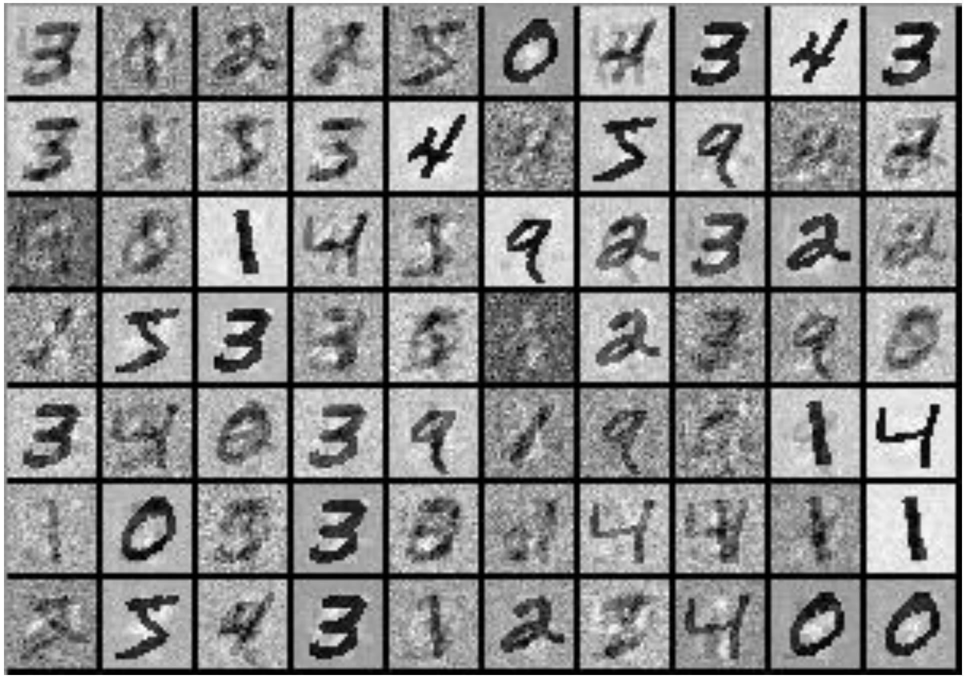
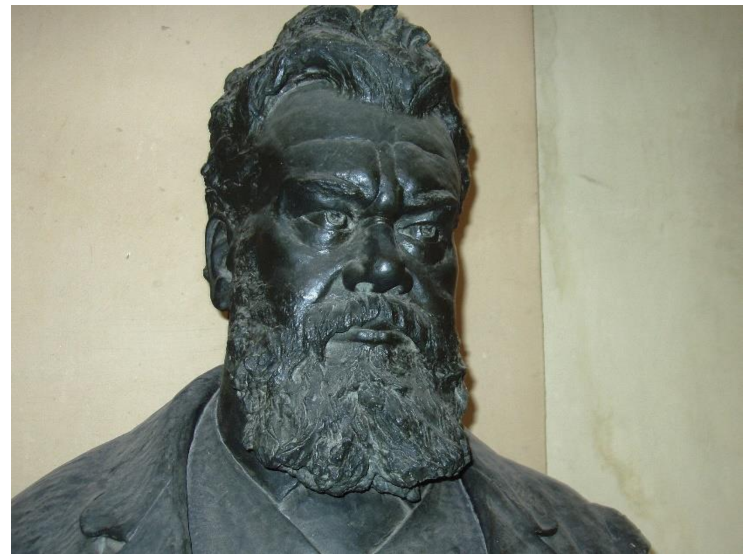
Training RBMs





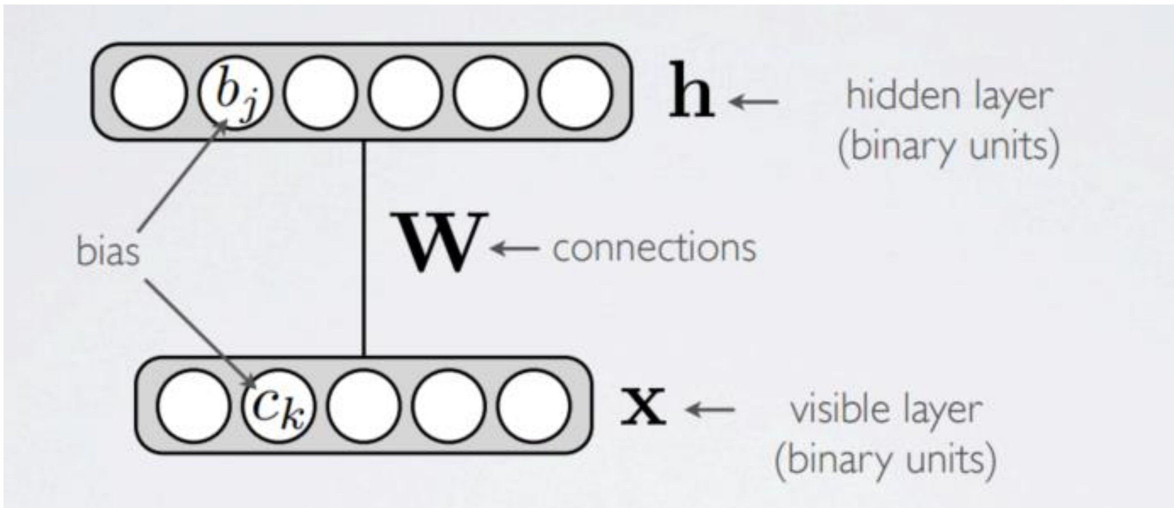
http://deeplearning4j.org/rbm-mnist-tutorial.html

Slides from Hugo Larochelle, Geoffrey Hinton, and Yoshua Bengio

CSC321: Intro to Machine Learning and Neural Networks, Winter 2016

Michael Guerzhoy

RBM Refresher



h, x:
Binary vecs.
$(h_i, x_j \in \{0,1\})$

$$E(x,h) = -h^T W x - c^T x - b^T h$$

$$= -\sum_j \sum_k W_{jk} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j$$

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

$$P(x) = \sum_{h'} P(x,h')$$

Last time

- We saw how to sample from an RBM
  - The weights and biases were fixed

Learning an RBM

- Want to find weights *W* and biases *b* and *c* such that the probability of the training set $P_{W,b,c}(\boldsymbol{x}) = \Pi_i P_{W,b,c}(x^{(i)})$ is maximized
- For a new input z, we hope that $P_{W,b,c}(z)$ will be large if z came from the same source as the training set
- Denote $P_{W,b,c} = P_\theta$

P(x)

- $P(x,h) = \frac{\exp(-E(x,h))}{Z}, Z = \sum_{(x',h')} \exp(-E(x',h'))$
- $P(x) = \sum_{h'} P(x,h') = \frac{\exp(-\mathrm{FreeE}(x))}{\mathrm{Z}}, \mathrm{Z} = \sum_{x'} \exp(-FreeE(x'))$
- Free Energy:

$$FreeE(x) = -\log \sum_{h'} \exp(-E(x,h'))$$

Proof:

$$\exp(-\mathrm{FreeE}(\mathrm{x})) = \sum_{h'} \exp(-E(x,h')) \propto P(x)$$

- $\log P(x) = -FreeE(x) - \log \sum_{x'} \exp(-FreeE(x'))$

$$\frac{\partial \log P_\theta(x)}{\partial \theta}$$

- $\frac{\partial \log P_\theta(x)}{\partial \theta} = \frac{\partial}{\partial \theta}\left(-FreeE(x) - \log \sum_{x'} \exp\left(-FreeE(x')\right)\right)$
- $= -\frac{\partial}{\partial \theta} FreeE(x) + \frac{1}{\sum_{x'} \exp\left(-FreeE(x')\right)} \sum_{x'} \exp\left(-FreeE(x')\right) \frac{\partial}{\partial \theta} FreeE(x')$
- $= -\frac{\partial}{\partial \theta} FreeE(x) + \frac{1}{Z} \sum_{x'} \exp\left(-FreeE(x')\right) \frac{\partial}{\partial \theta} FreeE(x')$
- $= -\frac{\partial}{\partial \theta} FreeE(x) + \sum_{x'} \exp \frac{\left(-FreeE(x')\right)}{Z} \frac{\partial}{\partial \theta} FreeE(x')$
- $= -\frac{\partial}{\partial \theta} FreeE(x) + \sum_{x'} P_\theta(x') \frac{\partial}{\partial \theta} FreeE(x')$

$$\frac{\partial \log P_\theta(x)}{\partial\theta} = -\frac{\partial}{\partial\theta} FreeE(x) + \sum_{x'} P_\theta(x') \frac{\partial}{\partial\theta} FreeE(x')$$

- Can approximate $\sum_{x'} P_\theta(x') \frac{\partial}{\partial\theta} FreeE(x')$ by only sampling some x' from $P_\theta(x')$, computing $\frac{\partial}{\partial\theta} FreeE(x')$ for those x', and averaging the results.
- Note: computing $\frac{\partial}{\partial\theta} FreeE(x')$ is a bit of a pain, but it's feasible

Sampling from $P_\theta(x')$

- Reminder from last time:
  - Guess an initial x'
  - Repeat:
    - Sample a new h using P(h|x)
    - Sample a new x using P(x|h)

Contrastive Divergence

- A shortcut that works really well in practice
- Start from x, a training sample (higher probability than for a random guess)
- Sample h given x, then sample a new x' given that h
- Now, for a single training sample x, use
- $\frac{\partial \log P_\theta(x)}{\partial \theta} \approx -\frac{\partial}{\partial \theta} FreeE(x) + \frac{\partial}{\partial \theta} FreeE(x')$

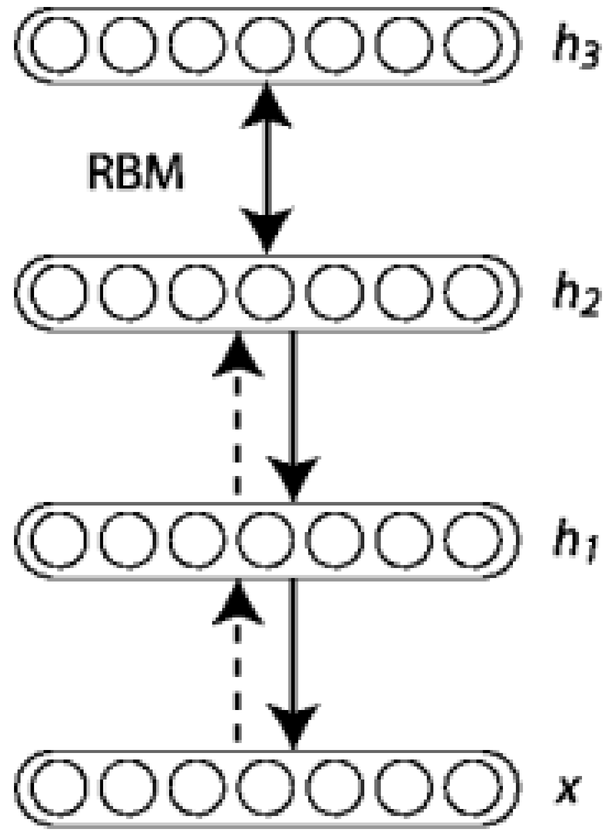
Approximation for $\sum_{x'} P_\theta(x') \frac{\partial}{\partial \theta} FreeE(x')$

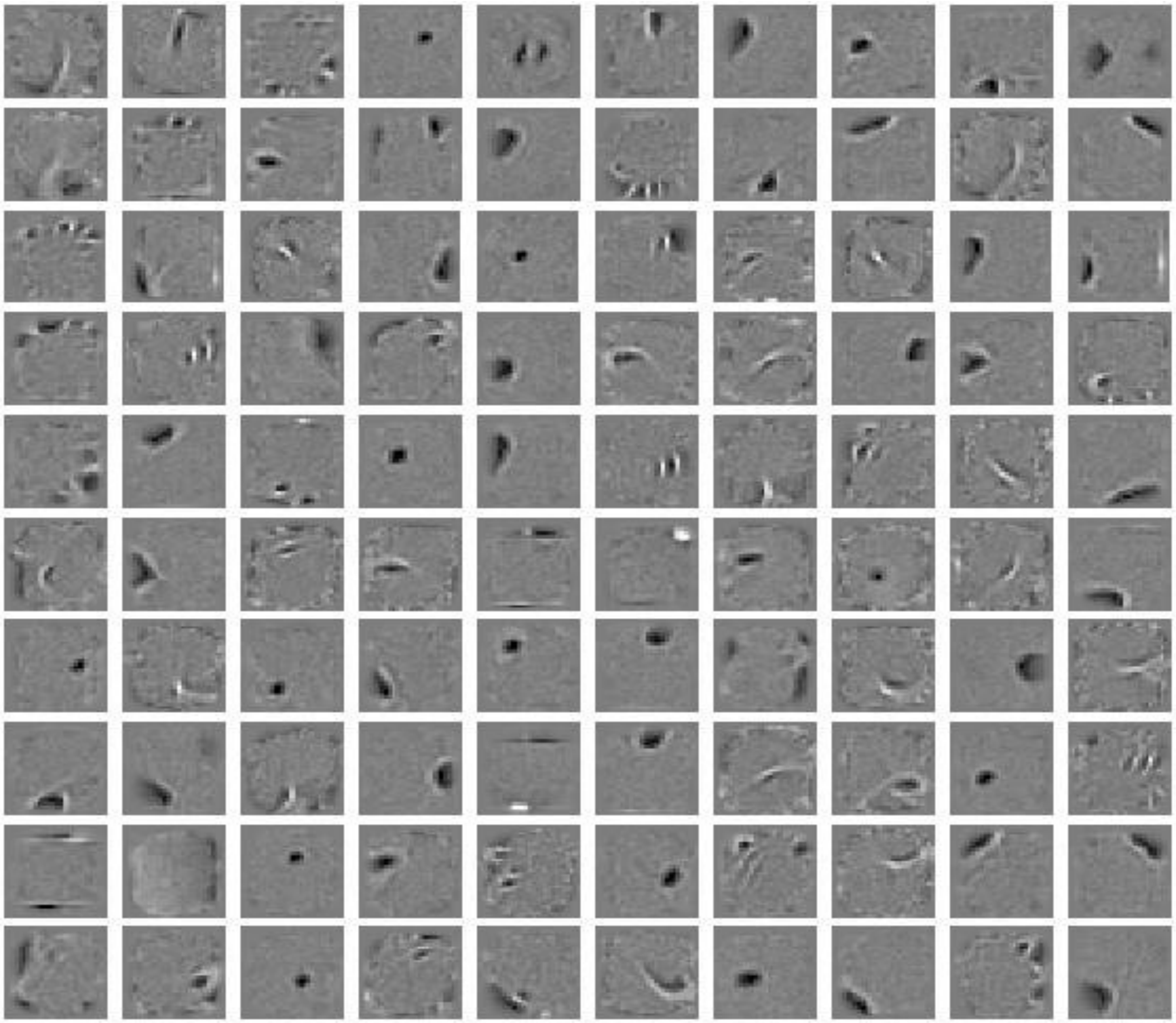
$$\frac{\partial \log P_\theta(x)}{\partial \theta} \approx -\frac{\partial}{\partial \theta} FreeE(x) + \frac{\partial}{\partial \theta} FreeE(x')$$

- x' is a "fantasy"/"dream"/"fake data" generated by the RBM using the current weights
  - Want to make the Free Energy for it large (i.e., want to make the probability of the dream small)
- x is data from the actual training set
  - Want to make the Free Energy for it small (i.e., want to make the probability of the real training set large)
- This is exactly what gradient ascent will do!
- (A reason to dream: it can make your model of the world better!)

Deep Belief Networks (not covered in detail)

- RBM's stacked on top of each other
- Train the bottom RBM, then sample h1 given each input x to get a new training set
- Now train the second RBM from the bottom
- …





Some features learned in the first hidden layer of a model of all 10 digit classes using 500 hidden units.