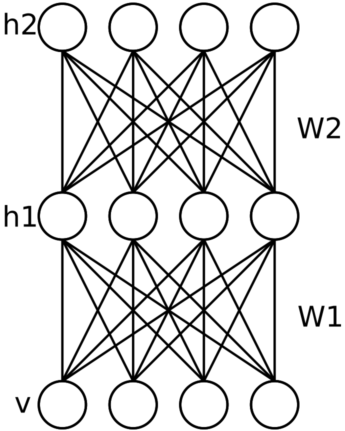
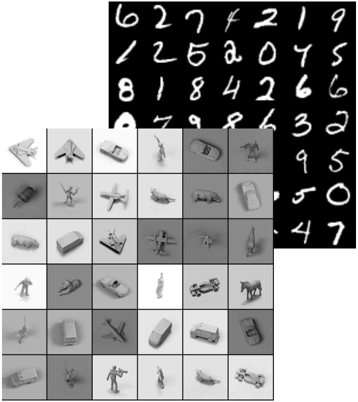
Deep Boltzmann Machines

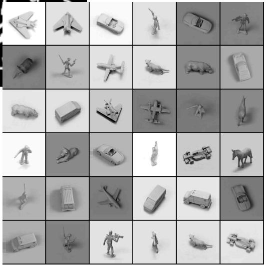
Machine Learning Seminar

TECHNISCHE UNIVERSITÄT DARMSTADT





Motivation



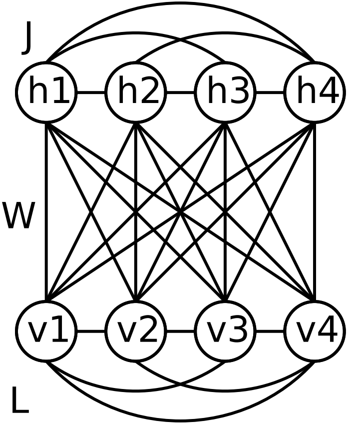
- Unsupervised Pretraining helps
- Structure of data not to be found in its labels
- but in the data itself!

Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

1. General Boltzmann Machines
2. Restricted Boltzmann Machines
3. Deep Boltzmann Machines
4. Neural Networks
5. Experimental Results
6. Conclusions

Boltzmann Machines
Definition



$$\mathbf{v} \in \{0,1\}^D$$

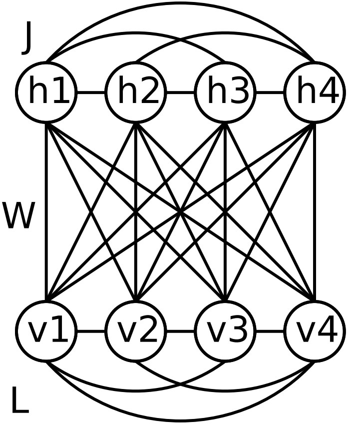
$$\mathbf{h} \in \{0,1\}^P$$

$$E(\mathbf{v},\mathbf{h};\theta) = -\frac{1}{2}\mathbf{v}^T\mathbf{L}\mathbf{v} - \frac{1}{2}\mathbf{h}^T\mathbf{J}\mathbf{h} - \mathbf{v}^T\mathbf{W}\mathbf{h}$$

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

$$\theta = \{\mathbf{W},\mathbf{L},\mathbf{J}\}$$

Boltzmann Machines
Gradients

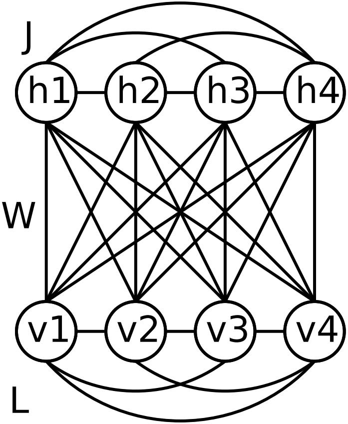


- Maximum Likelihood leads to the following gradients:

$$\Delta \mathbf{W} = \alpha(E_{P_{data}}[\mathbf{vh}^T] - E_{P_{model}}[\mathbf{vh}^T])$$

$$\Delta \mathbf{L} = \alpha(E_{P_{data}}[\mathbf{vv}^T] - E_{P_{model}}[\mathbf{vv}^T])$$

$$\Delta \mathbf{J} = \alpha(E_{P_{data}}[\mathbf{hh}^T] - E_{P_{model}}[\mathbf{hh}^T])$$



- Maximum Likelihood leads to the following gradients:

$$\Delta \mathbf{W} = \alpha(E_{P_{data}}[\mathbf{v}\mathbf{h}^T] - E_{P_{model}}[\mathbf{v}\mathbf{h}^T])$$

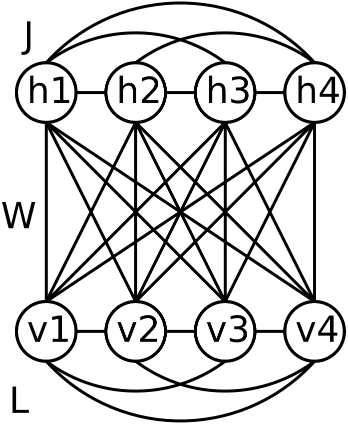
$$\Delta \mathbf{L} = \alpha(E_{P_{data}}[\mathbf{v}\mathbf{v}^T] - E_{P_{model}}[\mathbf{v}\mathbf{v}^T])$$

$$\Delta \mathbf{J} = \alpha(E_{P_{data}}[\mathbf{h}\mathbf{h}^T] - E_{P_{model}}[\mathbf{h}\mathbf{h}^T])$$

- $E_{P_{model}}[.]$ is the expectation over the distribution of the current model
  - approximated with samples

Boltzmann Machines

Gradients



- Maximum Likelihood leads to the following gradients:

$$\Delta \mathbf{W} = \alpha(E_{P_{data}}[\mathbf{v}\mathbf{h}^T] - E_{P_{model}}[\mathbf{v}\mathbf{h}^T])$$

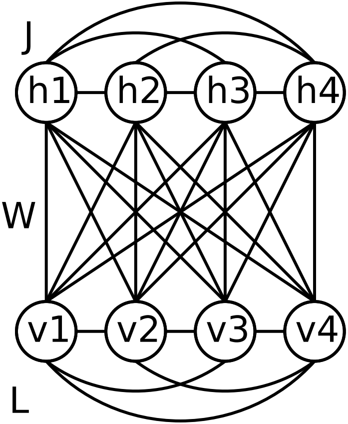
$$\Delta \mathbf{L} = \alpha(E_{P_{data}}[\mathbf{v}\mathbf{v}^T] - E_{P_{model}}[\mathbf{v}\mathbf{v}^T])$$

$$\Delta \mathbf{J} = \alpha(E_{P_{data}}[\mathbf{h}\mathbf{h}^T] - E_{P_{model}}[\mathbf{h}\mathbf{h}^T])$$

- $E_{P_{model}}[.]$ is the expectation over the distribution of the current model
  - approximated with samples
- $E_{P_{data}}[.]$ is the expectation over the completed data distribution
  - approximated e.g. with mean-field method

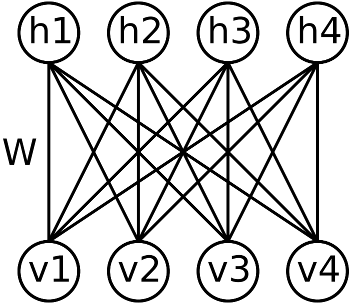
Boltzmann Machines
Difficulties

TECHNISCHE UNIVERSITÄT DARMSTADT



- cannot be trained directly
  - has to be approximated
- even inference is hard
  - e.g.: Gibbs sampling requires each node to be sampled independently
  - this takes a long time and is also an approximation

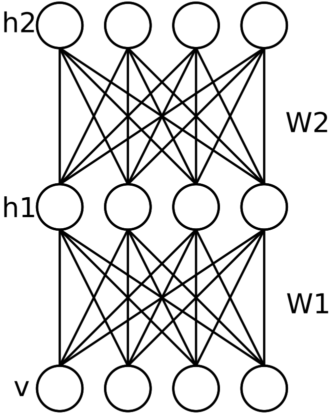
Restricted Boltzmann Machines



- most problems can be solved by setting $L = J = 0$.
- observing one layer makes the nodes of the other independent from each other.

Deep Boltzmann Machines

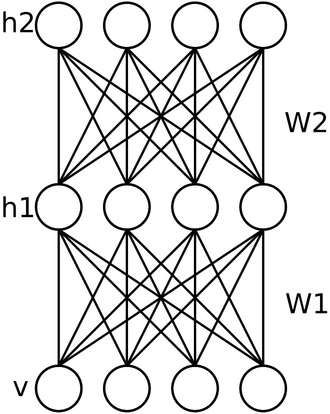
TECHNISCHE UNIVERSITÄT DARMSTADT



- Multiple RBMs stacked upon each other
- each layer captures complicated, higher-order correlations
- promising for object and speech recognition
- deals more robustly with ambigous inputs than e.g. deep belief networks
- may be trained the same way as a BM
  - is very slow
  - may get stuck in bad local optimum

Deep Boltzmann Machines

Two-Layer example



Model:

$$E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta) = -\mathbf{v}^T \mathbf{W}^1 \mathbf{h}^1 - \mathbf{h}^{1T} \mathbf{W}^2 \mathbf{h}^2$$

$$p(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}^1, \mathbf{h}^2} \exp(-E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta))$$

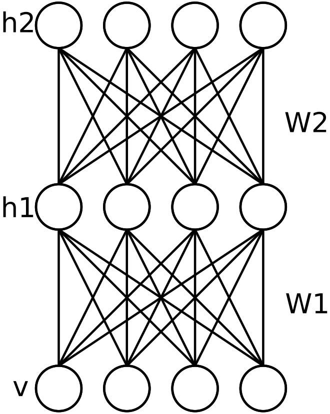
Conditional Probabilities:

$$p(h_j^1 | \mathbf{v}, \mathbf{h}^2) = \sigma(\sum_i W_{ij}^1 v_i + \sum_m W_{jm}^2 h_j^2)$$

$$p(h_m^2 | \mathbf{h}^1) = \sigma(\sum_j W_{im}^2 h_i^1)$$

$$p(v_i | \mathbf{h}^1) = \sigma(\sum_j W_{ij}^1 h_j^1)$$

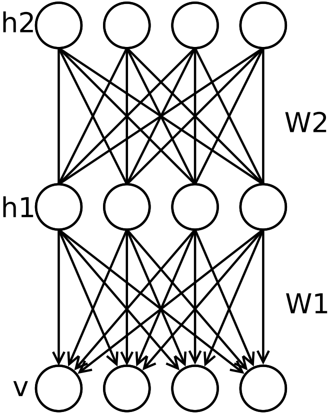
Deep Boltzmann Machines
Layer by Layer Pretraining



- Train each W_i individually by using the output of the lower RBM as input for the upper.
- The last RBM may be trained as it is.

Deep Boltzmann Machines

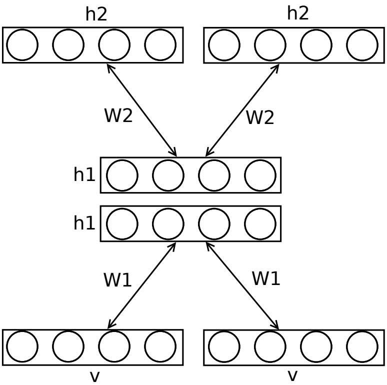
Deep Belief Networks



- Train each W_i individually by using the output of the lower RBM as input for the upper.
- The last RBM may be trained as it is.
- This results in a Deep Belief Network $p(\mathbf{v};\theta) = \sum_{\mathbf{h}^1} p(\mathbf{h}^1;\mathbf{W}^1)p(\mathbf{v}|\mathbf{h}^1;\mathbf{W}^1)$
- second RBM replaces $p(\mathbf{h}^1;\mathbf{W}^1)$ by $p(\mathbf{h}^1;\mathbf{W}^2) = \sum_{\mathbf{h}^2} p(\mathbf{h}^1,\mathbf{h}^2;\mathbf{W}^2)$
- if initialized correctly $p(\mathbf{h}^1;\mathbf{W}^2)$ is a better model

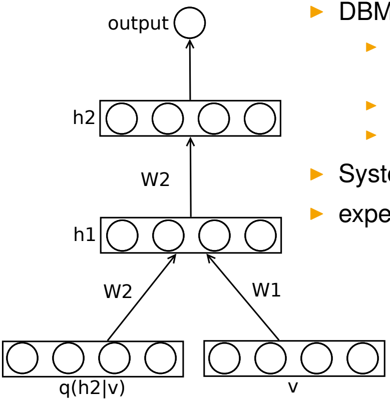
Deep Boltzmann Machines

Layer by Layer Pretraining



- modify RBMs s.th. learned weights are half
- conditionals for the bottom RBM:
  $p(h_j^1 = 1|\mathbf{v}) = \sigma(\sum_i W_{ij}^1 v_i + \sum_i W_{ij}^1 v_i)$
  $p(v_i = 1|\mathbf{h}^1) = \sigma(\sum_j W_{ij}^1 h_j^1)$
- conditionals for the top RBM:
  $p(h_j^1 = 1|\mathbf{h}^2) = \sigma(\sum_m W_{jm}^2 h_m^2 + \sum_m W_{jm}^2 h_m^2)$
  $p(h_m^2 = 1|\mathbf{h}^1) = \sigma(\sum_j W_{jm}^2 h_j^1)$
- after combining the distribution over $\mathbf{h}^1$ is:
  $p(h_j^1 = 1|\mathbf{v}, \mathbf{h}^2) = \sigma(\sum_i W_{ij}^1 v_i + \sum_m W_{jm}^2 h_m^2)$
- good initialization for mean-field method

Neural Networks



- DBM can be used to initialize a neural network
  - Weights of DBM become the weights of neural network
  - $q(\mathbf{h}|\mathbf{v})$ is the mean-field distribution of the posterior
  - neural network may be trained with backpropagation
- System may decide which of the input layers to use
- experiments show the use of both

Experimental Results

MNIST

Training Samples

2-layer DBM

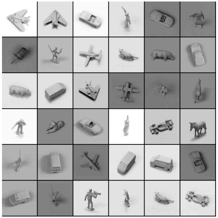
3-layer DBM

- handwritten digits (28x28 Pixels)
- 60.000 Trainings images
- 10.000 Test images
- two DBMs were trained
  - 2-layer (500 and 1000 hidden units)
  - 3-layer (500, 500 and 1000 hidden units)
- error rates for discriminative models:
  - 2-layer DBN (baseline): 1,2%
  - 2-layer DBM: 0,95%
  - 3-layer DBM: 1,01%

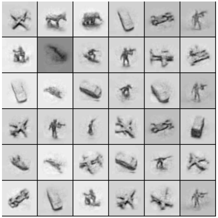
Experimental Results

NORB

TECHNISCHE UNIVERSITÄT DARMSTADT



Training Samples



Generated Samples

- 24.300 stereo images (96x96 pixels)
  - 50 different 3D toy objects
  - 5 generic classes
  - 4.300 were used as test set
  - larger pixels around edges for dimensionality reduction
- trained DBMs:
  - preprocessing layer with 4000 units
  - two further layers with 4000 units
  - completely unsupervised
  - second DBM trained with additional 1.16M training instances
- error rates:
  - SVM (baseline): 11,6%
  - DBM: 10,8%
  - DBM (with additional training data): 7.1%

Conclusions

- DBMs find good features to model the data
- DBMs can be used for unsupervised learning
- Unsupervised Learning helps generalization
- labels do not carry much information

Sources

- R. Salakhutdinov, G. Hinton. 2009
  Deep Boltzmann Machines
- G. Hinton, S. Osindero, and Y. W. Teh. 2006
  A fast learning Algorithm for deep belief nets