Object Detection
Lecture 10.3 - Introduction to deep learning (CNN)

Idar Dyrdal
Deep Learning

- Computational models composed of multiple processing layers (non-linear transformations)
- Used to learn representations of data with multiple levels of abstraction:
  - Learning a hierarchy of feature extractors
  - Each level in the hierarchy extracts features from the output of the previous layer (pixels → classes)
- Deep learning has dramatically improved state-of-the-art in:
  - Speech and character recognition
  - Visual object detection and recognition
- Convolutional neural nets for processing of images, video, speech and signals (time series) in general
- Recurrent neural nets for processing of sequential data (speech, text).
Deep Learning for Object Recognition

Millions of images

Millions of parameters

Thousands of classes
Traditional supervised learning

1. Training images
2. Feature extraction
3. Classifier training
4. Class labels

Handcrafted features
Deep learning

- Learning of weights in the processing layers
- Supervised, unsupervised (or semi-supervised) learning

1. Training images
2. Feature extraction
3. Classifier training
4. Classifier

Learned features
Semi-supervised learning

Labeled samples and (trained) linear decision boundary

Labeled and unlabeled samples and non-linear decision boundary
Artificial Neural Network (ANN)

Used in Machine Learning and Pattern Recognition:
- Regression
- Classification
- Clustering
- ...

Applications:
- Speech recognition
- Recognition of handwritten text
- Image classification
- ...

Network types:
- Feed-forward neural networks
- Recurrent neural networks (RNN)
- ...

Feed-forward ANN (non-linear classifier)
Mark 1 Perceptron (Rosenblatt, 1957-59)

\[
O = f \left( \sum_{k=1}^{d} i_k w_k \right)
\]
Activation functions

- Sigmoid (logistic function):
  \[ f(x) = \frac{1}{1 + e^{-x}} \]

- Hyperbolic tangent:
  \[ f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \]

- Rectified linear unit (ReLU):
  \[ f(x) = \max(x, 0) \]
Feed-forward neural network

Output layer

Hidden layer $H_2$

Hidden layer $H_1$

Input layer

\[ y_l = f(z_l) \]
\[ z_l = \sum_{k \in H_2} w_{kl} x_k \]

\[ y_k = f(z_k) \]
\[ z_k = \sum_{j \in H_1} w_{jk} x_j \]

\[ y_j = f(z_j) \]
\[ z_j = \sum_{i \in \text{Input}} w_{ij} x_i \]
Back-propagation

\[
\frac{\partial E}{\partial y_l} = y_l - t_l \\
\frac{\partial E}{\partial z_l} = \frac{\partial E}{\partial y_l} \frac{\partial y_l}{\partial z_l}
\]

\[
E(w) = \sum_{k=1}^{n} (t_i - y_i)^2
\]

\[
\frac{\partial E}{\partial y_k} = \sum_{i \in \text{Output}} w_{kl} \frac{\partial E}{\partial z_l}
\]

\[
\frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k}
\]

\[
\frac{\partial E}{\partial y_j} = \sum_{k \in H_2} w_{jk} \frac{\partial E}{\partial z_k}
\]

\[
\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_j}
\]
Convolutional Neural Network (CNN)

Used in Machine Vision and Image Analysis:
• Speech Recognition
• Image Recognition
• Video Recognition
• Image Segmentation
• …

Convolutional neural network:
• Multi-layer feed-forward ANN
• Combinations of convolutional and fully connected layers
• Convolutional layers with local connectivity
• Shared weights across spatial positions
• Local or global pooling layers

(A. Karpathy)
Typical CNN

(Aphex34)
Convolutional neural net

Input image

(credit: S. Lazebnik)
Convolutional neural net

Input

Feature Map

(credit: S. Lazebnik)
Convolutional neural net

Rectified Linear Unit (ReLU)

(credit: S. Lazebnik)
Convolutional neural net

Max pooling

Max-pooling: a non-linear down-sampling

Provide *translation invariance*

(credit: S. Lazebnik)
Convolutional neural net

Feature Maps

Feature Maps After Contrast Normalization

(credit: S. Lazebnik)
Convolutional neural net

Feature maps after contrast normalization

(credit: S. Lazebnik)
Example - Caffe Demos

The Caffe neural network library makes implementing state-of-the-art computer vision systems easy.

Classification

Click for a Quick Example

<table>
<thead>
<tr>
<th>Maximally accurate</th>
<th>Maximaly specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Egyptian cat</td>
<td>0.32645</td>
</tr>
<tr>
<td>tabby</td>
<td>0.16689</td>
</tr>
<tr>
<td>tiger cat</td>
<td>0.10922</td>
</tr>
<tr>
<td>Persian cat</td>
<td>0.06203</td>
</tr>
<tr>
<td>Siamese cat</td>
<td>0.05992</td>
</tr>
</tbody>
</table>

CNN took 0.112 seconds.

http://demo.caffe.berkeleyvision.org
Caffe Demos

The Caffe neural network library makes implementing state-of-the-art computer vision systems easy.

Classification

Click for a Quick Example

<table>
<thead>
<tr>
<th>Maximally accurate</th>
<th>Maximally specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>macaw</td>
<td>0.99995</td>
</tr>
<tr>
<td>lorikeet</td>
<td>0.00008</td>
</tr>
<tr>
<td>crane</td>
<td>0.00002</td>
</tr>
<tr>
<td>vulture</td>
<td>0.00002</td>
</tr>
<tr>
<td>flamingo</td>
<td>0.00002</td>
</tr>
</tbody>
</table>

CNN took 0.067 seconds.
The Caffe neural network library makes implementing state-of-the-art computer vision systems easy.

## Classification

**Click for a Quick Example**

<table>
<thead>
<tr>
<th>Maximally accurate</th>
<th>Maximally specific</th>
<th>0.20551</th>
</tr>
</thead>
<tbody>
<tr>
<td>suspension bridge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lakeside</td>
<td></td>
<td>0.16864</td>
</tr>
<tr>
<td>pier</td>
<td></td>
<td>0.12692</td>
</tr>
<tr>
<td>alp</td>
<td></td>
<td>0.05951</td>
</tr>
<tr>
<td>radio telescope</td>
<td></td>
<td>0.04751</td>
</tr>
</tbody>
</table>

CNN took 0.254 seconds.
Example - Semantic Segmentation (SegNet)

http://mi.eng.cam.ac.uk/projects/segnet/
Summary

Topics covered:
• Deep learning
• Artificial neural networks
• Convolutional neural networks

More information:
• Szeliski, chapter 14

Software:
• Caffe (http://caffe.berkeleyvision.org)
• TensorFlow (https://www.tensorflow.org/)
• MatConvNet (http://www.vlfeat.org/matconvnet)