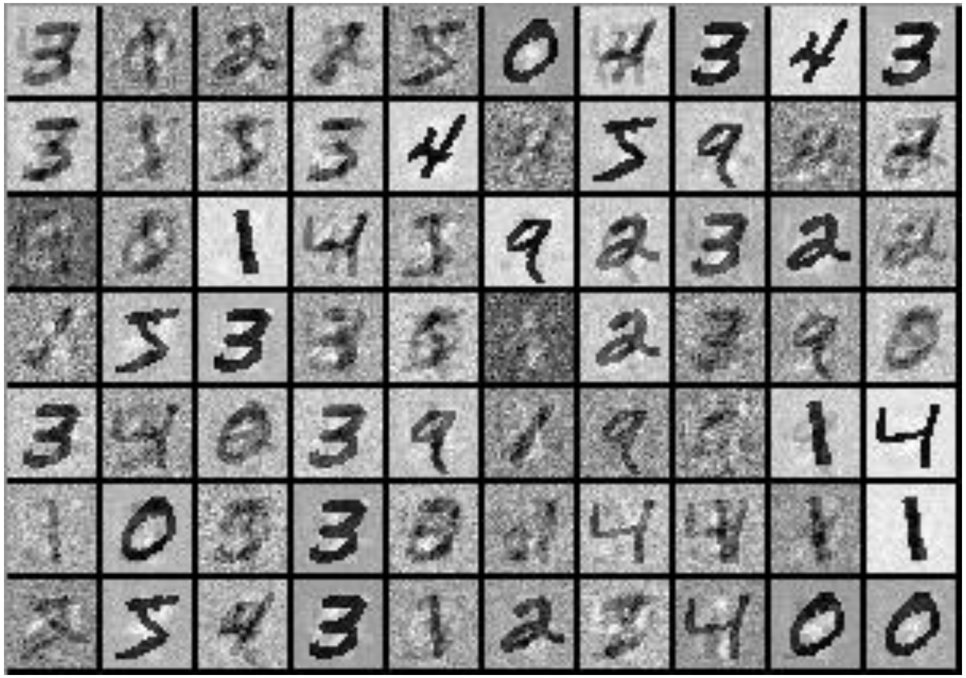
Restricted Boltzmann Machines



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

Unsupervised Learning

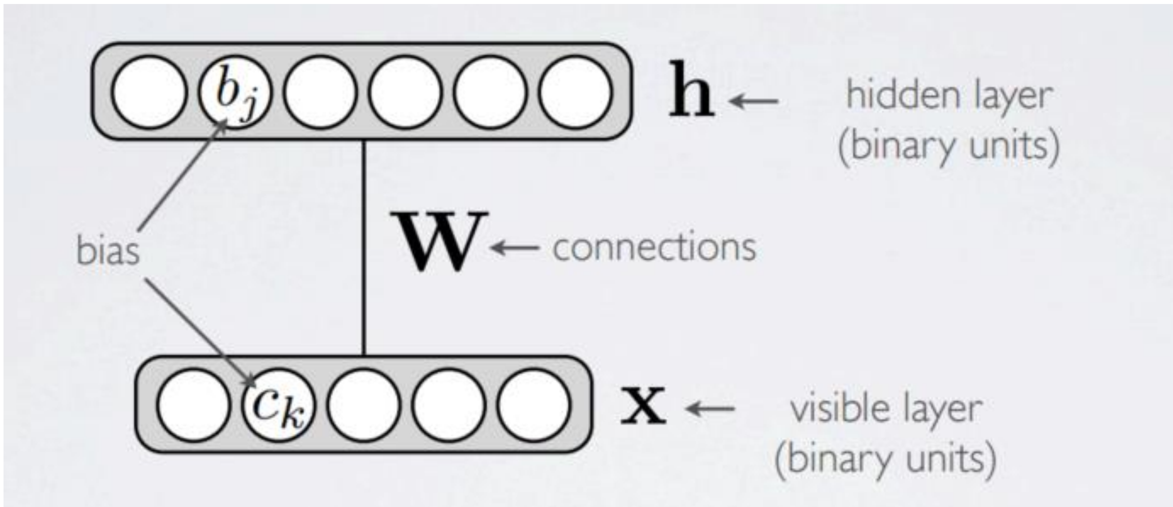
- Instead of having inputs and target outputs, we just have the inputs
- The goal is to learn something useful about the data
    - E.g., want to discover useful features of the data
        - Want to obtain the same kind of features we obtained when training e.g. AlexNet, but without having to supply labels for images
        - This is useful! Labelling images is difficult and expensive, and features can be useful for classification when there is not a lot of training data

Unsupervised Learning

- Find weights *W* s.t. $P_W(x)$ is high when *x* looks like the data in the training set, but $P_W(x)$ is low if *x* looks differently from the data in the training set
- $P_W(x)$ is the "probability of x"
  - How likely are we to observe x as a new training sample?

- In RBMs, we also use "hidden" variables h
- We imagine each sample in the training set consists of visible input x, and some hidden inputs h
- $P_W(x) = \sum_{h'} P_W(x, h')$

Restricted Boltzmann Machine (RBM)



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

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

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

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

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

RBM weights as features

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

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

- High probability => low Energy Function (E)
- Consider $W_{j,:}$
- The Energy Function is lower if $W_{j,k}$ is large when x_k is large, if $h_j = 1$
- So $W_{j,:}$ could be a template for x

RBM weights as features

- $W_{j,:}$ could be a template for x
- But that's only useful when h_j is on (i.e., = 1)
- So $P(x, h)$ will be high when
  - $h_j = 1$ for a *j* that's appropriate for the *x*
    - *Think: x is in class j*
  - The weights are such that $W_{j,:}$ is a template for x's of class *j*
- If the weights are good, $P(x)$ will be high too, since one of the terms in $P(x) = \sum_{h'} P(x, h')$ will be large

A view of the training set

- $x^{(i)} = \{1, 0, 1, 1, \ldots, 1\}$: observed. E.g., binarized image
- $h^{(i)}$=? (Unobserved.). Also a binary vector. We don't know what it is.
  - For example, a first coordinate equal to 1 might mean the sample represents the digit "0"
- It would be easy to assign probability if we knew the state of the hidden units
- The hidden layer makes it possible to assign reasonable probabilities using a relatively simple architecture

Computing $P(x)$ directly is hard

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

$2^{\dim(x)+\dim(h)}$ terms! Even computing this directly is hard

- (Note: before we just looked at the Energy Function (denominator), but that was just for intuition)
- Even computing P(x) is hard. Maximizing it with respect to W is also hard

Gibbs Sampling

- It turns out that it's possible to compute $P(h|x)$ and $P(x|h)$ easily. (I.e., if we know the visible units, it's easy to compute the probability distribution for the hidden units, and vice-versa)
- $P(h|x) = \prod_j P(h_j|x)$, $P(h_j = 1|x) = \sigma(b_j + W_{j,:}x)$
- $P(x|h) = \prod_j P(x_j|h)$, $P(x_j = 1|h) = \sigma(c_j + W_{j,:}^T h)$

Proof

- $P(h_j = 1|x) = \frac{P(h_j=1,x)}{P(x)}$

$$= \frac{P(h_j = 1, x)}{P(h_j = 0, x) + P(h_j = 1, x)}$$

$$= \frac{1}{1 + P(h_j = 0, x)/P(h_j = 1, x)}$$

- $\frac{P(h_j=0,x)}{P(h_j=1,x)} = \frac{\sum_{h'_j=0} \exp(-E(x,h'))/Z}{\sum_{h'_j=1} \exp(-E(x,h'))/Z} = \frac{\sum_{h'} \exp(\dots)}{\sum_{h'} \exp(\dots W_j x + b_j)}$

$$= \exp(-W_{j,:}x - b_j)$$

- So $P(h_j = 1|x) = \sigma(W_{j,:}x + b_j)$

$$\dots = -\sum_{j' \neq j} \sum_k W_{j'k} h_{j'} x_k - \sum_k c_k x_k - \sum_{j' \neq j} b_{j'} h_{j'}$$

Proof (cont'd, first two lines important)

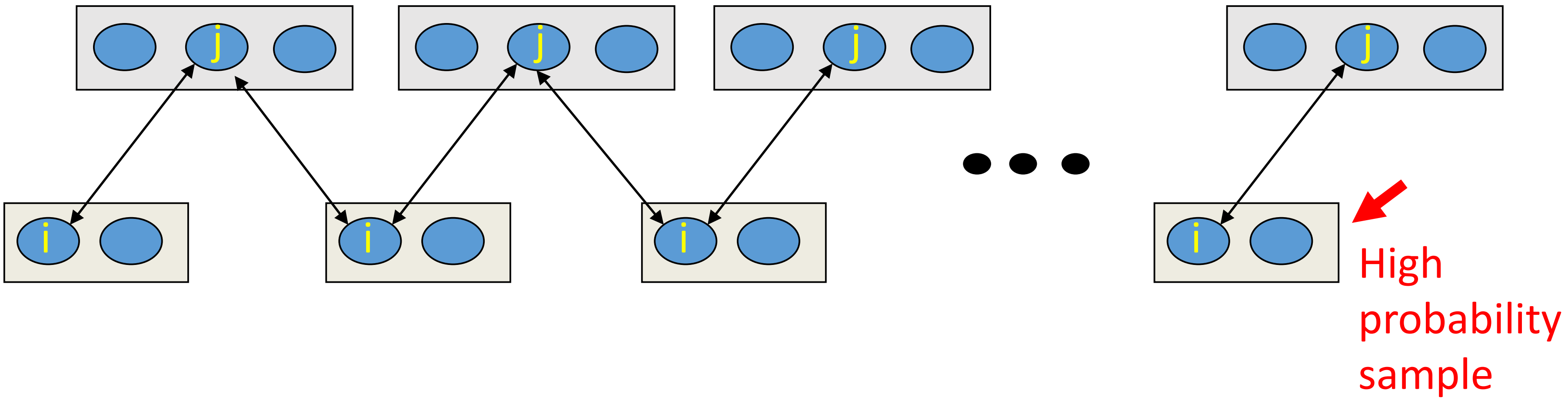
$$
\begin{aligned}
p(\mathbf{h}|\mathbf{x}) &= p(\mathbf{x},\mathbf{h}) / \sum_{\mathbf{h}'} p(\mathbf{x},\mathbf{h}') \\
&= \frac{\exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \cancel{\mathbf{c}^\top \mathbf{x}} + \mathbf{b}^\top \mathbf{h}) / \cancel{Z}}{\sum_{\mathbf{h}' \in \{0,1\}^H} \exp(\mathbf{h}'^\top \mathbf{W}\mathbf{x} + \cancel{\mathbf{c}^\top \mathbf{x}} + \mathbf{b}^\top \mathbf{h}') / \cancel{Z}} \\
&= \frac{\exp(\sum_j h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{\sum_{h'_1 \in \{0,1\}} \cdots \sum_{h'_H \in \{0,1\}} \exp(\sum_j h'_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h'_j)} \\
&= \frac{\prod_j \exp(h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{\sum_{h'_1 \in \{0,1\}} \cdots \sum_{h'_H \in \{0,1\}} \prod_j \exp(h'_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h'_j)} \\
&= \frac{\prod_j \exp(h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{\left(\sum_{h'_1 \in \{0,1\}} \exp(h'_1 \mathbf{W}_{1\cdot}\mathbf{x} + b_1 h'_1)\right) \dots \left(\sum_{h'_H \in \{0,1\}} \exp(h'_H \mathbf{W}_{H\cdot}\mathbf{x} + b_H h'_H)\right)} \\
&= \frac{\prod_j \exp(h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{\prod_j \left(\sum_{h'_j \in \{0,1\}} \exp(h'_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h'_j)\right)} \\
&= \frac{\prod_j \exp(h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{\prod_j (1 + \exp(b_j + \mathbf{W}_{j\cdot}\mathbf{x}))} \\
&= \prod_j \frac{\exp(h_j \mathbf{W}_{j\cdot}\mathbf{x} + b_j h_j)}{1 + \exp(b_j + \mathbf{W}_{j\cdot}\mathbf{x})} \\
&= \prod_j p(h_j|\mathbf{x})
\end{aligned}
$$

Gibbs Sampling for RBM (known weights)

- Initialize the x
- Sample the h given the x (Easy! We worked out the distribution)
- Sample the x given the h (Easy!)
- Repeat

- This allows us to see what kind of data the RBM is modelling (i.e., assigning high probability to)

Sampling from an RBM



If we wait long enough, the visible samples will be sampled according to the probability distribution of the RBM, since we are performing Gibbs sampling