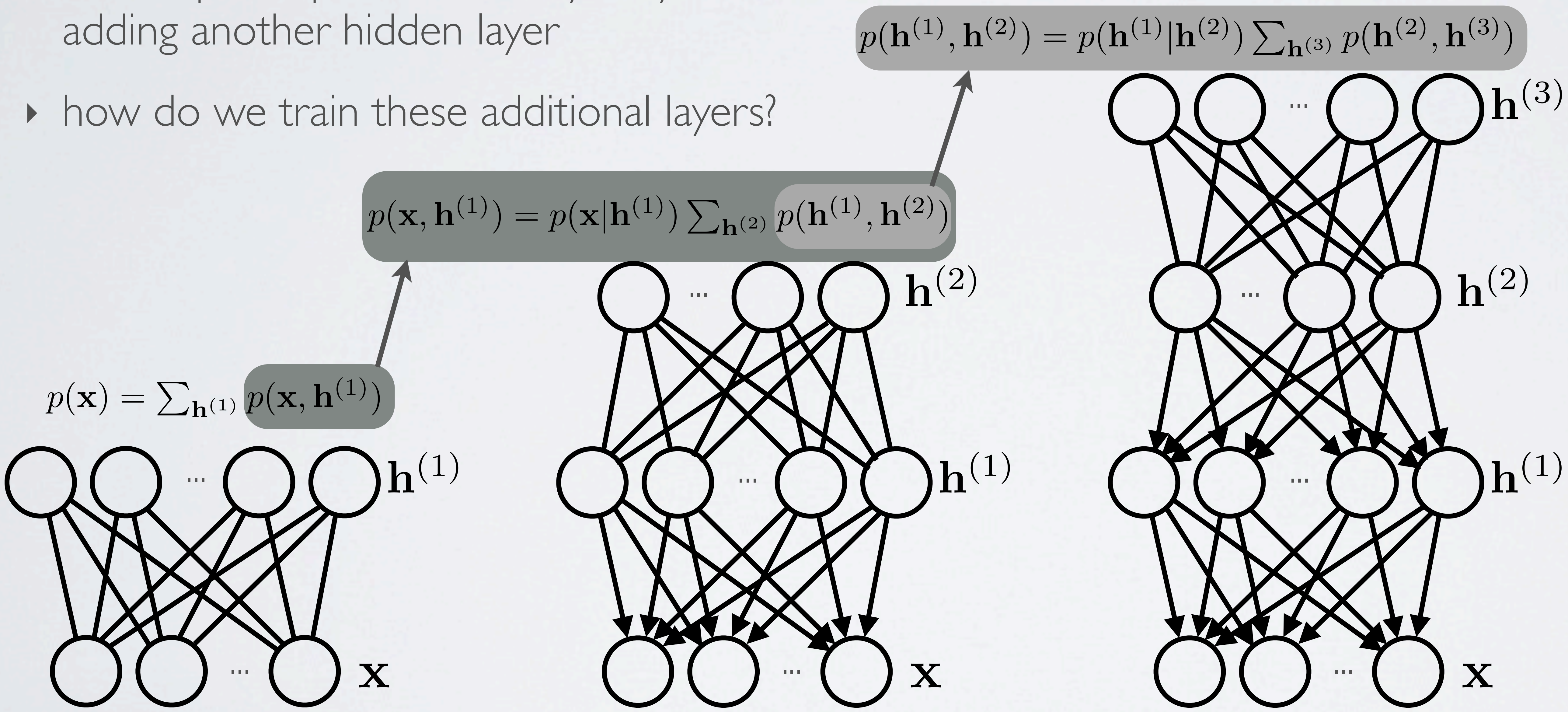
Neural networks

Deep learning - DBN pretraining

DEEP BELIEF NETWORK

Topics: deep belief network

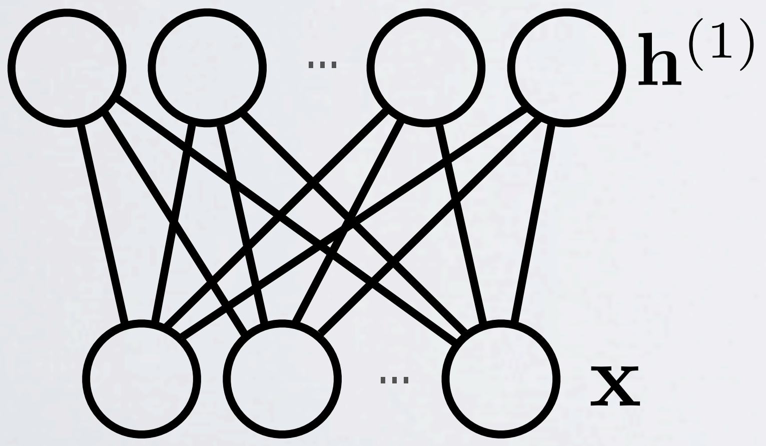
- This is where the RBM stacking procedure comes from
  - idea: improve prior on last layer by adding another hidden layer
  - how do we train these additional layers?

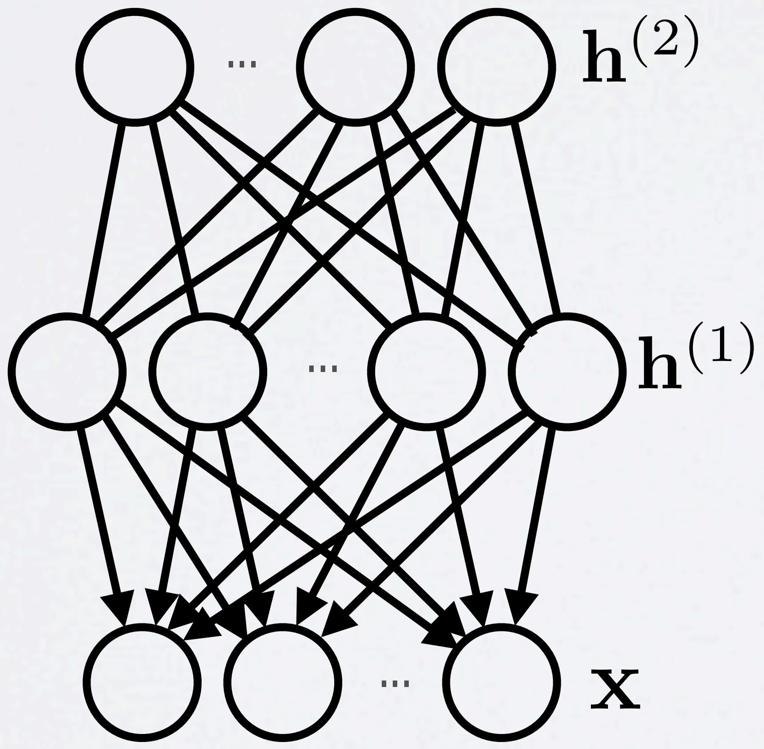


DEEP BELIEF NETWORK

Topics: deep belief network

- This is where the RBM stacking procedure comes from
  - idea: improve prior on last layer by adding another hidden layer
  - how do we train these additional layers?





DEEP BELIEF NETWORK

Topics: variational bound

- This is called a variational bound

$$\log p(\mathbf{x}) \geq \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{h}^{(1)}) - \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log q(\mathbf{h}^{(1)}|\mathbf{x})$$

  - if $q(\mathbf{h}^{(1)}|\mathbf{x})$ is equal to the true conditional $p(\mathbf{h}^{(1)}|\mathbf{x})$, then we have an equality
  - the more $q(\mathbf{h}^{(1)}|\mathbf{x})$ is different from $p(\mathbf{h}^{(1)}|\mathbf{x})$ the less tight the bound is
  - in fact, the difference between the left and right terms is the KL divergence between $q(\mathbf{h}^{(1)}|\mathbf{x})$ and $p(\mathbf{h}^{(1)}|\mathbf{x})$:

$$\mathrm{KL}(q||p) = \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log \left(\frac{q(\mathbf{h}^{(1)}|\mathbf{x})}{p(\mathbf{h}^{(1)}|\mathbf{x})} \right)$$

DEEP BELIEF NETWORK

Topics: variational bound

- This is called a variational bound

$$\begin{aligned}\log p(\mathbf{x}) \quad \geq \quad & \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \left(\log p(\mathbf{x}|\mathbf{h}^{(1)}) + \log p(\mathbf{h}^{(1)})\right) \\ & - \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log q(\mathbf{h}^{(1)}|\mathbf{x})\end{aligned}$$

  - for a single hidden layer DBN (i.e. an RBM), both $p(\mathbf{x}|\mathbf{h}^{(1)})$ and $p(\mathbf{h}^{(1)})$ depend on the parameters of the first layer
  - when adding a second layer, we model $p(\mathbf{h}^{(1)})$ using a separate set of parameters
    - they are the parameters of the RBM involving $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$
    - $p(\mathbf{h}^{(1)})$ is now the marginalization of the second hidden layer $p(\mathbf{h}^{(1)}) = \sum_{\mathbf{h}^{(2)}} p(\mathbf{h}^{(1)}, \mathbf{h}^{(2)})$

DEEP BELIEF NETWORK

Topics: variational bound

- This is called a variational bound

adding 2nd layer means untying the parameters in

$$\log p(\mathbf{x}) \geq \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \left(\log p(\mathbf{x}|\mathbf{h}^{(1)}) + \log p(\mathbf{h}^{(1)})\right) - \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log q(\mathbf{h}^{(1)}|\mathbf{x})$$

  - we can train the parameters of the new second layer by maximizing the bound
    - this is equivalent to minimizing the following, since the other terms are constant:

$$-\sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log p(\mathbf{h}^{(1)})$$

    - this is like training an RBM on data generated from $q(\mathbf{h}^{(1)}|\mathbf{x})$!

DEEP BELIEF NETWORK

Topics: variational bound

- This is called a variational bound

adding 2nd layer means untying the parameters in

$$\log p(\mathbf{x}) \geq \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \left(\log p(\mathbf{x}|\mathbf{h}^{(1)}) + \log p(\mathbf{h}^{(1)})\right)$$
$$- \sum_{\mathbf{h}^{(1)}} q(\mathbf{h}^{(1)}|\mathbf{x}) \log q(\mathbf{h}^{(1)}|\mathbf{x})$$

- for $q(\mathbf{h}^{(1)}|\mathbf{x})$ we use the posterior of the first layer RBM
  - equivalent to a feed-forward (sigmoidal) layer, followed by sampling
- by initializing the weights of the second layer RBM as the transpose of the first layer weights, the bound is initially tight
  - a 2 layer DBN with tied weights is equivalent to a 1 layer RBM

DEEP BELIEF NETWORK

Topics: variational bound

- This process of adding layers can be repeated recursively
  - we obtain the greedy layer-wise pre-training procedure for neural networks
- We now see that this procedure corresponds to maximizing a bound on the likelihood of the data in a DBN
  - in theory, if our approximation $q(\mathbf{h}^{(1)}|\mathbf{x})$ is very far from the true posterior, the bound might be very loose
  - this only means we might not be improving the true likelihood
  - we might still be extracting better features!
- Fine-tuning is done by the Up-Down algorithm
  - A fast learning algorithm for deep belief nets.
    Hinton, Teh, Osindero, 2006.