Outline

- Key concepts
- Deep Belief Networks
- Convolutional Neural Networks
A couple of questions

- Convolution
- Perceptron
- Feedforward Neural Network
- Backpropagation
- Boltzmann Machine
- Belief Net
Convolutions

- **Discrete**: \( f \otimes g[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \)
- **Continuous**: \( f \otimes g(n) = \int_{-\infty}^{\infty} f(m)g(n-m)dm \)
- **Image processing**: Image processing involves applying a weighted sum of pixel values.

![Image processing example](image.png)
Perceptron

\[ y = \varphi \left( \sum_{i=1}^{n} w_i x_i + b \right) \]
Feedforward Neural Network
Backpropagation

- Training algorithm for feedforward neural networks
- Gives a formula for the derivative of the cost function w. r. t. the weights in the network
- Based on \textit{propagating} the error of the network \textit{backwards} through the feedforward network
  - Error at layer $k+1$ $\rightarrow$ error at layer $k$
  - Error at last layer obtained easily $\rightarrow$ error of entire network obtainable
Restricted Boltzmann Machine

- An artificial neural network capable of learning a probability distribution characterising the (training) data
- Two layers – one hidden, one visible; fully connected
- Weight matrix, bias units
Restricted Boltzmann Machine

- Probability of states of hidden neurons given the states of visible neurons:
  \[ p(h_j = 1 | v) = \sigma \left( b_j + \sum_i v_i w_{ij} \right) \]

- Probability of states of visible neurons given the states of hidden neurons:
  \[ p(v_i = 1 | h) = \sigma \left( a_i + \sum_j h_j w_{ij} \right) \]

- Where \( \sigma(x) \) is the logistic function: \( \frac{1}{1 + e^{-x}} \)
Belief Network
Deep Belief Networks

Hinton et al., 2006
The Problem

- Representing complex characteristics of the data requires a complex belief network
- A complex (large) belief network is often impossible to train properly
The Solution

- The Deep Belief Network by Hinton et al. (2006)
  - layered architecture (layers of RBMs)
  - a greedy training algorithm (layer-by-layer)
Restricted Boltzmann Machine

- An artificial neural network capable of learning a probability distribution characterising the (training) data
- Two layers – one hidden, one visible; fully connected
- Weight matrix, bias units
Restricted Boltzmann Machine

- Probability of states of hidden neurons given the states of visible neurons:
  \[ p(h_j = 1 | v) = \sigma \left( b_j + \sum_i v_i w_{ij} \right) \]

- Probability of states of visible neurons given the states of hidden neurons:
  \[ p(v_i = 1 | h) = \sigma \left( a_i + \sum_j h_j w_{ij} \right) \]

- Where \( \sigma(x) \) is the logistic function: \( 1/(1 + e^{-x}) \)
The (training) Procedure

1. Train the first layer as an RBM on the input data
2. Use the ‘output’ of the layer as a representation of the data, and as the input of the next layer
3. Train the next layer
4. Repeat 2. and 3.
5. Fine tune the network (supervised gradient descent)
The Results

- MNIST digit recognition
  - handwritten digits
  - 28 x 28 grey values + labels
  - 60,000 digits in training set, 10,000 in test set
- Hinton et al., 2006: 1.25 % test error
  - “Inside the mind” of the DBN:

```
0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9
```
The Usage

- Standalone
- With a classifier (e.g. logistic regression) on top
- As pre-training for supervised models
Convolutional Neural Networks
Convolution

- Discrete: \[ f \otimes g[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \]

- Continuous: \[ f \otimes g(n) = \int_{-\infty}^{\infty} f(m)g(n-m)dm \]

- Image processing:
The Ideas

- Sparse connectivity
The Ideas

- Shared weights
The Training

- Backpropagation
  - adapted to learn convolutional layers
  - also deals with max-pooling layers (send error to “argmax” only)
The results

- **MNIST**
  - less than 1% (multiple entries, close to 0.2%)

- **ImageNet ILSVRC**
  - object recognition challenge
  - 1 200 000 training images
  - 150 000 testing images
  - various resolutions
  - labeled (using AMT) as ~1000 different categories
  - top-5 and top-1 evaluation
The results

- Krizhevsky et al., 2012:
  - downsampling of images to 256x256
  - 60 million parameters
  - uses a couple of “tricks”
    - Rectified linear units
    - overlapping max-pooling
    - output: 1000-way softmax
    - data augmentation – translation & reflection, simulated illumination variation
    - dropout
Rectified Linear Units

- **Sigmoid activation**: \( \frac{1}{1+e^{-x}} \)
- **Tanh activation**: \( \tanh(x) \)
- **ReLU**: \( \max(0, x) \)
Max-pooling

- Take the maximum of the inputs
- A form of downsampling
Softmax activation

- A way to interpret the output the last layer of a network as a probability distribution
- Essential idea: Normalize the outputs so they
  - Lie between 0 and 1
  - Sum to 1
- Formula: \( p_i = \frac{e^{q_i}}{\sum_{j=1}^{m} e^{q_j}} \),
  where \( q_i \) is the input of the i-th unit.
Dropout

- Training technique
- When training, randomly set some of the neuron outputs to zero
- Reduces co-adaptation of neurons (reliance on each other)
Data Augmentation

- „Create more data“
- e. g. take not only the existing data, but also transformations of the data which „make sense“ (keeps the label, is part of same set,...)
- Reduces overfitting
The finished product

- $253,440 - 186,624 - 64,896 - 64,896 - 43,264 - 4096 - 4096 - 1000$
Results

- top-5 error: 16.4%
- second place: ~26%
What else?

- $253,440 - 186,624 - 64,896 - 64,896 - 43,264 - 4096 - 4096 - 1000$
Similarity (Krizhevsky, 2012)
Similarity (MUFIN, 2005)
Resources

- neuralnetworksanddeeplearning.com
  - free online book draft
- deeplearning.net
  - Python tutorial to DBNs, CNNs
- image-net.org
  - the ILSVRC challenge
  - MNIST dataset & results
Papers

- Bengio, 2009: Learning Deep Architectures for AI
  - survey with details
- Hinton et al., 2006: A Fast Learning Algorithm for Deep Belief Nets
  - Deep belief networks
- Krizhevsky et al., 2012: ImageNet Classification with Deep Convolutional Neural Networks
  - Convolutional network
- Schmidhuber, 2014: Deep Learning in Neural Networks: An Overview
  - huge survey
Presentations

- [GoogLeNet](http://image-net.org/challenges/LSVRC/2014/slides/GoogLeNet.pptx), Google’s winning implementation with improvements, 2014
  - J. Matas (ČVUT) talks about the history of object detection and why it is upside down since Krizhevsky 2012, Prague Computer Science Seminar, 2014
- [A Shallow Introduction into the Deep Machine Learning](https://cw.felk.cvut.cz/wiki/_media/courses/ae4m33mpv/deep_learning_mpv.pdf)
  - J. Čech: A Shallow Introduction into the Deep Machine Learning, 2014, a nice readable introduction to Krizhevsky and others with lots of examples