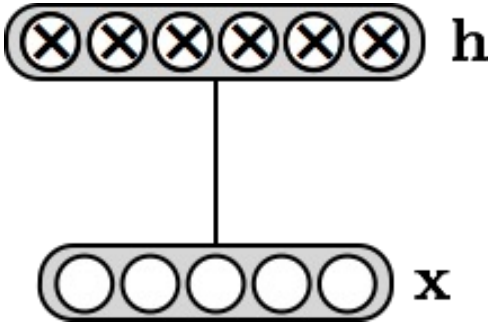
- z_i is any variable in the Markov network (x_k or h_j in an RBM)
- $\mathrm{Ne}(z_i)$ are the neighbors of z_i in the Markov network

Free Energy

- What about computing marginal $p(\mathbf{x})$?

$$
\begin{aligned}
p(\mathbf{x}) &= \sum_{\mathbf{h}\in\{0,1\}^H} p(\mathbf{x},\mathbf{h}) = \sum_{\mathbf{h}\in\{0,1\}^H} \exp(-E(\mathbf{x},\mathbf{h}))/Z \\
&= \exp\left(\mathbf{c}^\top\mathbf{x} + \sum_{j=1}^{H} \log(1+\exp(b_j + \mathbf{W}_{j\cdot}\mathbf{x}))\right)/Z \\
&= \exp(-F(\mathbf{x}))/Z
\end{aligned}
$$

Free Energy



Free Energy

- What about computing marginal $p(\mathbf{x})$?

$$
\begin{aligned}
p(\mathbf{x}) &= \sum_{\mathbf{h}\in\{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h})/Z \\
&= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1\in\{0,1\}} \cdots \sum_{h_H\in\{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j\right) /Z \\
&= \exp(\mathbf{c}^\top \mathbf{x}) \left(\sum_{h_1\in\{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot}\mathbf{x} + b_1 h_1)\right) \dots \left(\sum_{h_H\in\{0,1\}} \exp(h_H \mathbf{W}_{H\cdot}\mathbf{x} + b_H h_H)\right) /Z \\
&= \exp(\mathbf{c}^\top \mathbf{x}) \left(1 + \exp(b_1 + \mathbf{W}_{1\cdot}\mathbf{x})\right) \dots \left(1 + \exp(b_H + \mathbf{W}_{H\cdot}\mathbf{x})\right) /Z \\
&= \exp(\mathbf{c}^\top \mathbf{x}) \exp(\log(1 + \exp(b_1 + \mathbf{W}_{1\cdot}\mathbf{x}))) \dots \exp(\log(1 + \exp(b_H + \mathbf{W}_{H\cdot}\mathbf{x})))/Z \\
&= \exp\left(\mathbf{c}^\top \mathbf{x} + \sum_{j=1}^{H} \log(1 + \exp(b_j + \mathbf{W}_{j\cdot}\mathbf{x}))\right) /Z
\end{aligned}
$$

- Also known as Product of Experts model.

Free Energy

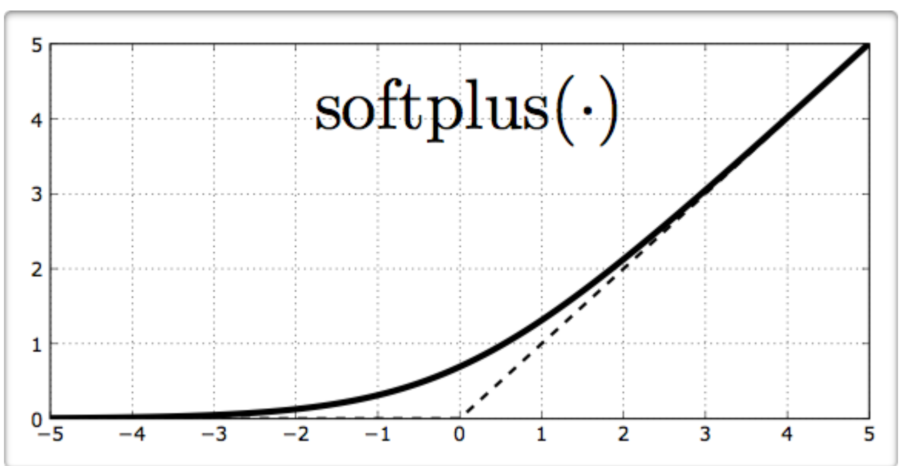
$$p(\mathbf{x}) = \exp\left(\mathbf{c}^\top\mathbf{x} + \sum_{j=1}^{H}\log(1+\exp(b_j+\mathbf{W}_{j\cdot}\mathbf{x}))\right)/Z$$

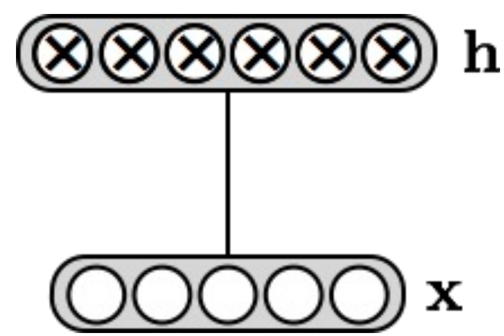
$$= \exp\left(\mathbf{c}^\top\mathbf{x} + \sum_{j=1}^{H}\mathrm{softplus}(b_j+\mathbf{W}_{j\cdot}\mathbf{x})\right)/Z$$

bias the probability of each x_i

bias of each feature

feature expected in x

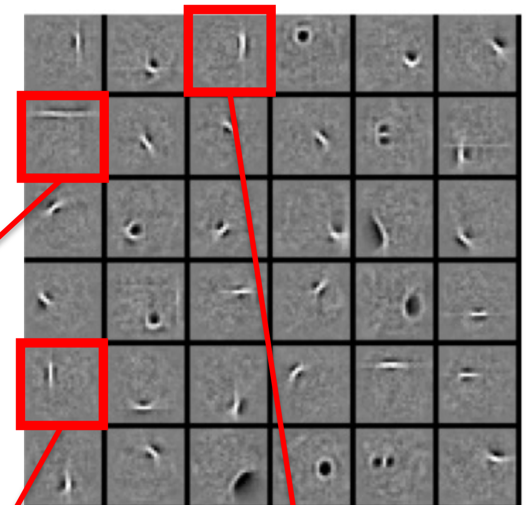




Learning Features

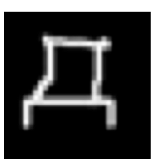
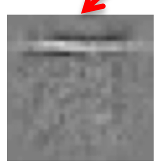
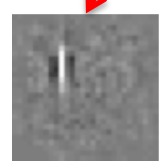
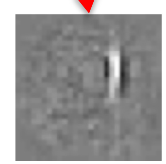
Observed Data
Subset of 25,000 characters

Learned W: "edges"
Subset of 1000 features



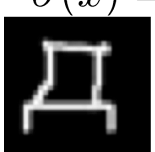
Most hidden variables are off

New Image: $p(h_7 = 1|v)$ $p(h_{29} = 1|v)$

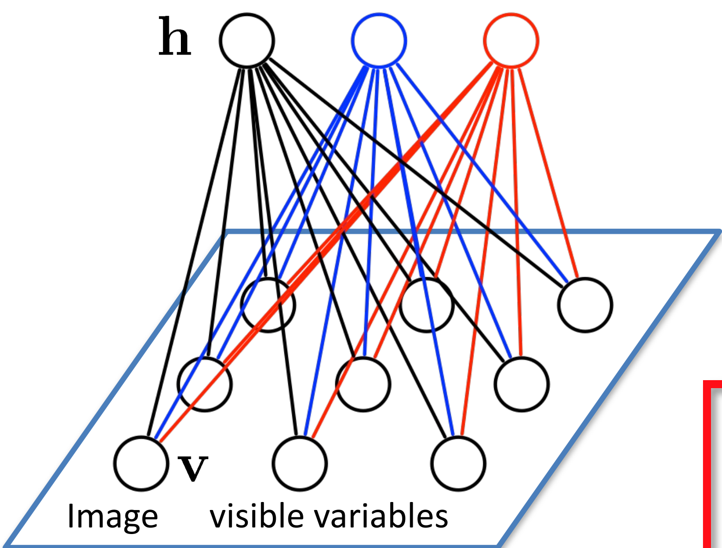
 $= \sigma\Big(0.99 \times$  $+\ 0.97 \times$  $+\ 0.82 \times$  $\cdots\Big)$

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

Logistic Function: Suitable for modeling binary images

Represent:  as $P(\mathbf{h}|\mathbf{v}) = [0,\ 0,\ 0.82,\ 0,\ 0,\ 0.99,\ 0,\ 0\ \ldots\ ]$

Model Learning



- Given a set of *i.i.d.* training examples we want to minimize the average negative log-likelihood (NLL):

$$\frac{1}{T}\sum_t l(f(\mathbf{x}^{(t)})) = \frac{1}{T}\sum_t -\log p(\mathbf{x}^{(t)})$$

Remember:

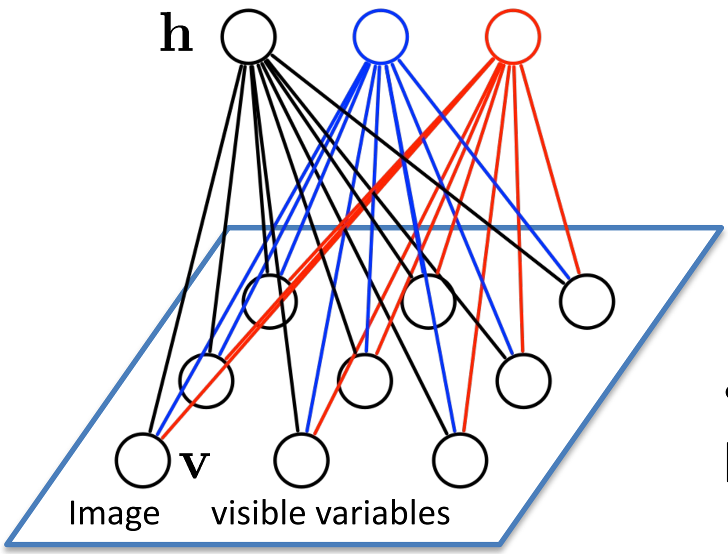
$$p(\mathbf{x},\mathbf{h}) \quad = \quad \exp(-E(\mathbf{x},\mathbf{h}))/Z$$

- Derivative of the negative log-likelihood objective:

$$\frac{\partial -\log p(\mathbf{x}^{(t)})}{\partial\theta} = \underbrace{\mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)},\mathbf{h})}{\partial\theta}\middle|\mathbf{x}^{(t)}\right]}_{\text{Positive Phase}} - \underbrace{\mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial\theta}\right]}_{\text{Negative Phase}}$$

Hard to compute

Model Learning



$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

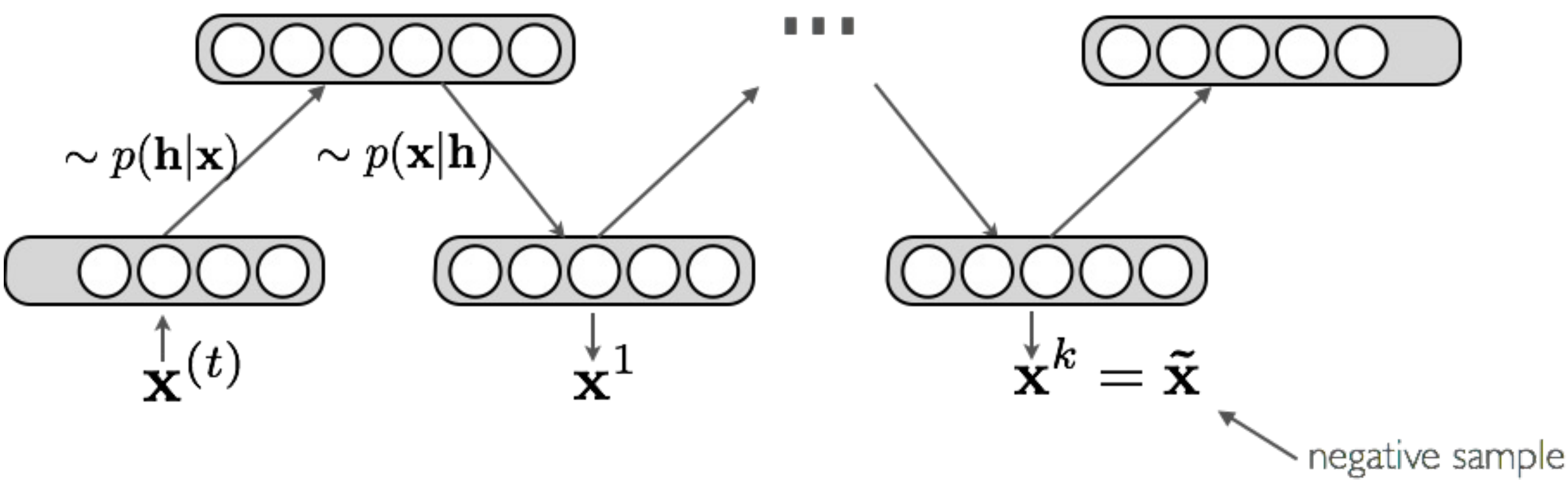
- Derivative of the negative log-likelihood objective:

$$\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \underbrace{\mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta}\Big|\mathbf{x}^{(t)}\right]}_{\text{Data-Dependent Expectations w.r.t } P(h|x)} - \underbrace{\mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]}_{\text{Model: Expectation w.r.t joint } P(x,h)}$$

- Second term: intractable due to exponential number of configurations.

Contrastive Divergence

- Key idea behind Contrastive Divergence:
  - Replace the expectation by a point estimate at $\tilde{\mathbf{x}}$
  - Obtain the point $\tilde{\mathbf{x}}$ by Gibbs sampling
  - Start sampling chain at $\mathbf{x}^{(t)}$

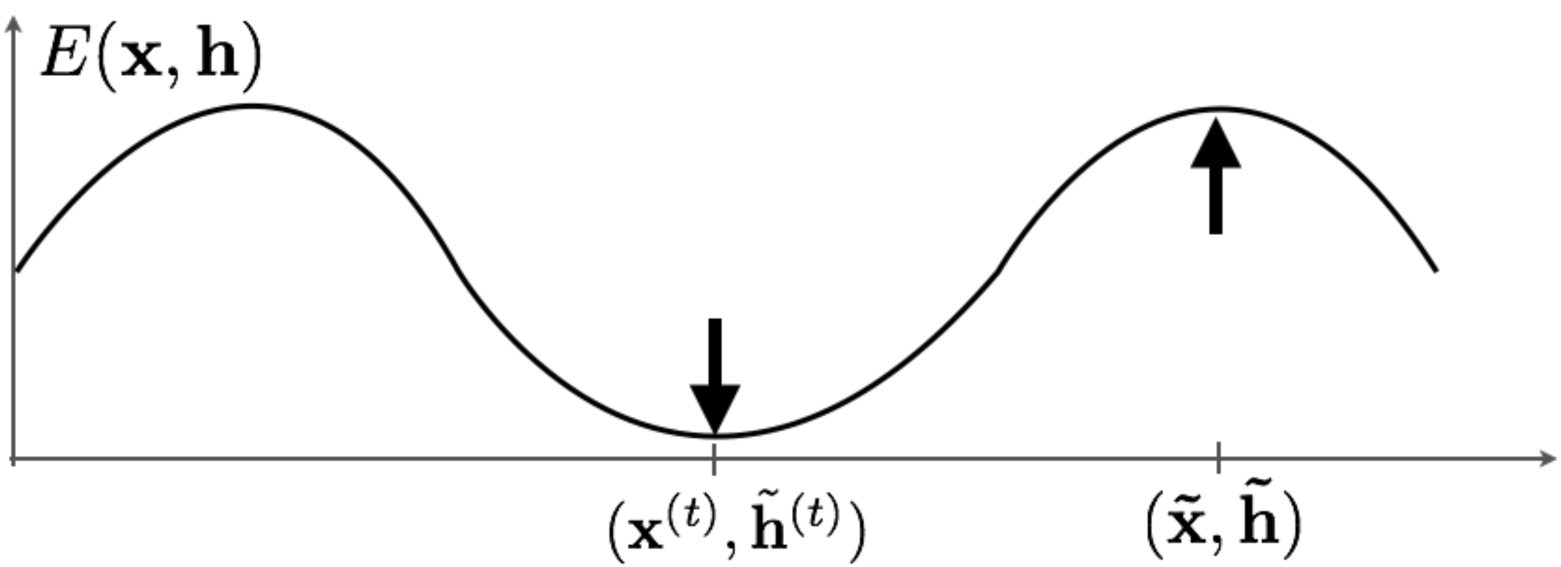


Hinton, Neural Computation, 2002

Contrastive Divergence

- Intuition: $\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \middle| \mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]$

$$\mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \middle| \mathbf{x}^{(t)}\right] \approx \frac{\partial E(\mathbf{x}^{(t)}, \tilde{\mathbf{h}}^{(t)})}{\partial \theta} \qquad \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right] \approx \frac{\partial E(\tilde{\mathbf{x}}, \tilde{\mathbf{h}})}{\partial \theta}$$



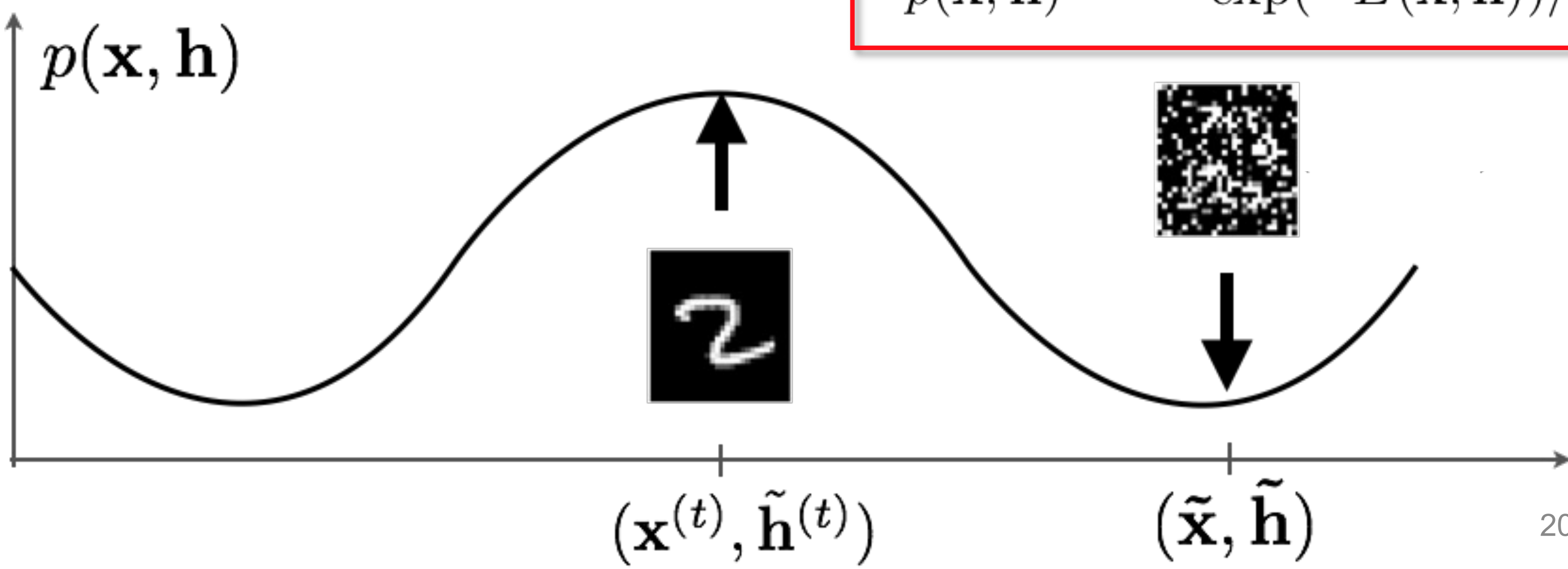
Contrastive Divergence

- Intuition: $\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \middle| \mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]$

$$\mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \middle| \mathbf{x}^{(t)}\right] \approx \frac{\partial E(\mathbf{x}^{(t)}, \tilde{\mathbf{h}}^{(t)})}{\partial \theta} \qquad \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right] \approx \frac{\partial E(\tilde{\mathbf{x}}, \tilde{\mathbf{h}})}{\partial \theta}$$

Remember:

$$p(\mathbf{x}, \mathbf{h}) \quad = \quad \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$



Deriving Learning Rule

- Let us look at derivative of $\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial \theta}$ for $\theta = W_{jk}$

$$\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial W_{jk}} = \frac{\partial}{\partial W_{jk}}\left(-\sum_{jk} W_{jk} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j\right)$$

$$= -\frac{\partial}{\partial W_{jk}} \sum_{jk} W_{jk} h_j x_k$$

$$= -h_j x_k$$

Remember:

$$E(\mathbf{x},\mathbf{h}) \quad = \quad -\mathbf{h}^\top \mathbf{W}\mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h}$$

- Hence:

$$\nabla_{\mathbf{W}} E(\mathbf{x},\mathbf{h}) = -\mathbf{h}\,\mathbf{x}^\top$$

Deriving Learning Rule

- Let us now derive $\mathbb{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial \theta}\middle|\mathbf{x}\right]$

$$\mathbb{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x},\mathbf{h})}{\partial W_{jk}}\middle|\mathbf{x}\right] = \mathbb{E}_{\mathbf{h}}\left[-h_j x_k\middle|\mathbf{x}\right] = \sum_{h_j\in\{0,1\}} -h_j x_k p(h_j|\mathbf{x})$$

$$= -x_k p(h_j = 1|\mathbf{x})$$

- Hence:

$$\mathrm{E}_{\mathbf{h}}\left[\nabla_{\mathbf{W}} E(\mathbf{x},\mathbf{h})\,|\mathbf{x}\right] = -\mathbf{h}(\mathbf{x})\;\mathbf{x}^\top$$

$$\boxed{\begin{aligned}\mathbf{h}(\mathbf{x}) &\stackrel{\text{def}}{=} \begin{pmatrix} p(h_1=1|\mathbf{x}) \\ \cdots \\ p(h_H=1|\mathbf{x})\end{pmatrix} \\ &= \mathrm{sigm}(\mathbf{b}+\mathbf{W}\mathbf{x})\end{aligned}}$$

Deriving Learning Rule

- Hence:

$$\mathrm{E}_{\mathbf{h}}\left[\nabla_{\mathbf{W}} E(\mathbf{x}, \mathbf{h})\,|\mathbf{x}\right] = -\mathbf{h}(\mathbf{x})\; \mathbf{x}^{\top}$$

$$\boxed{\begin{aligned}\mathbf{h}(\mathbf{x}) &\stackrel{\text{def}}{=} \begin{pmatrix} p(h_1{=}1|\mathbf{x}) \\ \cdots \\ p(h_H{=}1|\mathbf{x}) \end{pmatrix} \\ &= \operatorname{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x})\end{aligned}}$$

$$\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta}\,\middle|\mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]$$

$$\sum_{\mathbf{x},\mathbf{h}} -\mathbf{h}(\mathbf{x})\mathbf{x}^{\top}\mathbf{P}(\mathbf{x}, \mathbf{h})$$

Easy to
compute exactly

Difficult to compute:
exponentially many
Configurations.

Approximate Learning

- An approximation to the gradient of the log-likelihood objective:

$$\frac{\partial - \log p(\mathbf{x}^{(t)})}{\partial \theta} = \mathrm{E}_{\mathbf{h}}\left[\frac{\partial E(\mathbf{x}^{(t)}, \mathbf{h})}{\partial \theta} \middle| \mathbf{x}^{(t)}\right] - \mathrm{E}_{\mathbf{x},\mathbf{h}}\left[\frac{\partial E(\mathbf{x}, \mathbf{h})}{\partial \theta}\right]$$

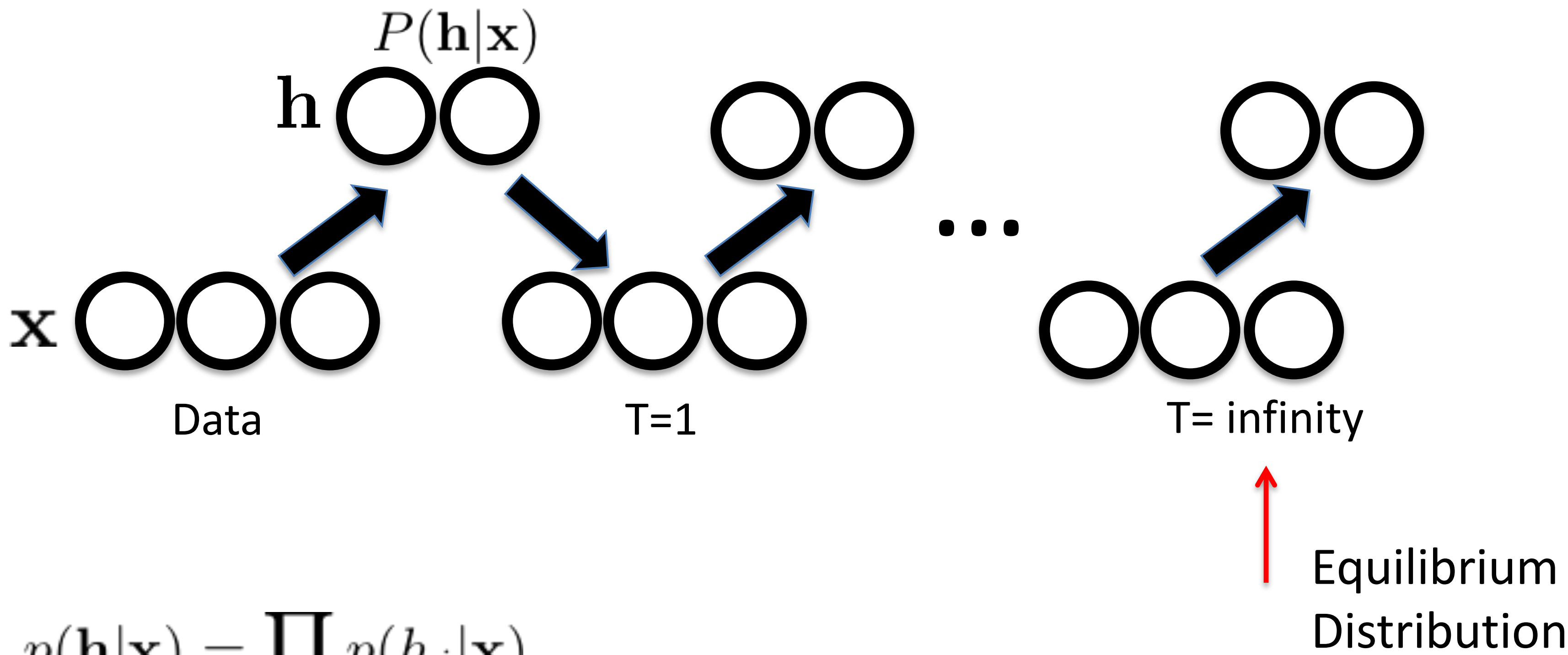
$$\sum_{\mathbf{x},\mathbf{h}} -\mathbf{h}(\mathbf{x})\mathbf{x}^\top \mathbf{P}(\mathbf{x}, \mathbf{h})$$

- Replace the average over all possible input configurations by samples.
- Run MCMC chain (Gibbs sampling) starting from the observed examples.

- Initialize $x^0 = x$
- Sample h^0 from $P(h \mid x^0)$
- For t=1:T
  - Sample x^t from $P(x \mid h^{t-1})$
  - Sample h^t from $P(h \mid x^t)$

Approximate ML Learning for RBMs

Run Markov chain (alternating Gibbs Sampling):

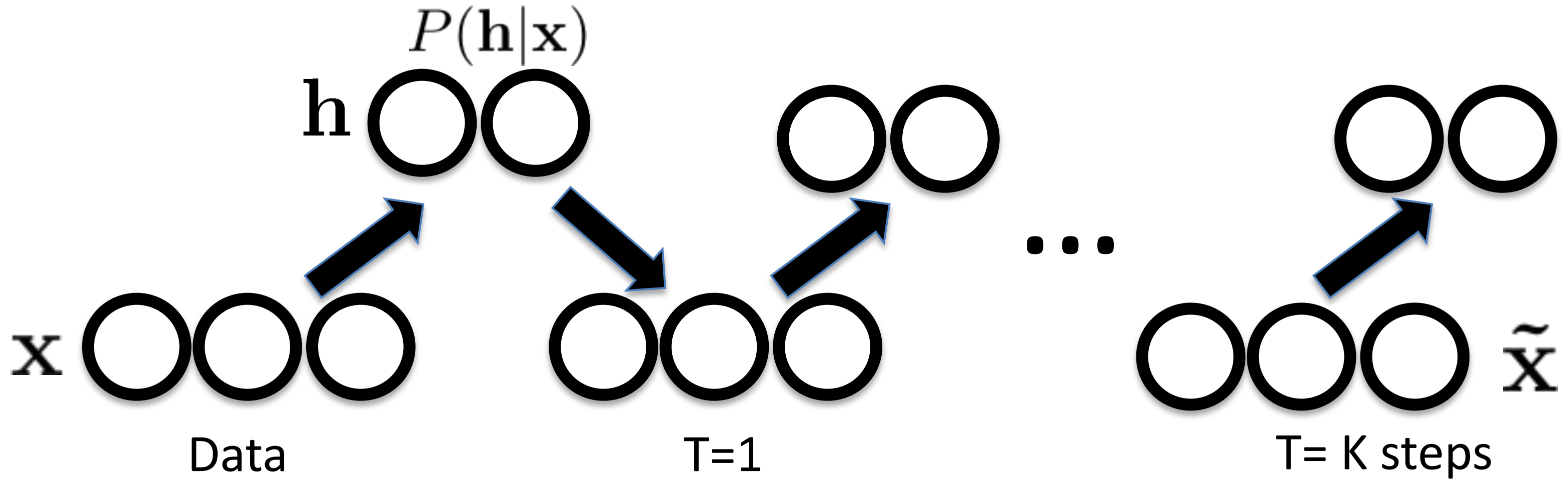


$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

Contrastive Divergence

Run Markov chain (alternating Gibbs Sampling):



- K is typically set to 1.

$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$$

$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h})$$

Deriving Learning Rule

$$\mathbf{x}^{(t)} \qquad \tilde{\mathbf{x}} \qquad\qquad \theta = \mathbf{W}$$

$$\begin{aligned}
\mathbf{W} &\Longleftarrow \mathbf{W} - \alpha \left(\nabla_{\mathbf{W}} - \log p(\mathbf{x}^{(t)}) \right) \\
&\Longleftarrow \mathbf{W} - \alpha \left(\mathrm{E}_{\mathbf{h}} \left[\nabla_{\mathbf{W}} E(\mathbf{x}^{(t)}, \mathbf{h}) \,\middle|\, \mathbf{x}^{(t)} \right] - \mathrm{E}_{\mathbf{x},\mathbf{h}} \left[\nabla_{\mathbf{W}} E(\mathbf{x}, \mathbf{h}) \right] \right) \\
&\Longleftarrow \mathbf{W} - \alpha \left(\mathrm{E}_{\mathbf{h}} \left[\nabla_{\mathbf{W}} E(\mathbf{x}^{(t)}, \mathbf{h}) \,\middle|\, \mathbf{x}^{(t)} \right] - \mathrm{E}_{\mathbf{h}} \left[\nabla_{\mathbf{W}} E(\tilde{\mathbf{x}}, \mathbf{h}) \,|\tilde{\mathbf{x}} \right] \right) \\
&\Longleftarrow \mathbf{W} + \alpha \left(\mathbf{h}(\mathbf{x}^{(t)})\, {\mathbf{x}^{(t)}}^{\top} - \mathbf{h}(\tilde{\mathbf{x}})\, \tilde{\mathbf{x}}^{\top} \right)
\end{aligned}$$

Learning rate

CD-k Algorithm

- For each training example $\mathbf{x}^{(t)}$
  - Generate a negative sample $\tilde{\mathbf{x}}$ using k steps of Gibbs sampling, starting at the data point $\mathbf{x}^{(t)}$
  - Update model parameters:

$$\mathbf{W} \Longleftarrow \mathbf{W} + \alpha \left(\mathbf{h}(\mathbf{x}^{(t)})\, \mathbf{x}^{(t)\top} - \mathbf{h}(\tilde{\mathbf{x}})\, \tilde{\mathbf{x}}^{\top} \right)$$

$$\mathbf{b} \Longleftarrow \mathbf{b} + \alpha \left(\mathbf{h}(\mathbf{x}^{(t)}) - \mathbf{h}(\tilde{\mathbf{x}}) \right)$$

$$\mathbf{c} \Longleftarrow \mathbf{c} + \alpha \left(\mathbf{x}^{(t)} - \tilde{\mathbf{x}} \right)$$

  - Go back to 1 until stopping criteria

CD-k Algorithm

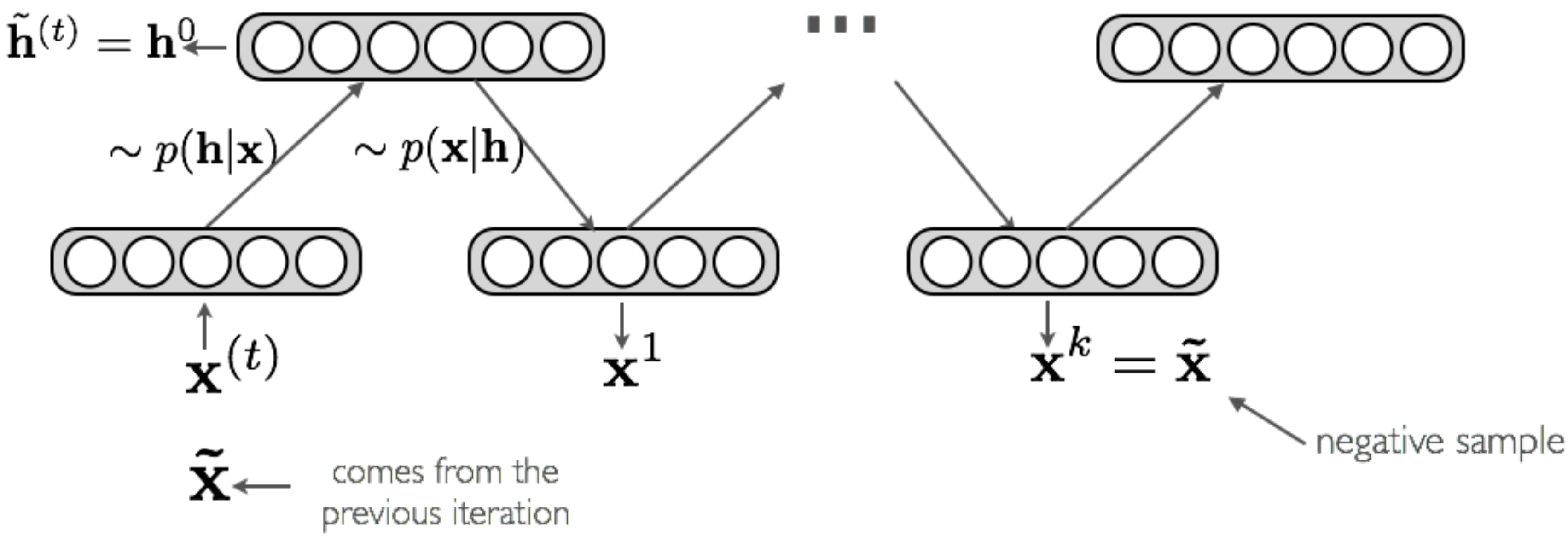
- CD-k: contrastive divergence with k iterations of Gibbs sampling

- In general, the bigger k is, the less biased the estimate of the gradient will be

- In practice, k=1 works well for learning good features and for pre-training

Persistent CD: Stochastic ML Estimator

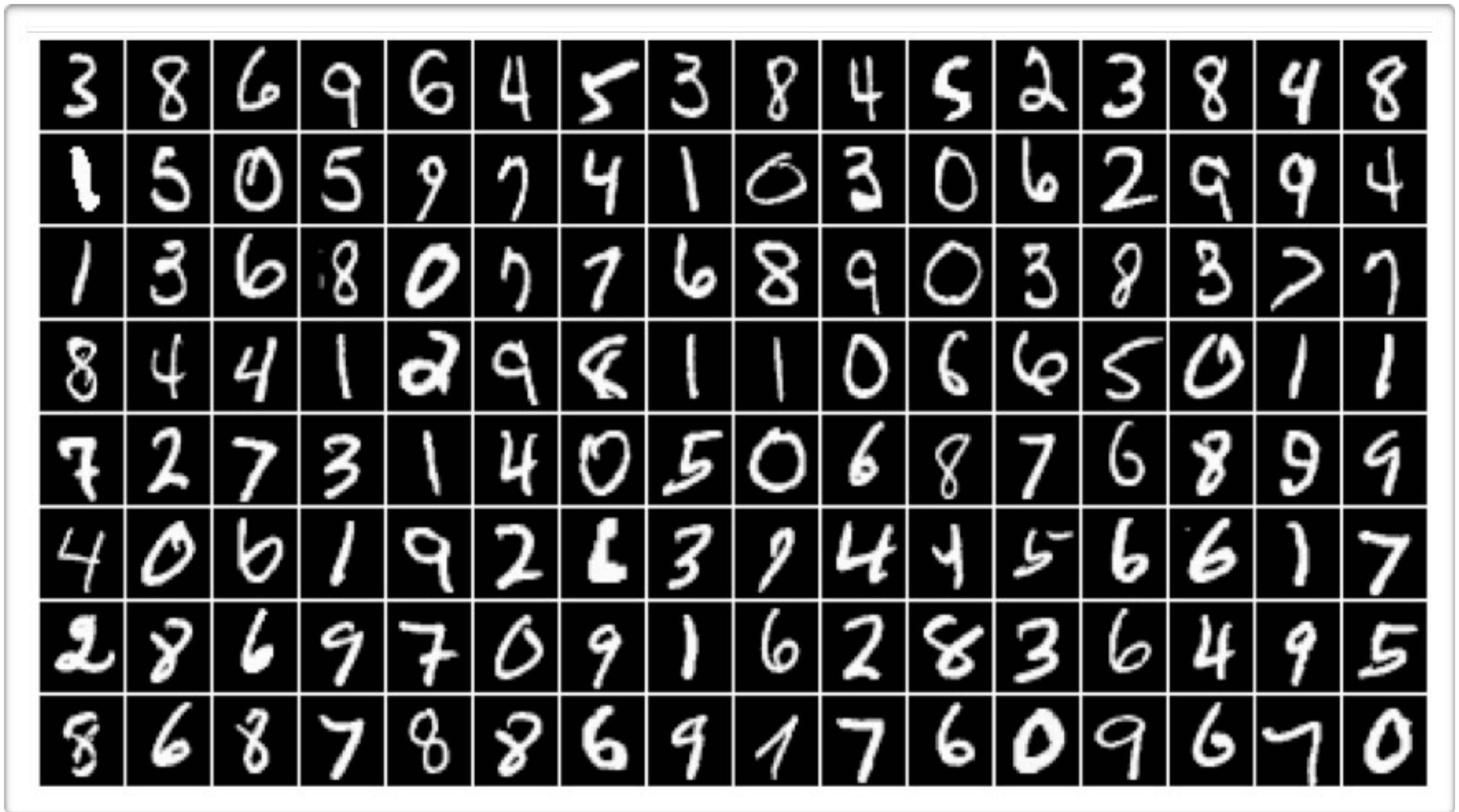
- Idea: instead of initializing the chain to $\mathbf{x}^{(t)}$, initialize the chain to the negative sample of the last iteration



Tieleman, ICML, 2008

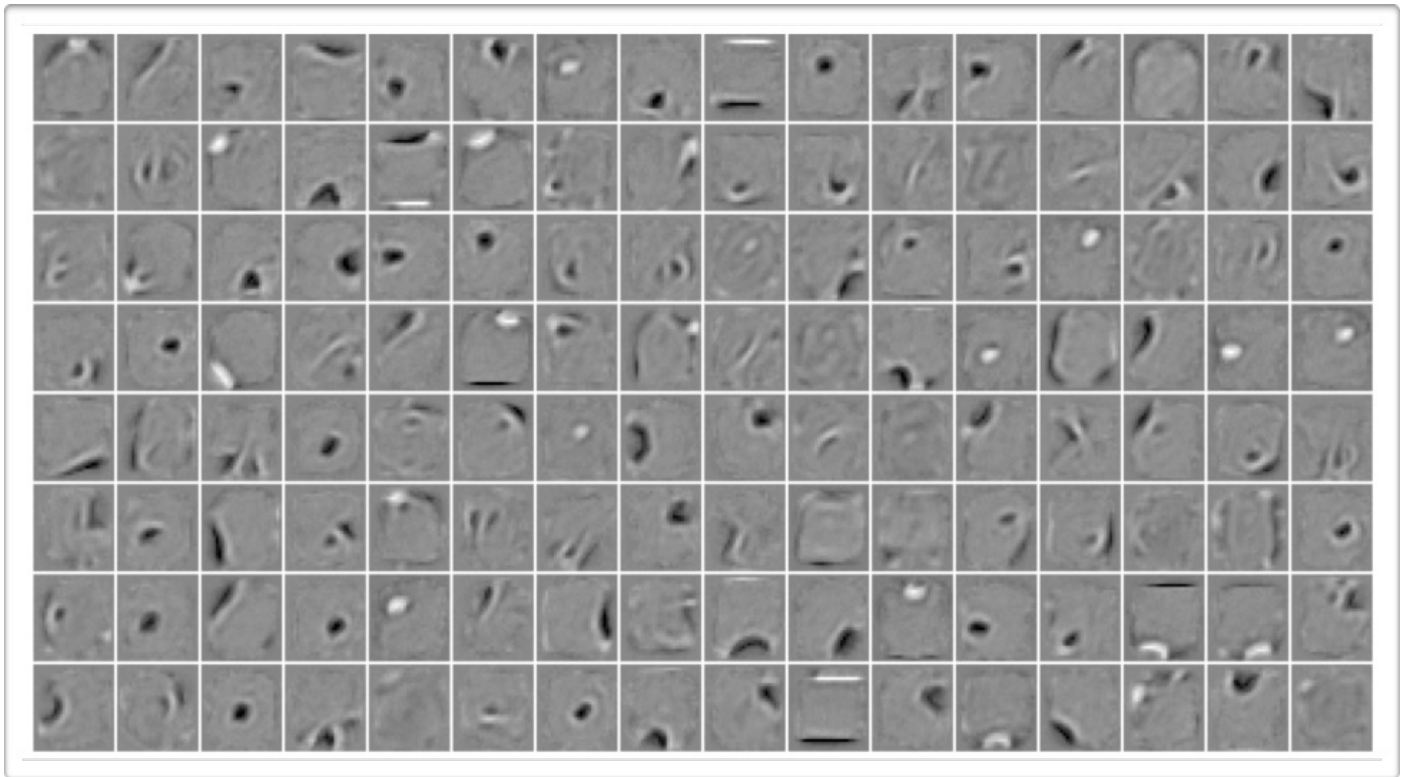
Example: MNIST

- MNIST dataset:



Learned Features

- MNIST dataset:



(Larochelle et al., JMLR 2009)

Tricks and Debugging

- Unfortunately, it is not easy to debug training RBMs (e.g. using gradient checks)
- We instead rely on approximate "tricks"
    - we plot the average stochastic reconstruction $||\mathbf{x}^{(t)} - \tilde{\mathbf{x}}||^2$ and see if it tends to decrease
    - for inputs that correspond to image, we visualize the connection coming into each hidden unit as if it was an image
    - gives an idea of the type of visual feature each hidden unit detects
    - we can also try to approximate the partition function Z and see whether the (approximated) NLL decreases

(Salakhutdinov, Murray, ICML 2008)

Gaussian Bernoulli RBMs

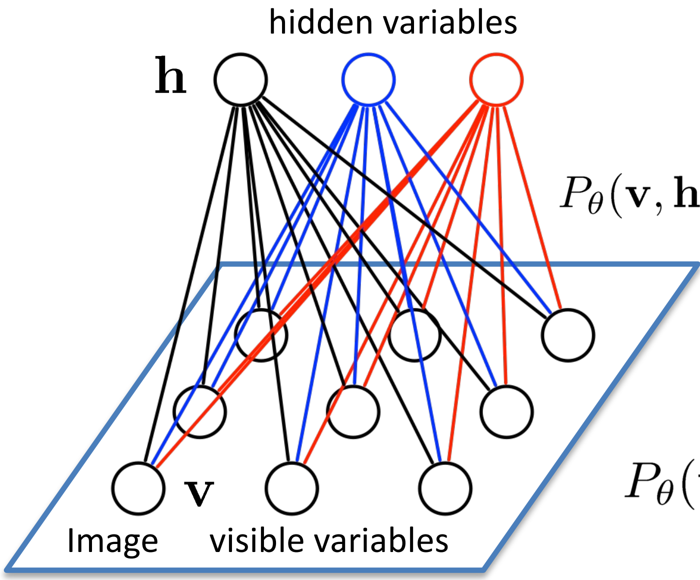
- Let x represent a real-valued (unbounded) input.

  - add a quadratic term to the energy function

$$E(\mathbf{x},\mathbf{h}) = -\mathbf{h}^\top \mathbf{W}\mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h} + \frac{1}{2}\mathbf{x}^\top \mathbf{x}$$

  - In this case $p(\mathbf{x}|\mathbf{h})$ becomes a Gaussian distribution with mean $\boldsymbol{\mu} = \mathbf{c} + \mathbf{W}^\top \mathbf{h}$ and identity covariance matrix
  - recommend to normalize the training set by:
    - subtracting the mean of each input
    - dividing each input by the training set standard deviation
  - should use a smaller learning rate than in the regular RBM

Gaussian Bernoulli RBMs

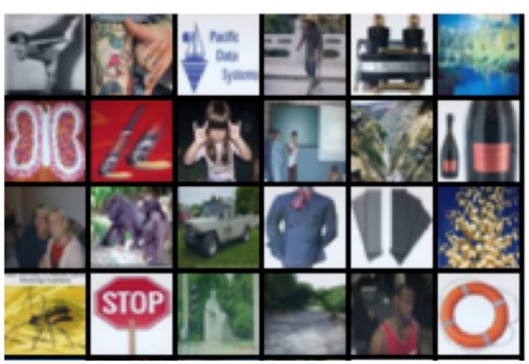


$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{i=1}^{D} \sum_{j=1}^{F} \overbrace{W_{ij} h_j \frac{v_i}{\sigma_i}}^{\textbf{Pair-wise}} + \sum_{i=1}^{D} \overbrace{\frac{(v_i - b_i)^2}{2\sigma_i^2}}^{\textbf{Unary}} + \sum_{j=1}^{F} \overbrace{a_j h_j} \right)$$

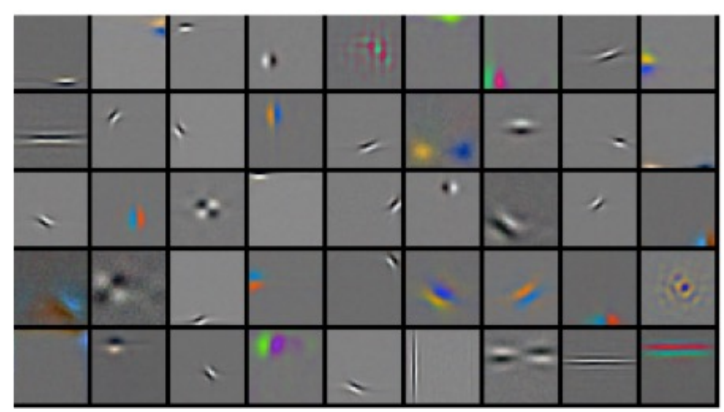
$$\theta = \{W, a, b\}$$

$$P_\theta(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} P_\theta(v_i|\mathbf{h}) = \prod_{i=1}^{D} \mathcal{N}\left(b_i + \sum_{j=1}^{F} W_{ij} h_j, \sigma_i^2 \right)$$

4 million **unlabelled** images



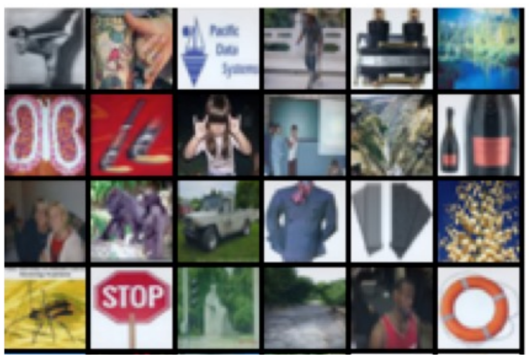
Learned features (out of 10,000)



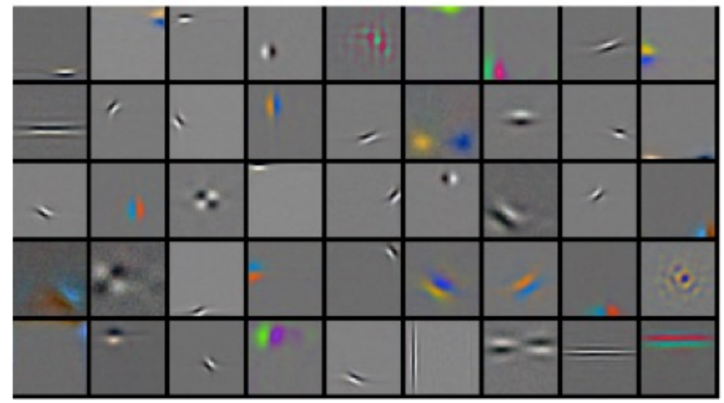
(Notation: vector x is replaced with v).

Gaussian Bernoulli RBMs

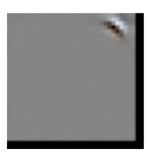
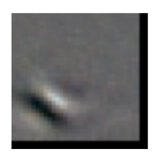
4 million **unlabelled** images



Learned features (out of 10,000)



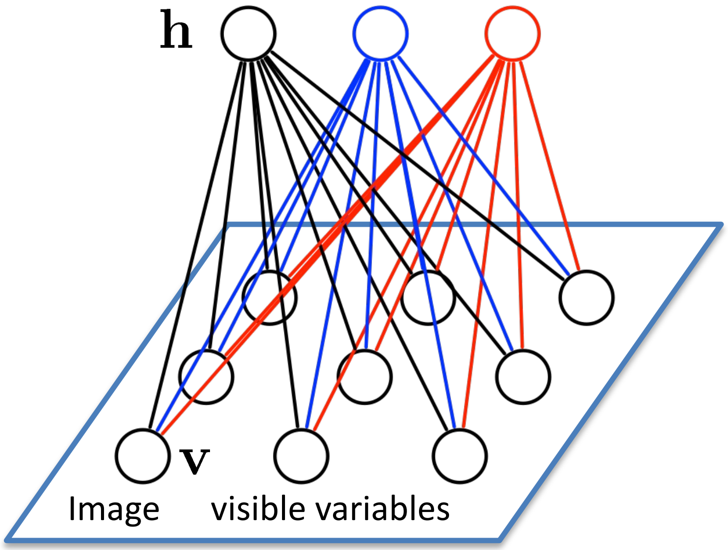
$p(h_7 = 1|v)$ $p(h_{29} = 1|v)$

$=\ 0.9\ *$  $+\ 0.8\ *$  $+\ 0.6\ *$  $\ldots$

New Image

RBMs for Images

Gaussian-Bernoulli RBM:



Interpretation: Mixture of exponential number of Gaussians

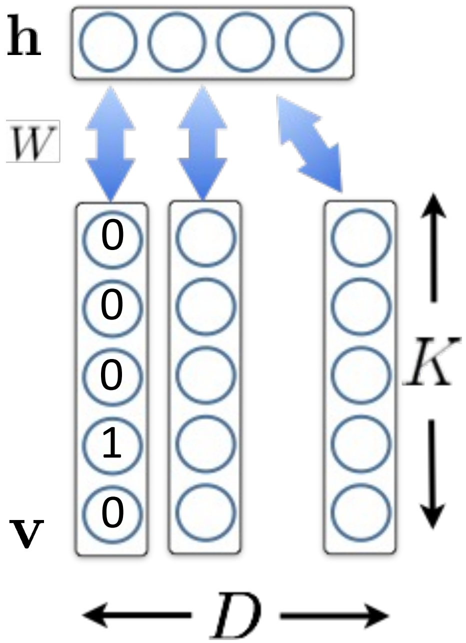
$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}} P_\theta(\mathbf{v}|\mathbf{h}) P_\theta(\mathbf{h}),$$

where

$$P_\theta(\mathbf{h}) = \int_{\mathbf{v}} P_\theta(\mathbf{v}, \mathbf{h}) d\mathbf{v}$$ is an implicit prior, and

$$P(v_i = x|\mathbf{h}) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(x - b_i - \sigma_i \sum_j W_{ij} h_j)^2}{2\sigma_i^2}\right)$$ Gaussian

RBMs for Word Counts



$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\overbrace{\sum_{i=1}^{D} \sum_{k=1}^{K} \sum_{j=1}^{F} W_{ij}^k v_i^k h_j}^{\textbf{Pair-wise}} + \overbrace{\sum_{i=1}^{D} \sum_{k=1}^{K} v_i^k b_i^k + \sum_{j=1}^{F} h_j a_j}^{\textbf{Unary}} \right)$$

$$\theta = \{W, a, b\}$$

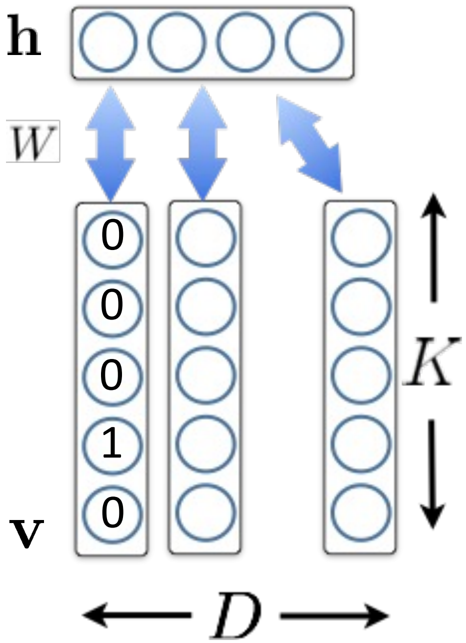
$$P_\theta(v_i^k = 1 | \mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{q=1}^{K} \exp\left(b_i^q + \sum_{j=1}^{F} h_j W_{ij}^q\right)}$$

Replicated Softmax Model: undirected topic model:

- Stochastic 1-of-K visible variables.
- Stochastic binary hidden variables $\mathbf{h} \in \{0, 1\}^F$.
- Bipartite connections.

(Salakhutdinov & Hinton, NIPS 2010, Srivastava & Salakhutdinov, NIPS 2012)

RBMs for Word Counts



$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\overbrace{\sum_{i=1}^{D}\sum_{k=1}^{K}\sum_{j=1}^{F} W_{ij}^k v_i^k h_j}^{\textbf{Pair-wise}} + \overbrace{\sum_{i=1}^{D}\sum_{k=1}^{K} v_i^k b_i^k + \sum_{j=1}^{F} h_j a_j}^{\textbf{Unary}} \right)$$

$$\theta = \{W, a, b\}$$

$$P_\theta(v_i^k = 1|\mathbf{h}) = \frac{\exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{q=1}^{K} \exp\left(b_i^q + \sum_{j=1}^{F} h_j W_{ij}^q\right)}$$

Reuters dataset:
804,414 **unlabeled**
newswire stories
Bag-of-Words

Learned features: ``topics''

russian	clinton	computer	trade	stock
russia	house	system	country	wall
moscow	president	product	import	street
yeltsin	bill	software	world	point
soviet	congress	develop	economy	dow

RBMs for Word Counts

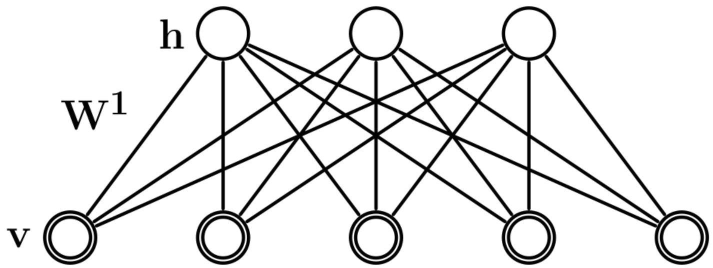
One-step reconstruction from the Replicated Softmax model.

Input	**Reconstruction**
chocolate, cake	cake, chocolate, sweets, dessert, cupcake, food, sugar, cream, birthday
nyc	nyc, newyork, brooklyn, queens, gothamist, manhattan, subway, streetart
dog	dog, puppy, perro, dogs, pet, filmshots, tongue, pets, nose, animal
flower, high, 花	flower, 花, high, japan, sakura, 日本, blossom, tokyo, lily, cherry
girl, rain, station, norway	norway, station, rain, girl, oslo, train, umbrella, wet, railway, weather
fun, life, children	children, fun, life, kids, child, playing, boys, kid, play, love
forest, blur	forest, blur, woods, motion, trees, movement, path, trail, green, focus
españa, agua, granada	españa, agua, spain, granada, water, andalucía, naturaleza, galicia, nieve

Collaborative Filtering

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\left(\sum_{ijk} W_{ij}^k v_i^k h_j + \sum_{ik} b_i^k v_i^k + \sum_j a_j h_j \right)$$

Binary hidden: user preferences



Multinomial visible: user ratings

Netflix dataset:
480,189 users
17,770 movies
Over 100 million ratings

NETFLIX

Learned features: ``genre''

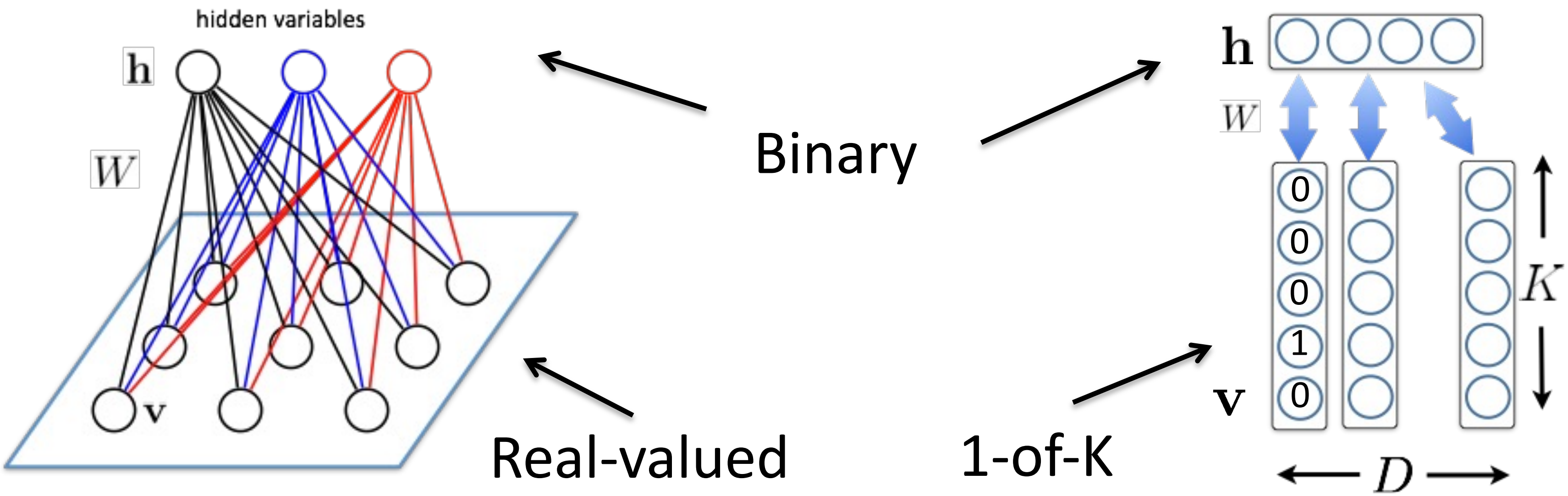
Fahrenheit 9/11	Independence Day
Bowling for Columbine	The Day After Tomorrow
The People vs. Larry Flynt	Con Air
Canadian Bacon	Men in Black II
La Dolce Vita	Men in Black
Friday the 13th	Scary Movie
The Texas Chainsaw Massacre	Naked Gun
Children of the Corn	Hot Shots!
Child's Play	American Pie
The Return of Michael Myers	Police Academy

State-of-the-art performance on the Netflix dataset.

(Salakhutdinov, Mnih, Hinton, ICML 2007)

Different Data Modalities

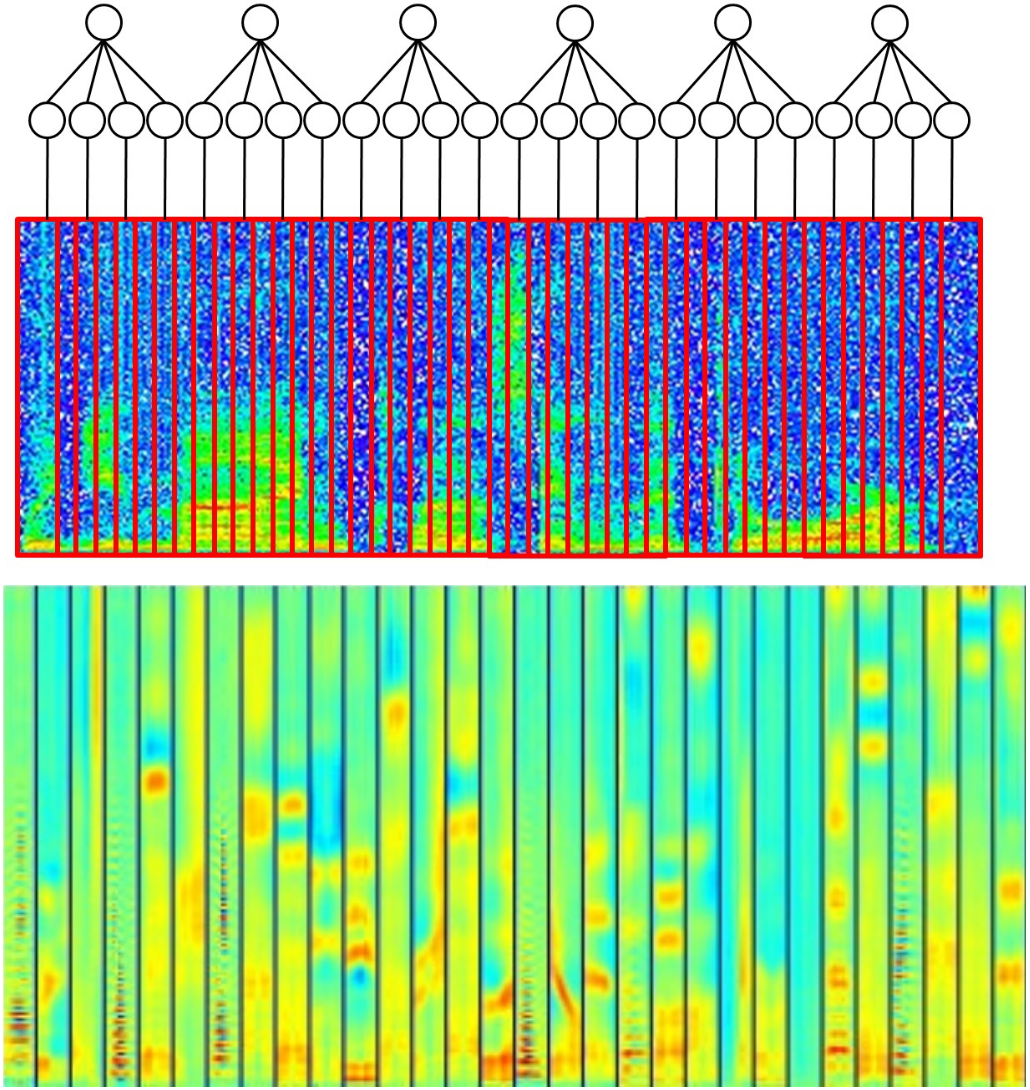
- Binary/Gaussian/Softmax RBMs: All have binary hidden variables but use them to model different kinds of data.



- It is easy to infer the states of the hidden variables:

$$P_\theta(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{F} P_\theta(h_j|\mathbf{v}) = \prod_{j=1}^{F} \frac{1}{1+\exp(-a_j - \sum_{i=1}^{D} W_{ij} v_i)}$$

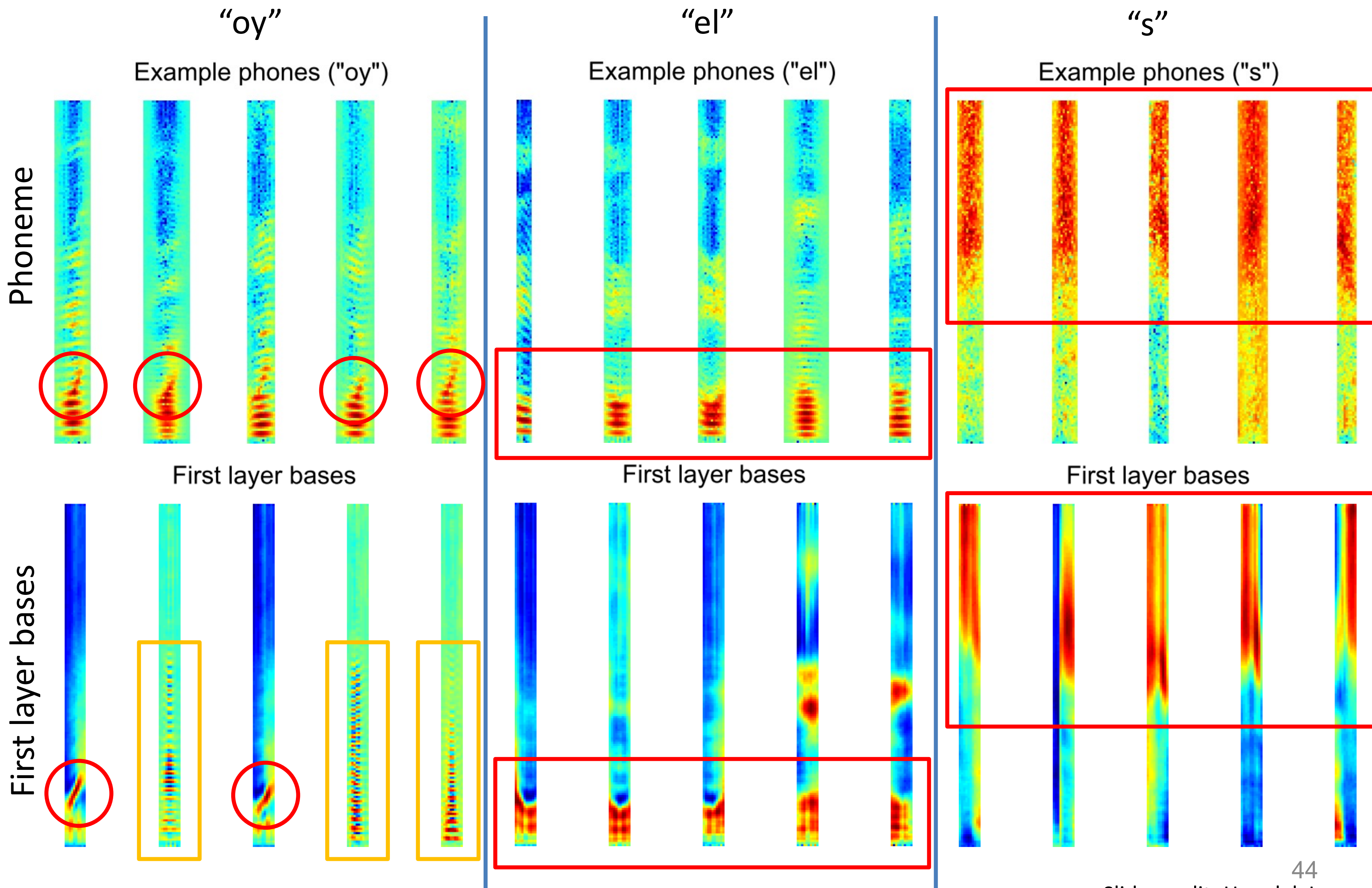
Speech



Learned first-layer bases

Lee et.al., NIPS 2009

Comparison of bases to phonemes



Slide credit: Honglak Lee

Product of Experts

The joint distribution is given by:

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\Big(\sum_{ij} W_{ij} v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j\Big)$$

Marginalizing over hidden variables:

$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}} P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \prod_i \exp(b_i v_i) \overbrace{\prod_j \Big(1 + \exp(a_j + \sum_i W_{ij} v_i)\Big)}^{\textbf{Product of Experts}}$$

government	clinton	bribery	mafia	stock	…
authority	house	corruption	business	wall	
power	president	dishonesty	gang	street	
empire	bill	corrupt	mob	point	
federation	congress	fraud	insider	dow	

Silvio Berlusconi

Topics "government", "corruption" and "mafia" can combine to give very high probability to a word "Silvio Berlusconi".

Product of Experts

The joint distribution is given by:

$$P_\theta(\mathbf{v},\mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp\Big(\sum_{ij} W_{ij} v_i h_j + \sum_i b_i v_i + \sum_j a_j h_j\Big)$$

Marginalizing c

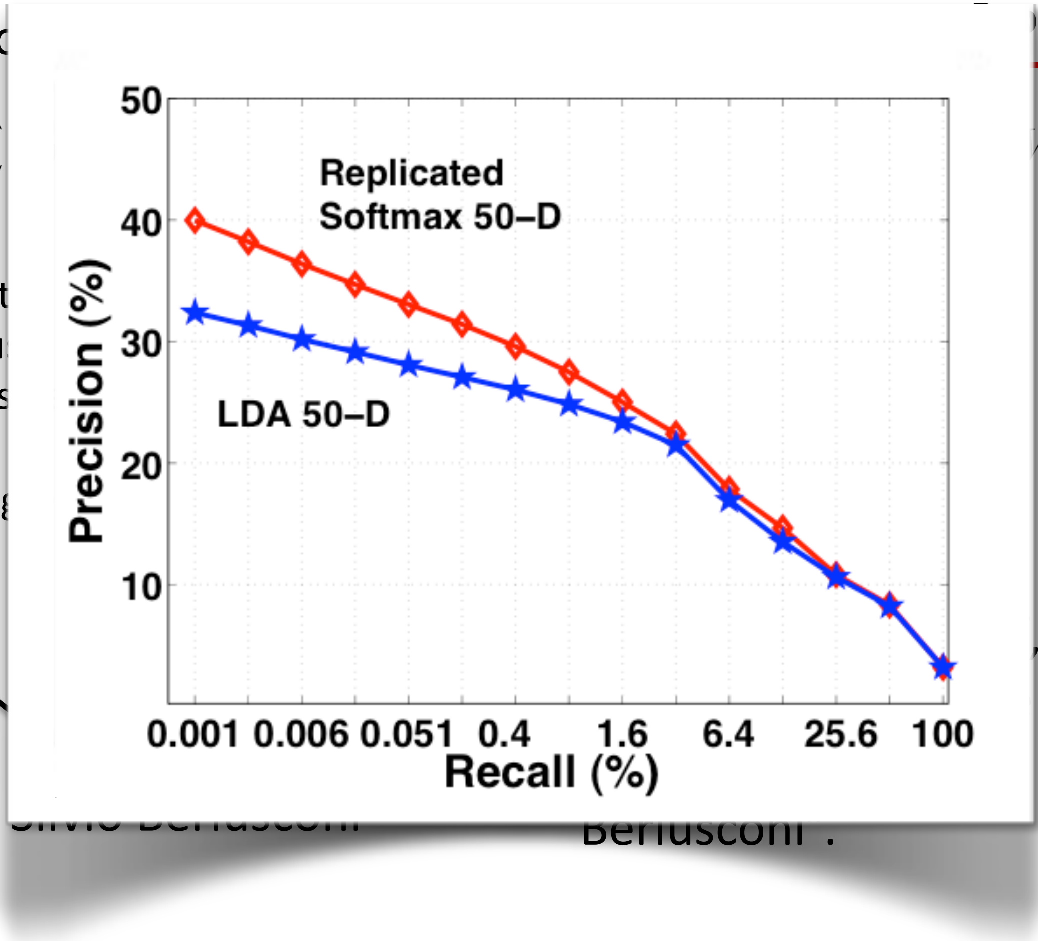
$$P_\theta(\mathbf{v}) = \sum_{\mathbf{h}}$$

duct of Experts

$$W_{ij} v_i)\Big)$$

government
authority
power
empire
federation

clint
hou
pres
bill
con



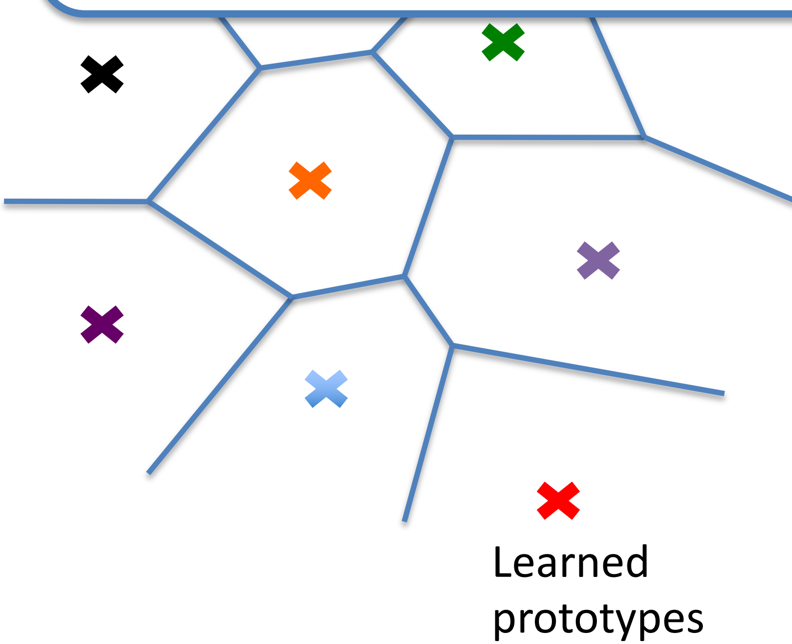
, "corruption" and
to give very high
"Silvio

Berlusconi".

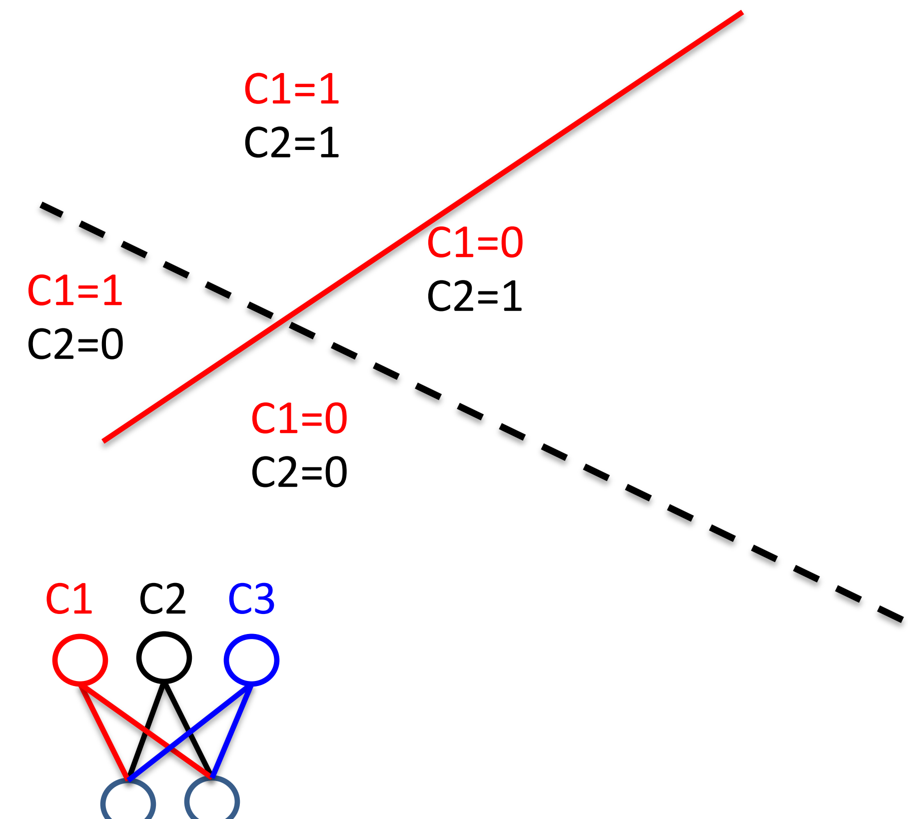
Local vs. Distributed Representations

- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- Parameters for each region.
- # of regions is linear with # of parameters.



- RBMs, Factor models, PCA, Sparse Coding, Deep models

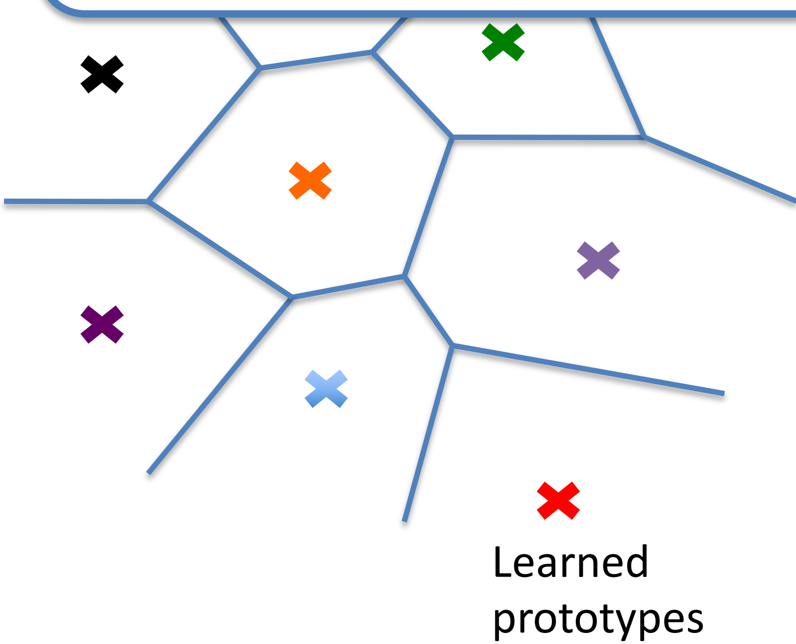


Bengio, 2009, Foundations and Trends in Machine Learning

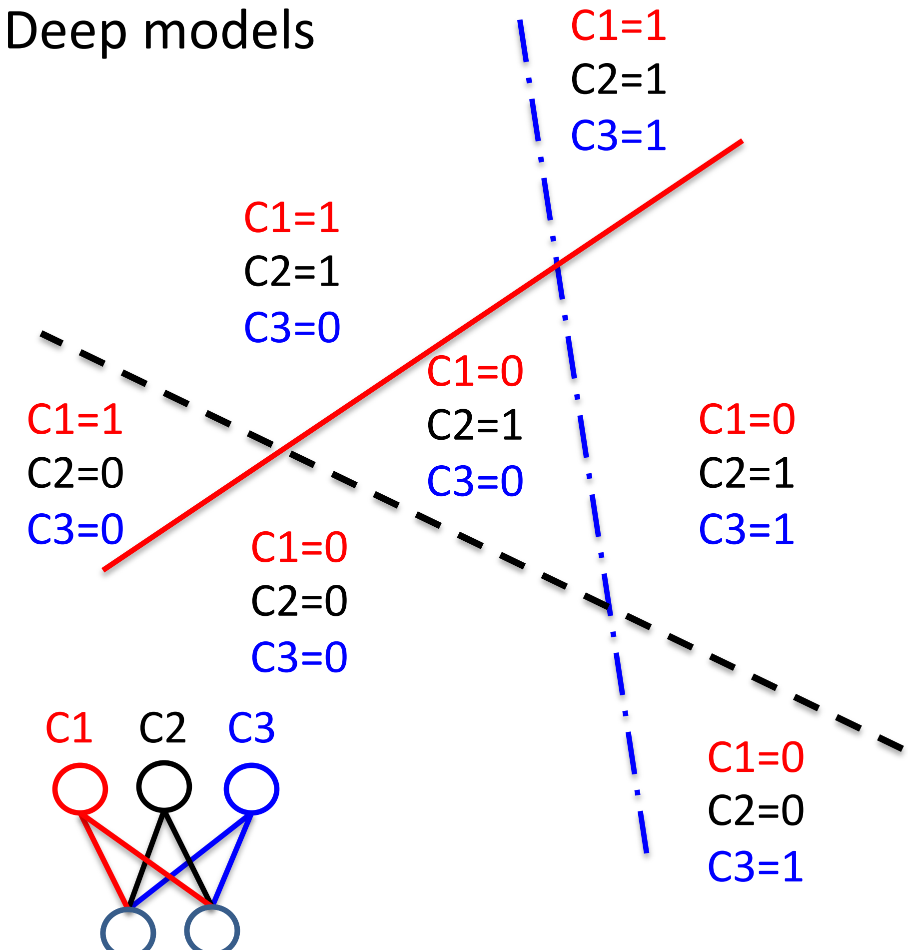
Local vs. Distributed Representations

- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- Parameters for each region.
- # of regions is linear with # of parameters.



- RBMs, Factor models, PCA, Sparse Coding, Deep models



Bengio, 2009, Foundations and Trends in Machine Learning

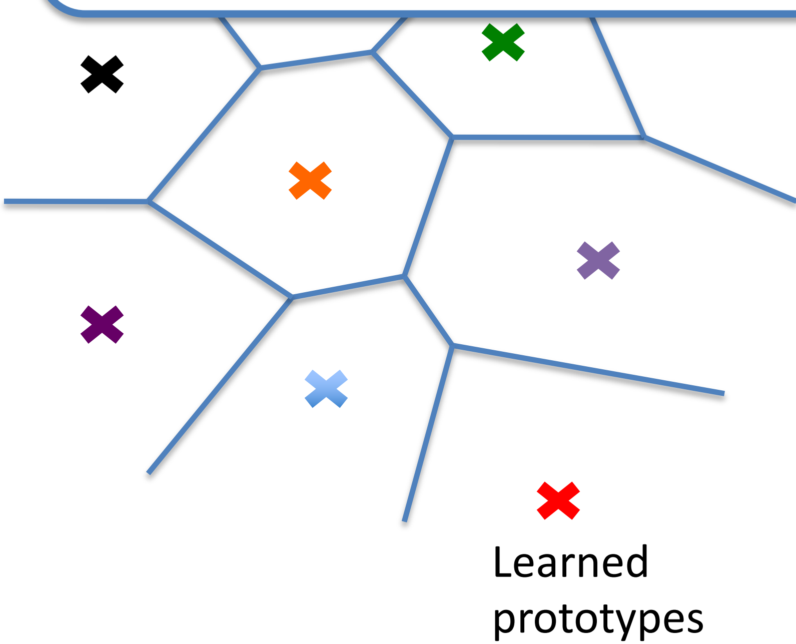
Local vs. Distributed Representations

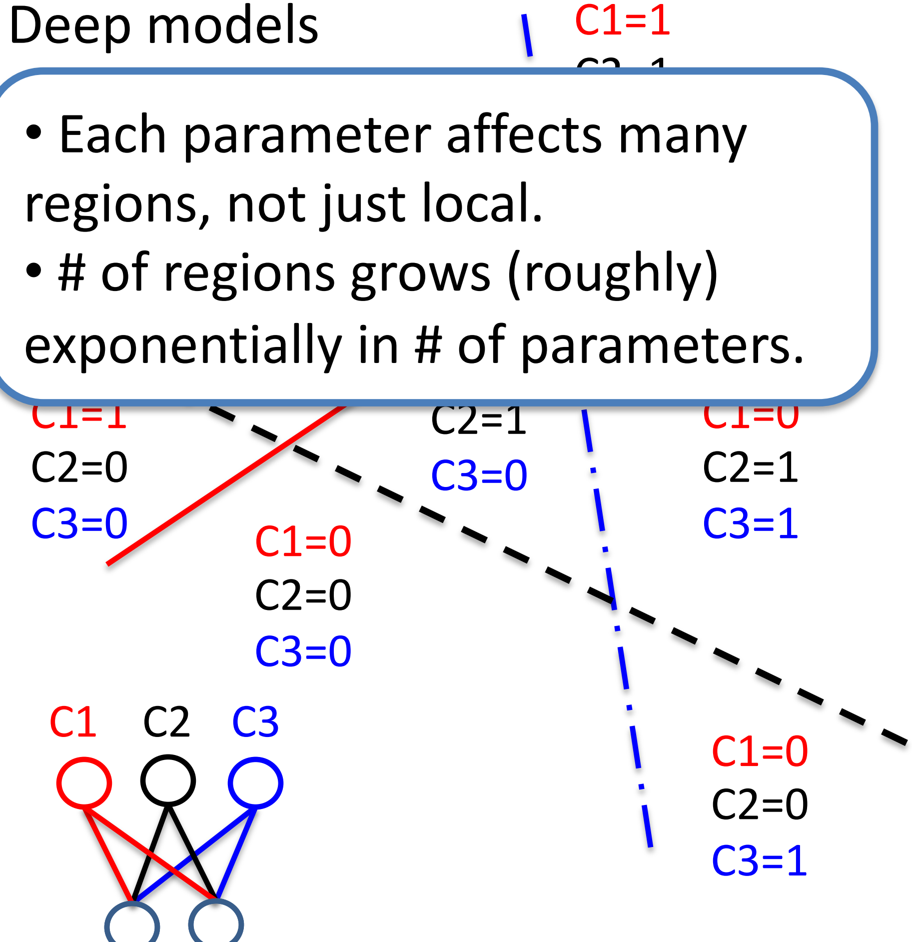
- Clustering, Nearest Neighbors, RBF SVM, local density estimators

- RBMs, Factor models, PCA, Sparse Coding, Deep models

- Parameters for each region.
- # of regions is linear with # of parameters.

- Each parameter affects many regions, not just local.
- # of regions grows (roughly) exponentially in # of parameters.

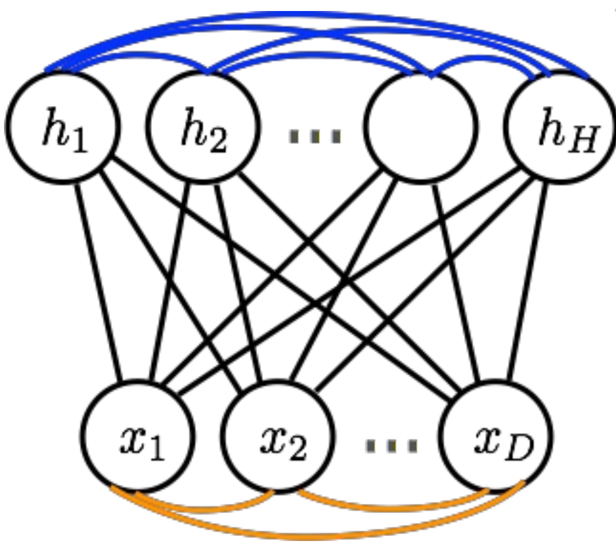




Bengio, 2009, Foundations and Trends in Machine Learning

Boltzmann Machines

- The original Boltzmann machine has lateral connections in each layer



$$E(\mathbf{x},\mathbf{h}) = -\mathbf{h}^\top \mathbf{W}\mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h} \boxed{-\frac{1}{2}\mathbf{x}^\top \mathbf{V}\mathbf{x}} \boxed{-\frac{1}{2}\mathbf{h}^\top \mathbf{U}\mathbf{h}}$$

- ➢ when only one layer has lateral connection, the model is called a semi-restricted Boltzmann machine