Here we have a graphical prob. dist.

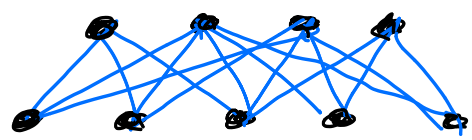
$P_\lambda(\vec{v}, \vec{h})$ ie. a joint distribution

"learning" is adjusting parameters λ so that

$$P_\lambda(\vec{v}) = \sum_{\vec{h}} P_\lambda(\vec{v}, \vec{h}) \simeq P(\vec{v})$$

Restricted Boltzmann Machine (Hinton, Smolénsk '86)

n of $\vec{h}$



$\vec{v}$ m of these

no intra-layer couplings

$$E_\lambda = -\sum_{ij} W_{ij} v_i h_j - \sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} c_j h_j$$

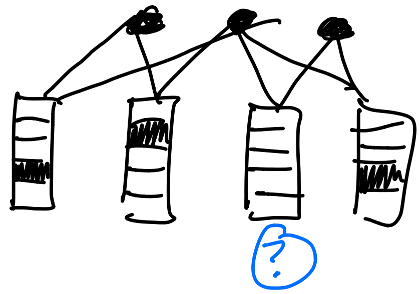
$v_i, h_j = 0, 1$

parameters are $\lambda = (W, \vec{b}, \vec{c})$

and
$$P_\lambda(\vec{v}, \vec{h}) = \frac{1}{Z_\lambda} e^{-E_\lambda(\vec{v}, \vec{h})}$$

Industry applications

- Netflix competition: collaborative filtering



- topic modelling: interpreting latent units as "semantic structure"

- Unsupervised "pre-training" for deep learning (pre-2012)

Now: Training an RBM: $\mathcal{D} = \{\vec{v}\}$ drawn from an unknown $P(\vec{v})$ adjusting λ so that

$$p_\lambda(\vec{v}) = \sum_h p(\vec{v}, \vec{h}) \simeq P(\vec{v})$$

Next: Sampling: after training, generate new $\vec{v}, \vec{h}$ drawn from $p(\vec{v}, \vec{h})$

and calculate $\langle O \rangle_{p_\lambda(\vec{v},\vec{h})} = \frac{1}{Z} \sum_{\vec{v}} \sum_{\vec{h}} O\, p_\lambda(\vec{v},\vec{h})$

note: if we define $O = O_v$ (e.g. E_v)

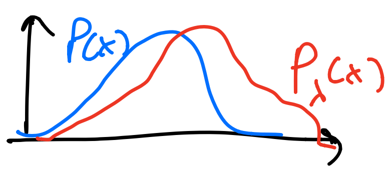
$$\langle O_v \rangle_{p_\lambda(\vec{v},\vec{h})} = \frac{1}{Z} \sum_{v} O_v \underbrace{\sum_{\vec{h}} p_\lambda(\vec{v},\vec{h})}_{p_\lambda(\vec{v}) \approx P(\vec{v})}$$

note

$$Z = \sum_{v} \sum_{h} p_\lambda(\vec{v},\vec{h}) = \sum_{\vec{v}} p_\lambda(\vec{v})$$

TRAINING: Define a cost function

$$KL(P \| p_\lambda) = \sum_{\vec{x}} P(\vec{x}) \log \frac{P(\vec{x})}{p_\lambda(\vec{x})}$$



$$KL(P \| p_\lambda) = \underbrace{\sum_{x} P(\vec{x}) \log P(\vec{x})}_{\text{entropy of } P \equiv -H_D} - \underbrace{\sum_{x} P(\vec{x}) \log p_\lambda(\vec{x})}_{\text{depends on } \lambda}$$

recall $D = \{\vec{x}\}$, $P_{data}(\vec{x}) = \frac{1}{\|D\|} \sum_{\vec{x}_k \in D} \delta_{\vec{x},\vec{x}_k}$

$$KL(P \| p_\lambda) \simeq \underbrace{-H_D - \frac{1}{\|D\|} \sum_{\vec{x} \in D} \log p_\lambda(\vec{x})}_{\text{use this for the cost fn } C_\lambda}$$

Now: use gradient descent

$$\lambda' = \lambda - \eta \nabla_\lambda C_\lambda$$

Let's consider a single training example $\vec{v}$

$$\log p_\lambda(\vec{v}) = \log \frac{1}{Z} \sum_h e^{-E(v,h)}$$

$$= \log \sum_h e^{-E} - \log \sum_{vh} e^{-E}$$

The gradient:

$$\frac{\partial \log p_\lambda}{\partial \lambda} = \frac{\partial}{\partial \lambda}\left(\log \sum_h e^{-E}\right) - \frac{\partial}{\partial \lambda}\left(\log \sum_{vh} e^{-E}\right)$$

$$= -\frac{1}{\sum_h e^{-E}} \sum_h e^{-E} \cdot \frac{\partial E}{\partial \lambda} + \frac{1}{\sum_{vh} e^{-E}} \sum_{vh} e^{-E} \frac{\partial E}{\partial \lambda}$$

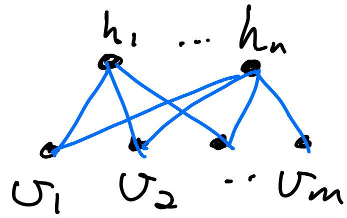
using $p(h|v) = \dfrac{p(v,h)}{p(v)} = \dfrac{\frac{1}{Z} e^{-E}}{\frac{1}{Z} \sum_h e^{-E}} = \dfrac{e^{-E}}{\sum_h e^{-F}}$

$$\boxed{\frac{\partial \log p_\lambda}{\partial \lambda} = -\sum_h p(h|v)\frac{\partial E}{\partial \lambda} + \sum_{v h} p(v,h)\frac{\partial E}{\partial \lambda}}$$

*

and recall $\frac{\partial E}{\partial W_{ij}} = -v_i h_j$ etc

Recall the RBM graph



then

$$p(\vec{h}|\vec{v}) = \prod_{i=1}^{n} p(h_i|\vec{v})$$

$$p(\vec{v}|\vec{h}) = \prod_{i=1}^{m} p(v_i|\vec{h})$$

This gives "Block" Gibbs sampling.

Simple to calculate an analytical expression

$$p(h_j = 1 | \vec{v}) = \sigma\left(\sum_{i=1}^{m} W_{ij} v_i + c_i\right)$$

$$p(v_i = 1 | \vec{h}) = \sigma\left(\sum_{j=1}^{n} W_{ij} h_j + b_j\right)$$

$\uparrow$ sigmoid function

Back to ⊛, let's calculate first term for $\lambda = W_{ij}$

$$\sum_{\vec{h}} p(\vec{h}|\vec{v}) \frac{\partial E}{\partial W_{ij}} = \sum_{h} \underbrace{p(\vec{h}|\vec{v})}_{p(h_i|\vec{v})\,p(\vec{h}_{-i}|\vec{v})} h_i v_j$$

$$= \sum_{h_i} p(h_i|\vec{v}) h_i v_j \sum_{\vec{h}_{-i}} p(\vec{h}_{-i}|\vec{v})$$

trace $\quad \sum_{\{\vec{h}_{-i}=0,1\}}$

ie $\left(p(h_1=0|\vec{v}) + p(h_1=1|\vec{v})\right)\cdot()$

$$= \sum_{h_i} p(h_i|\vec{v}) h_i v_j$$

$$= 0 + p(h_i = 1|\vec{v}) v_j$$

$$= \sigma\left(\sum_{k=1}^{m} W_{ik} v_k + c_k\right) v_j$$

Next, look at second term in ⊛

$$\sum_{\vec{v},\vec{h}} p(\vec{v},\vec{h}) \frac{\partial E}{\partial \lambda} = \sum_{\vec{v}} p(\vec{v}) \sum_{\vec{h}} p(\vec{h}|\vec{v}) \frac{\partial E}{\partial \lambda}$$

Inner sum is tractable, but $\sum_{\vec{v}}$ is a trace over 2^m states, and is not.

How is this sum handled? With something called "contrastive divergence"
just fudge the estimator with a very <u>short</u> Markov Chain:

$$\vec{v}_0 \to \vec{h}_0 \to \vec{v}_1 \to \vec{h}_1 \cdots \vec{v}_k$$

get a <u>rough</u> estimate with a small ($k = 1 \cdots 5$) number of steps in the Markov chain.

For completeness, remember we want to use the entire training set $\{\vec{v}\} = \mathcal{D}$

$$C_\lambda = -\frac{1}{\|\mathcal{D}\|} \sum_{\vec{v} \in \mathcal{D}} \log P_\lambda(\vec{v})$$

$$\nabla_\lambda C_\lambda = -\frac{1}{\|\mathcal{D}\|} \sum_{v \in \mathcal{D}} \frac{\partial}{\partial \lambda} \log P_\lambda(\vec{v})$$

$$\frac{\partial \log P_\lambda(\vec{v})}{\partial \lambda} = -\sum_{\vec{h}} p(\vec{h} | \vec{v}) \frac{\partial E(v,h)}{\partial \lambda} + \sum_{h} p(\vec{h} | \vec{v}^k) \frac{\partial E(v^k, h)}{\partial \lambda}$$