Machine Learning

Lecture 7 - Image data and convolutional neural networks

Mattias Villani

Department of Statistics
Stockholm University

Department of Computer and Information Science
Linköping University

mattiasvillani.com @matvil mattiasvillani

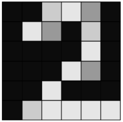
Lecture overview

- Image data
- Filters and convolutions
- Convolutional neural networks

Image are 2D-grid structured data

- **Image data**:
  - are structured. Pixels on a 2D grid.
  - nearby pixels tend to be correlated.
  - lines, corners, shapes.

Image



Data representation

0.0	0.0	0.8	0.9	0.6	0.0
0.0	0.9	0.6	0.0	0.8	0.0
0.0	0.0	0.0	0.0	0.9	0.0
0.0	0.0	0.0	0.9	0.6	0.0
0.0	0.0	0.9	0.0	0.0	0.0
0.0	0.8	0.9	0.9	0.9	0.9

Input variables

$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$	$x_{1,6}$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	$x_{2,5}$	$x_{2,6}$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$	$x_{3,5}$	$x_{3,6}$
$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$	$x_{4,5}$	$x_{4,6}$
$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$	$x_{5,5}$	$x_{5,6}$
$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$	$x_{6,5}$	$x_{6,6}$

- **3D data**: activity in brain **voxels** in neuroscience. 2D video.
- **4D data**: activity in brain voxels over time.

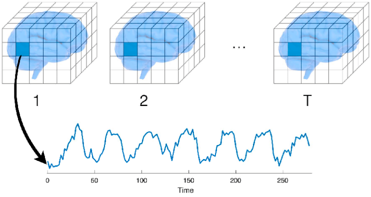
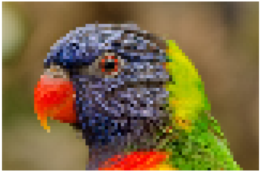


Image data - pixels

Original

Twice less pixels

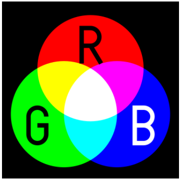
4 times less



16 times less

RGB data

- Red-Green-Blue triples.
- (r, g, b) where $r, g, b \in \{0, 1, \ldots, 255\}$.



Red pixel (217,7,0)

Yellow pixel (219,222,4)

Gray pixel (135,137,144)

- Grayscale: (r, g, b) where $r = g = b$.
- Other (better) color spaces exist.

Convolution in 1D

- **Convolutional networks**: uses convolution instead of general matrix multiplication in at least one of the layers.
- **Discrete convolution** for 1-dimensional data (e.g. time)

$$z_t = \sum_{\tau=-\infty}^{\infty} x_\tau w_{t-\tau}$$

- Example: 3-point (weighted) **moving average**

$$z_t = \frac{1}{4}x_{t-1} + \frac{1}{2}x_t + \frac{1}{4}x_{t+1}.$$

- Convolutions are often **sparse linear transformations**

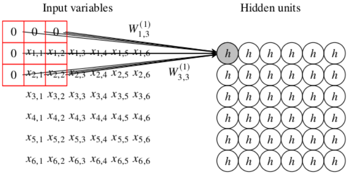
$$\begin{pmatrix} z_2 \\ z_3 \\ \\ z_T \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & \cdots & 0 \\ \vdots & & & & & & 0 \\ 0 & 0 & 0 & \cdots & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \\ x_{T-1} \end{pmatrix}$$

Convolution in 2D

- **2D-convolution** (input x has been **zero-padded**):

$$z_{ij} = \sum_{k=1}^{F} \sum_{l=1}^{F} x_{i+k-1,j+l-1} W_{kl}$$

- **Zero-padding** to handle edges



- Simple **blur filter**

$$W = \begin{pmatrix} 0.050 & 0.125 & 0.050 \\ 0.125 & 0.300 & 0.125 \\ 0.050 & 0.125 & 0.050 \end{pmatrix}$$

- **Gaussian blur**

Edge detecting filters

- 2D-convolution

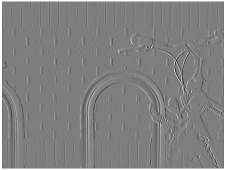
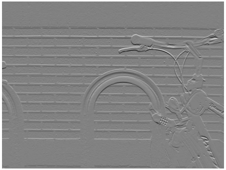
$$z_{ij} = \sum_{k=1}^{F} \sum_{l=1}^{F} x_{i+k-1,j+l-1} W_{kl}$$

- Sobel filter to detect vertical edge

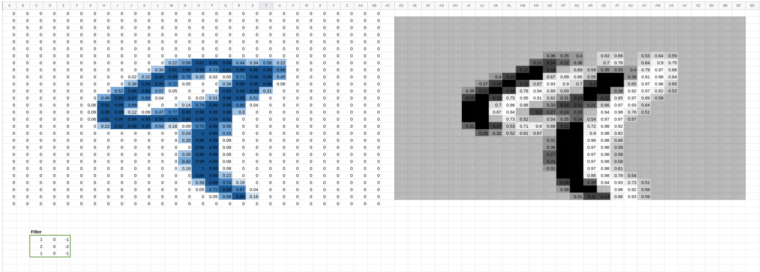
$$W = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$$

- Sobel filter to detect horizontal edge

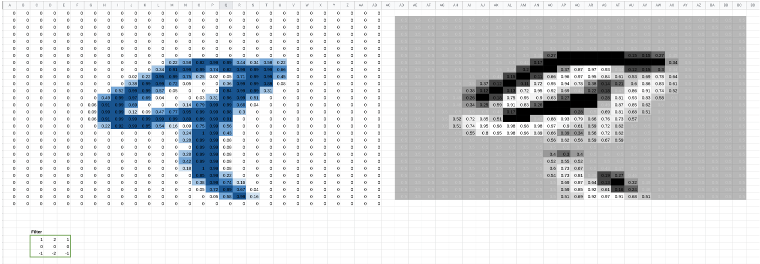
$$W = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

Edge detecting filters - vertical



Edge detecting filters - horizontal

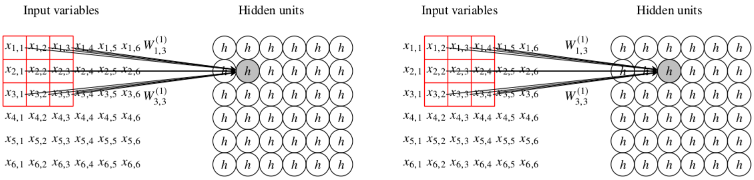


Filter

1	2	1
0	0	0
-1	-2	-1

Convolutional neural networks

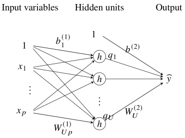
- **Convolutional neural networks (CNN)**:
  - **Keeps the 2D** structure of images.
  - Uses **convolutions** (filters) in middle layers.
  - **Sparse weights**: hidden units depend only on some inputs.
  - **Parameter sharing**: same filter coefficients over whole grid. **Equivariant to translations**.
  - **Filter coefficients are learned** by SGD like any weights.



Multiple channels and tensor weights

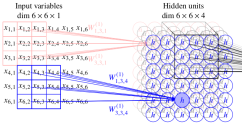
- CNNs use **multiple filters** to capture different shapes. E.g. horizontal and vertical edge detectors may be learned.
- Each filter gives a **channel**. Weights as 3D-**tensors**.
- Also inputs can have multiple channels, e.g. RGB images.

Fully connected



$$W_{i,j}^{(\ell)}$$

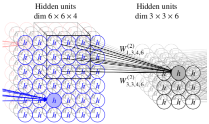
Multiple channel CNN



$$W_{i,j,k}^{(\ell)}$$

$$k = 1, \dots, 4$$

Dual multiple channel CNN

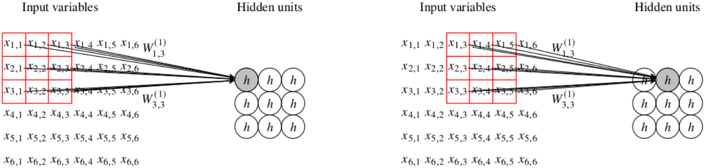


$$W_{i,j,k,l}^{(\ell)}$$

$$k = 1, \dots, 4$$
$$l = 1, \dots, 6$$

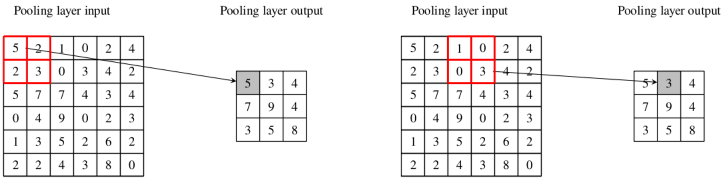
Stride

- **Downsampling** of images to reduce image size.
- **Stride** of size k: Keep only every kth filter output.

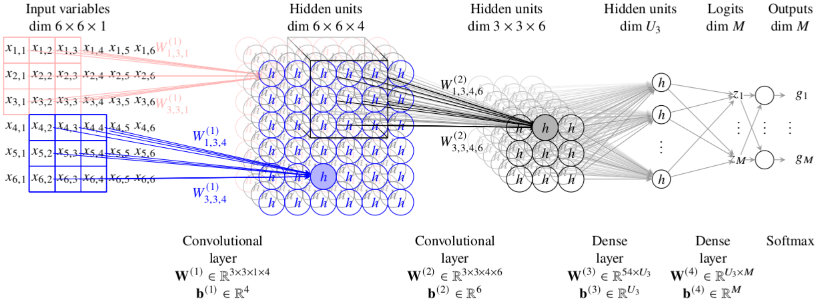


Pooling

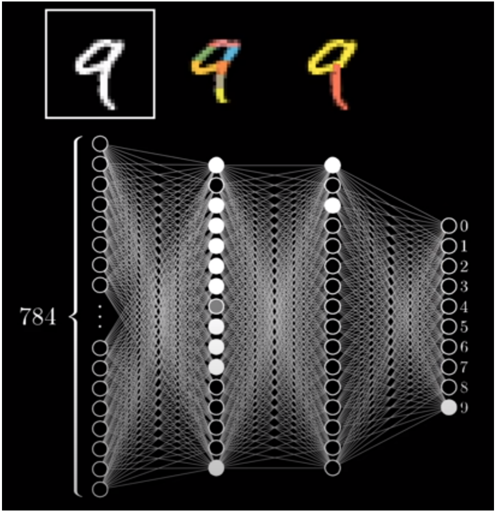
- **Average pooling**: keep only average of a region.
- **Max pooling**: keep only max of a region.
- Alternative to downsampling by stride.
- Pooling gives **invariance to small translations**.
- Pooling is applied as a new layer without parameters.



Deep convolutional neural networks



Recall: dense network learns features



From video by 3Blue1Brown: `https://youtu.be/aircAruvnKk`

Convolutional neural networks learns features

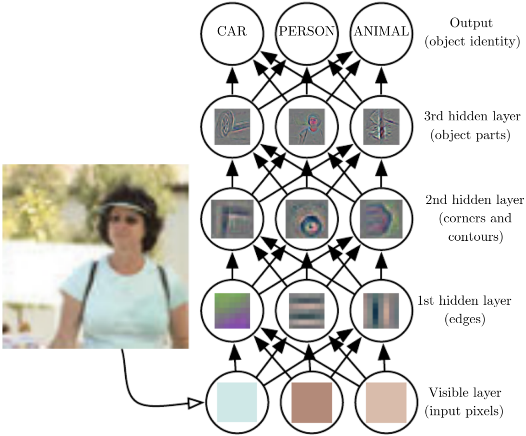


Figure from Goodfellow et al (2016).

Feature learning

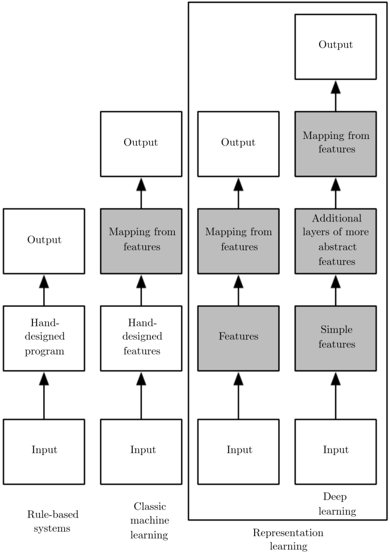


Figure from Goodfellow et al (2016).

Deeper is better

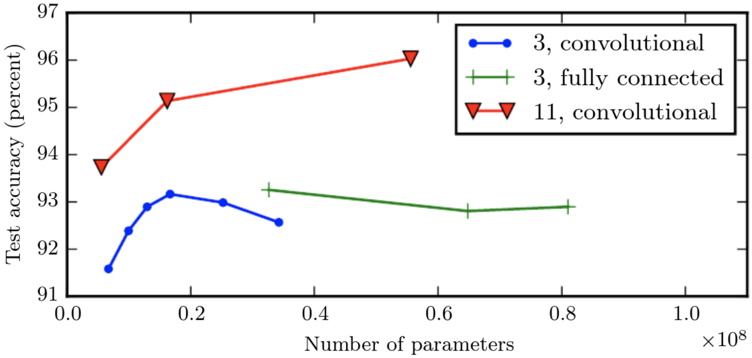


Figure from Goodfellow et al (2016).

Regularization in deep learning

- **Early stopping** - monitor test performance and stop optimization when test performance deteriorates.
- **L1/L2 regularization** of network weights.
- **Dropout**
  - remove hidden units with probability r from a layer at each mini-batch step during optimization.
  - Approximates bagging of networks or Bayesian variable selection.