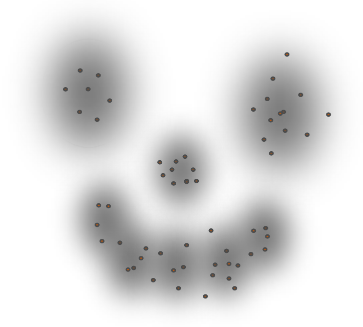
Modelling Random Distributions - Hierarchial Graphical Models

Unsupervised Machine Learning $\rightarrow$ Model a distribution from given data

$$\{\mathbf{x}^i\}_{i\in I} \rightarrow P(\mathbf{x})$$



Maximize the log-likelihood:

$$LL(\theta) = \sum_{\mathbf{x}\in Data} \log(P(\mathbf{x}|\theta))$$

There are in general two modes to operate on such a given distribution:

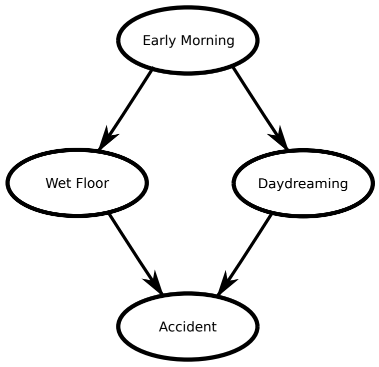
- Sampling:
$$P(\mathbf{x}) \rightarrow \mathbf{x}_{\sim P(\mathbf{x})}$$
- Likelihood Evaluation:
$$\mathbf{x} \rightarrow P(\mathbf{x})$$

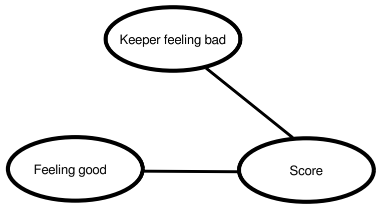
Ideally both is feasible using the specific model.

Simplification of a probability distribution $\rightarrow$ Graphical Models

Directed Models vs Undirected Models: Bayesian Networks vs Markov Random Fields

Using conditional independence of random variables (independence of single data dimensions)





This corresponds to modeling the distribution as a product of conditional distributions vs a product of random factors:

$$P(ac, wf, dd, ac) = P(ac|wf, dd)P(dd|em)P(wf|em)P(em)$$

vs.

$$P(kfb, fg, sc) = \phi(fg, sc)\phi(kfb, sc)$$

Both incorporate some (conditional) independencies of random variables (data dimensions).

In general the distributions look like:

- Bayesian Network:
$$P(\mathbf{x}) = \prod_i P(x_i|x_i^{parents})$$
- Markov Random Field:
$$P(\mathbf{x}) = \prod_{c \in Cliques} \phi(x_1^c \dots x_{cn}^c)$$

Boltzmann machines - model MRF logits $\leftrightarrow$ negative interaction energies

$$P(\mathbf{x}) = \frac{1}{Z} exp(-\frac{1}{T}\sum_{ij} x_i W_{ij} x_j)$$

Restricting the energy to quadratic terms is already a big simplification from $\mathcal{O}(d^N)$ to $\mathcal{O}(N^2 d^2)$ parameters. So larger scale interactions are lost.

Two modes:

- Likelihood evaluation: Hard to find partition sum $Z = \sum_{\mathbf{x}} exp(-\frac{1}{T}\mathbf{x}^T\mathbf{W}\mathbf{x})$
- Sampling: -> Gibbs sampling (Markov Chain Monte Carlo)

Gibbs sampling

Algorithm:

1. Take a random state $\mathbf{x}$ as initial state
2. Pick a random state index i
3. Draw a single variable x_i from the conditional distribution $P(x_i'|\mathbf{x}_{\setminus i})$ given all current other variables
4. Replacing the modified variable leads the new state

Repeat this for a long time, the distribution then converges to the correct $P(\mathbf{x})$.

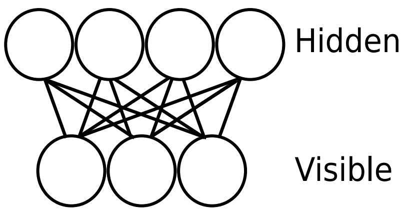
Restricted Boltzmann machines

Separation into visible and hidden layer. Only interactions between those layers are allowed. The marginal distribution on the visible units can then model richer interactions.

$$P(\mathbf{v},\mathbf{h}) = \frac{1}{Z} exp(-\mathbf{v^T W h} - \mathbf{a^T v} - \mathbf{b^T h})$$
$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v},\mathbf{h})$$

Also the conditional distributions are simple to evaluate (including normalization):

$$P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h}) = \prod_i \sigma(\sum_j v_i W_i j h_j + a_i)$$
$$P(\mathbf{h}|\mathbf{v}) = \prod_i P(h_i|\mathbf{v}) = \prod_i \sigma(\sum_j v_j W_j i h_i + b_i)$$



Sampling

Reminder:

Energy field model probability:

$$p(v,h;\theta) = \frac{1}{Z(\theta)} exp(-E(v,h;\theta))$$
$$Z(\theta) = \sum_x exp(-E(v,h;\theta))$$

Bernoulli RBM

$$E_{RBM} = -v^T W h - b^T v - a^T h$$
$$p(h_j = 1|v) = \sigma(\sum_i W_{ij} v_i + a_j)$$
$$p(v_i = 1|h) = \sigma(\sum_j W_{ij} h_j + b_i)$$

Gaussian Bernoulli RBM

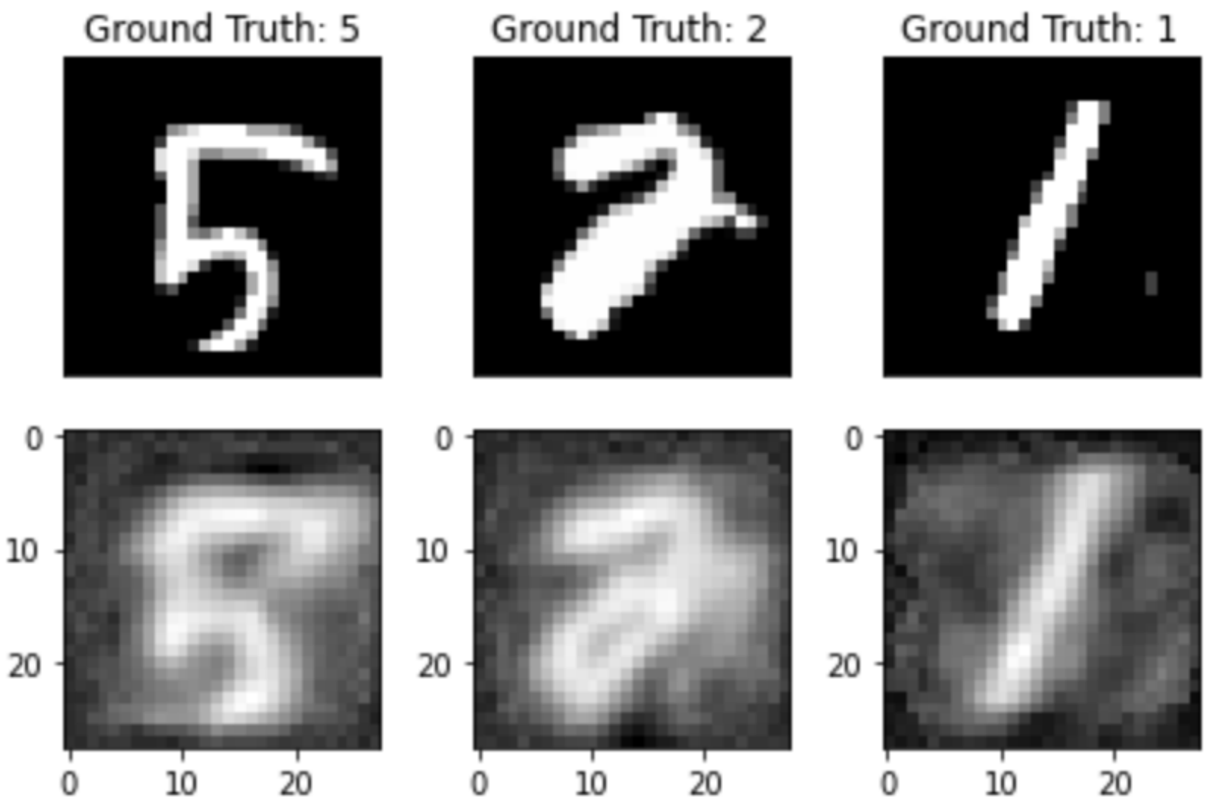
$$E_{gRBM} = \sum_i \frac{(v_i - a_i)}{2\sigma_i^2} - \sum_j b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij}$$

$$p(v|h) = \mathcal{N}(\sum_{i,j} h_j W_{ij} + b_i, \sigma^2)$$

With unit variance:

$$\langle v_i | h \rangle = \sum_j W_{ij} h_j + b_i$$

```
In [1]: def reconstruct(self, data, n=1, std = 1, mean = 1, gaussian = False):
            for i in range(n):
                hcs = self.get_hidden_sample(data)
                if gaussian:
                  data = self.get_visible_mean(hidden = hcs, gaussian = True)
                else:
                  data = self.get_visible_sample(hcs)
                return data
```



Contrastive divergence (CD_k)

Learning

- Iid. sample $\chi = \{v_n\}_{n=1}^N$ from a data distribution.
- Find model with parameters θ, such that it maximizes the log likelihood of measuring θ: $L(\chi;\theta)$.
- Use gradient ascent learning for θ:
$$\theta^{\tau+1} = \theta^{\tau} + \eta \frac{\partial L(\theta;\chi)}{\partial \theta}|_{\theta^{\tau}}$$
### Gradient ascent
$$L(\chi;\theta) = \frac{1}{N}\sum_{n=1}^{N} log\, p(v_n;\theta) = \langle log\, p(v;\theta)\rangle_0 = -\langle E(v;\theta)\rangle_0 - log\, Z(\theta)$$
with $p_0(v) = \frac{1}{N}\sum_{n=1}^{N}\delta(v-v_n)$ and $p_\infty(v) = \sum_h p(v,h;\theta)_{Model}$ This leads to:\
$$\frac{\partial L(\theta;\chi)}{\partial \theta} = -\langle\frac{\partial E(v;\theta)}{\partial \theta}\rangle_{0,\, positive\ phase} + \langle\frac{\partial E(v;\theta)}{\partial \theta}\rangle_{\infty,\, negative\ phase}$$

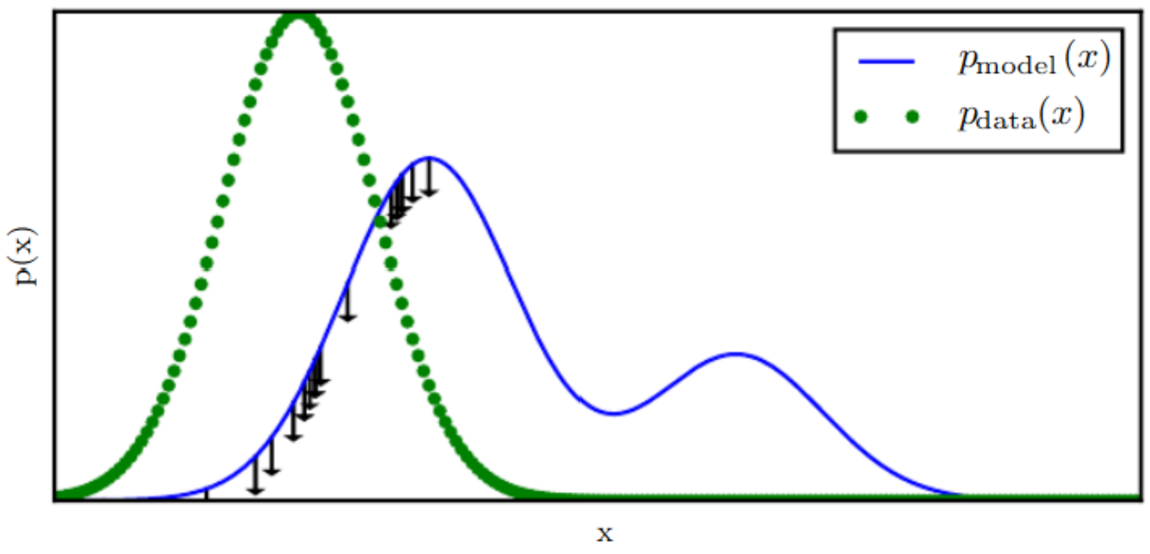
Thus, we get:
$$\frac{\partial L(v;\theta)}{\partial W} = \langle vh^T\rangle_0 - \langle vh^T\rangle_\infty$$
$$\frac{\partial L(v;\theta)}{\partial a} = \langle h\rangle_0 - \langle h\rangle_\infty$$
$$\frac{\partial L(v;\theta)}{\partial b} = \langle v\rangle_0 - \langle v\rangle_\infty$$

Problem: finding $\langle\rangle_\infty$ requires finding Z, but Z is exponentially large in number of dimensions of v and h.

General idea

Approximate the true expectation of the model $\langle\rangle_\infty$ with the probability distribution after k Gibbs steps $\langle\rangle_k$, starting with χ. The gradient steps:
$$\frac{\partial L(v;\theta)}{\partial W} = \langle vh^T\rangle_0 - \langle vh^T\rangle_k$$
$$\frac{\partial L(v;\theta)}{\partial a} = \langle h\rangle_0 - \langle h\rangle_k$$
$$\frac{\partial L(v;\theta)}{\partial b} = \langle v\rangle_0 - \langle v\rangle_k$$
New gradient follows approximately gradient of KL-divergence $CD_n = KL(p_0||p_\infty) - KL(p_n||p_\infty)$\ Advantage: \
It shows good results for models with feasible complexity.

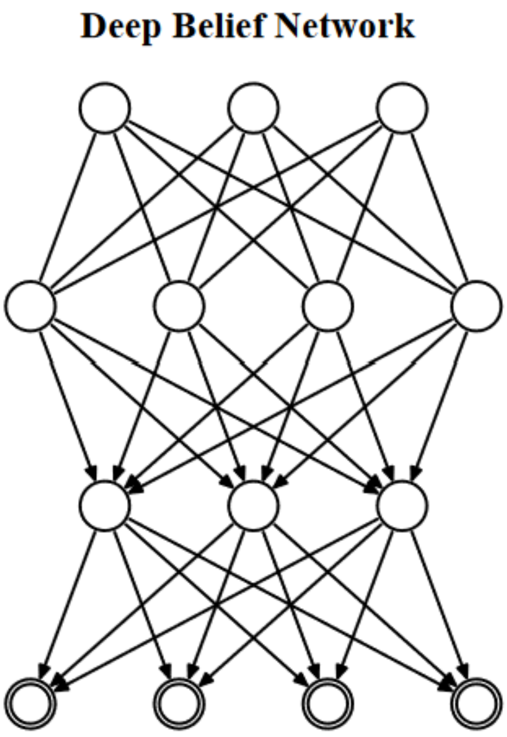
Problem with contrastive divergence



Graph illustrates, how a spurious mode is not caught in the negative phase of CD_1 https://www.deeplearningbook.org/contents/partition.html (https://www.deeplearningbook.org/contents/partition.html) \ The more complex the model, the more likely/pronounced are spurious modes.

Contrastive persistent divergence

Instead of initializing the MCMC chain of the first gibbs step with data, it is initialized with the last state of the model. It only works when the gradient step is small enough, that the equilibrium state of the last model is close the equlibrium state of the next one.



DBN

- N-2 directed layers
- 2 undirected layers at the top

For 3-layer DBN with v, h^1, h^2 neurons, $W^1, W^2 \in \theta$ and approximate distribution $Q(h^1|v)$:

$$L(v;\theta) \geq \sum_{h^1} Q(h^1|v)L(h^1;W^2) + R(v,h^1,h^2,W^1,Q)$$

Training

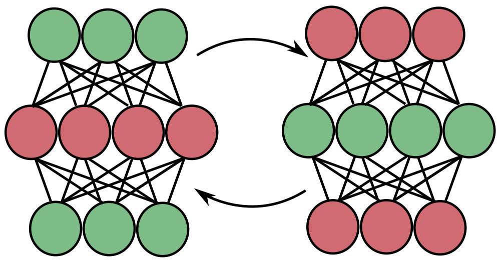
At initialisation choose $Q = p(h^1|v;W^1)$. Then the 3-layer DBN acts exactly like the RBM

- by freezing W^1 and maximizing $L(h^1;W^2)$, we are increasing $L(v;\theta)$.
- maximizing $L(h^1;W^2)$ is done by training an RBM with input h^1 and hidden units h^2
- This process can be repeated for N total layers. ## Inference
- Initialize the top two layers randomly\
- run k gibbs step to get a sample according to probability function of the top RBM\
- pass the activations of the unit layers wise down, according to the learned parameters of the RBMs\
- The activation probability of the lowest RBM should be according to the probability distribution of the examples

Deep Boltzmann machines

One can also expand Restricted Boltzmann machines to multiple layers. In this case intermediate layers depend on both the upper and lower layer.

For training a Deep Boltzmann machine one can use an alternating scheme between even and odd layer updates.



Sources: \ [1] https://www.deeplearningbook.org/contents/generative_models.html (https://www.deeplearningbook.org/contents/generative_models.html) \ [2] https://www.cs.toronto.edu/~hinton/absps/cdmiguel.pdf (https://www.cs.toronto.edu/~hinton/absps/cdmiguel.pdf) \ [3] https://tspace.library.utoronto.ca/bitstream/1807/19226/3/Salakhutdinov_Ruslan_R_200910_PhD_thesis.pdf (https://tspace.library.utoronto.ca/bitstream/1807/19226/3/Salakhutdinov_Ruslan_R_200910_PhD_thesis.pdf)