Introduction to Neural Networks

Part I : Neural information processing

Web site of this course: http://pattern-recognition.weebly.com.

Two topics

Neural information processing

- Origins
- Perceptron
- Multilayer perceptron (MLP)
- Convolutional neural network (CNN)

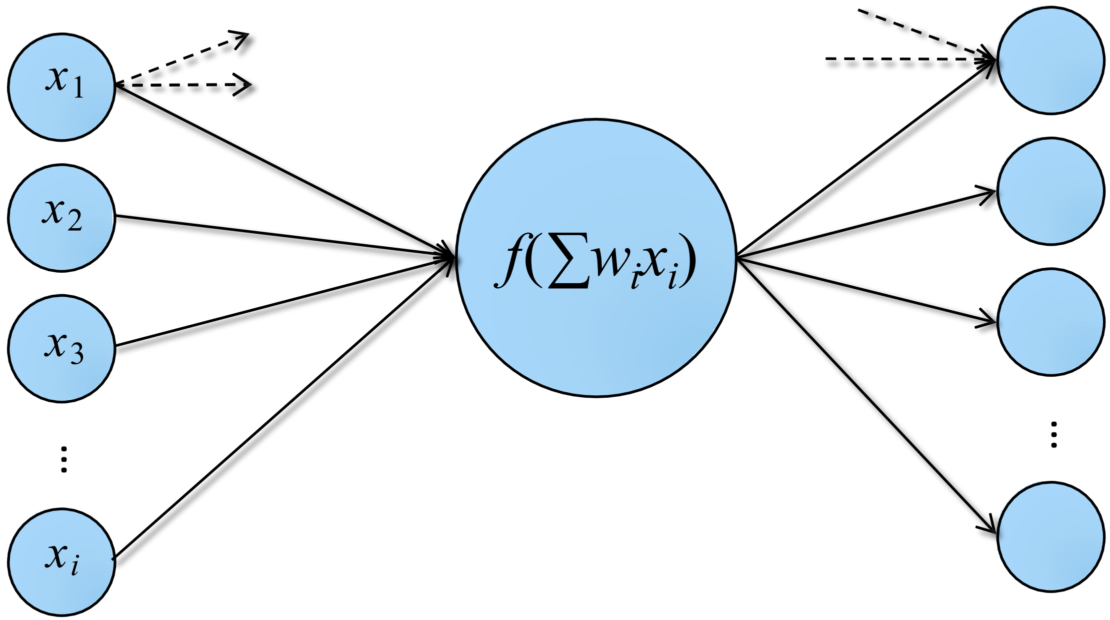
Training multilayer networks

- Perceptron learning
- Optimization basics
- Stochastic gradient descent
- Backpropagation learning algorithms

Neural Information Processing

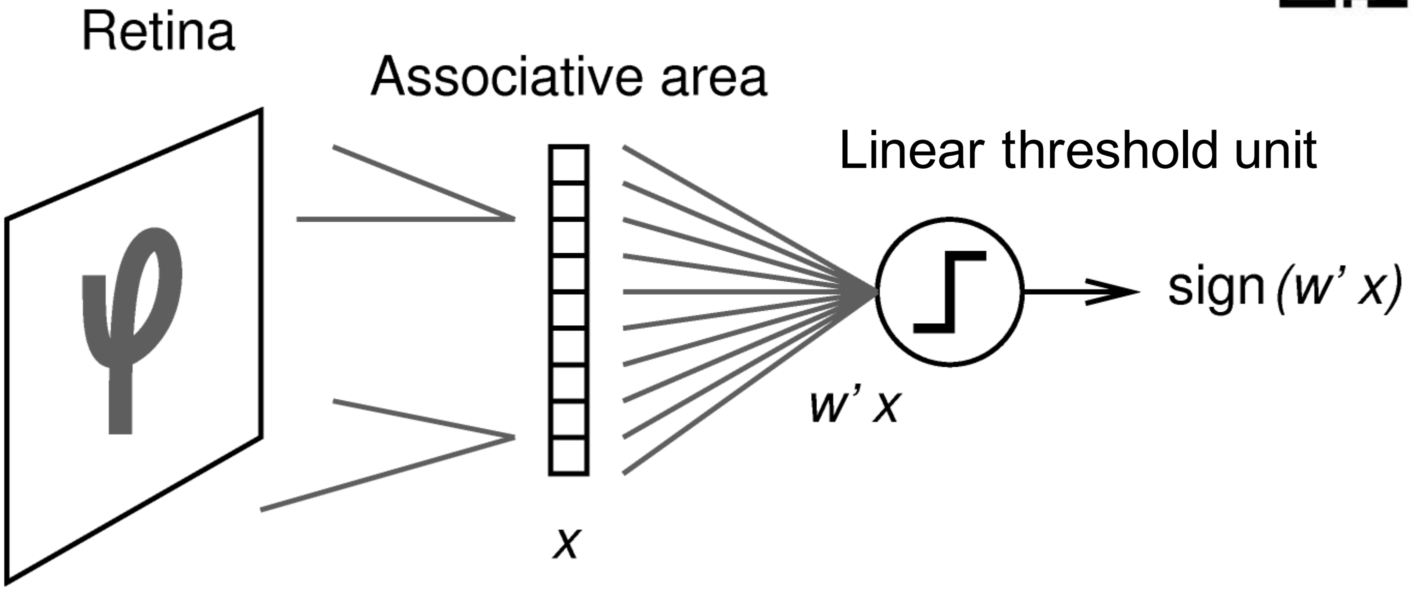
- **Origins**
- Perceptron
- Multilayer perceptron
- Convolutional neural network

McCulloch & Pitts (1943)



A simplified neuron model: the Linear Threshold Unit.

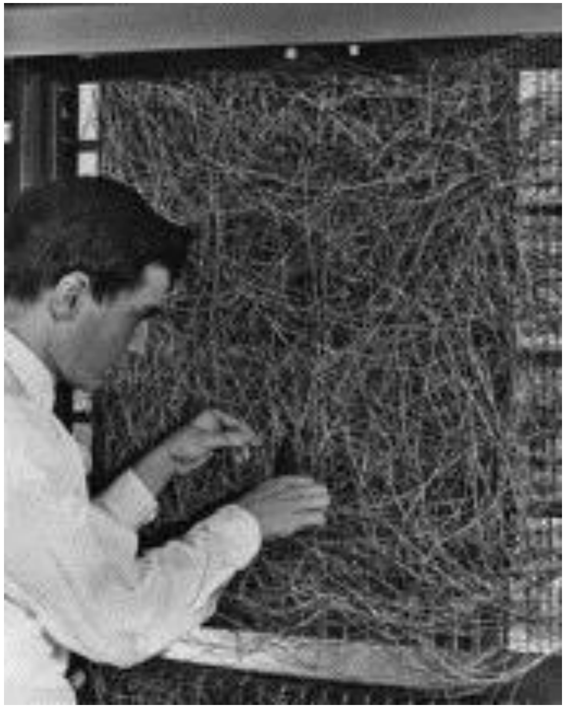
The perceptron (1957)



Supervised learning of the weights using the Perceptron algorithm.

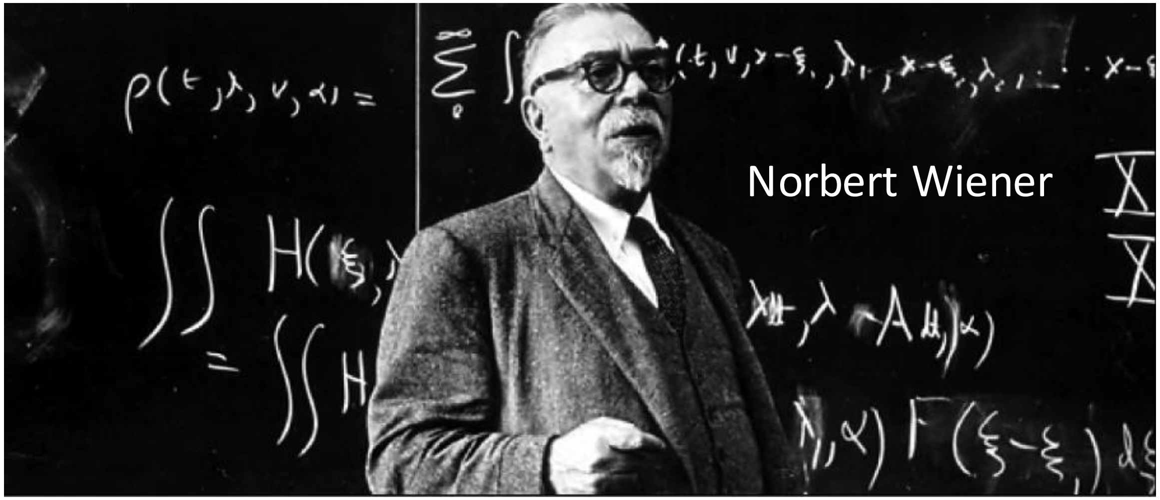
Rosenblatt 1957

The perceptron is a machine



Frank Rosenblatt

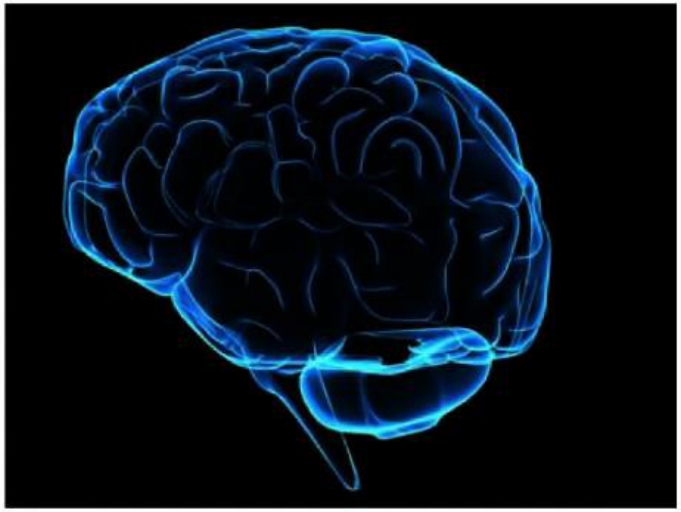
Cybernetics (1948)



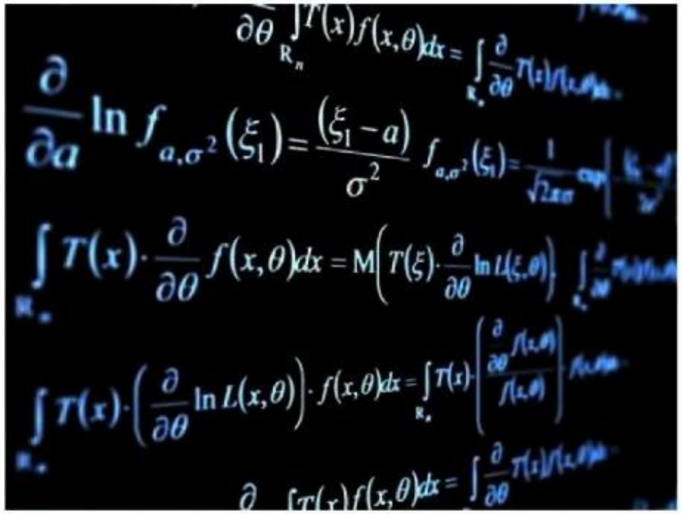
- Mature communication technologies, nascent computing technologies
- Redefining the man-machine boundary

Two camps to design computers

Biological computer



Mathematical computer

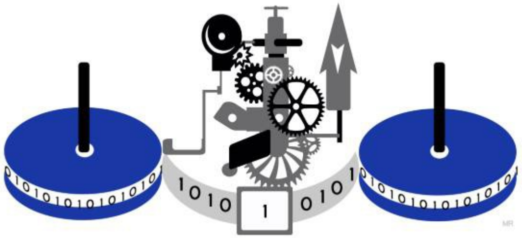


- Which model to emulate : brain or mathematical logic ?
- Mathematical logic won.

Computing with symbols

General computing machines

- Turing machine
- von Neumann machine



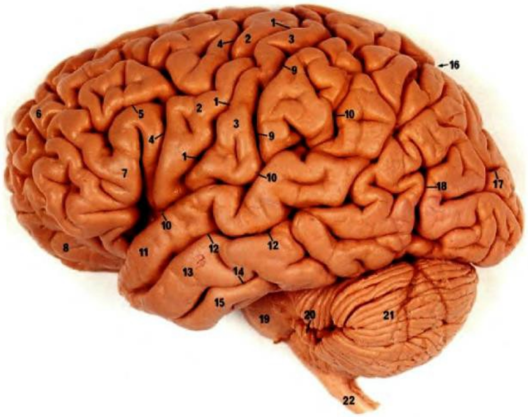
Engineering

- Programming
  (reducing a complex task into
  a collection of simple tasks.)
- Computer language
- Debugging
- Operating systems
- Libraries

Computing with the brain

An engineering perspective of brain

- Compact
- Energy efficient (20 watts)
- 10^{12} Glial cells (power, cooling, support)
- 10^{11} Neurons (soma + wires)
- 10^{14} Connections (synapses)
- Volume = 50% glial cells + 50% wires.



Could brain be a *general* computing machine?

- Basically, brain is
  - Slow for mathematical logic, arithmetic, etc.
  - Very fast for vision, speech, language, social interactions, etc.
- Because of brain evolution : vision -> language –> logic.

Success stories

Record performance

- MNIST (1988, 2003, 2012)
- ImageNet (2012)
- …

MNIST

ImageNet

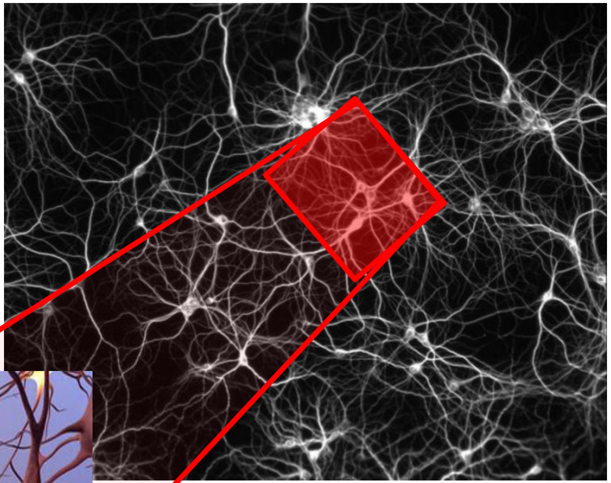
Real applications

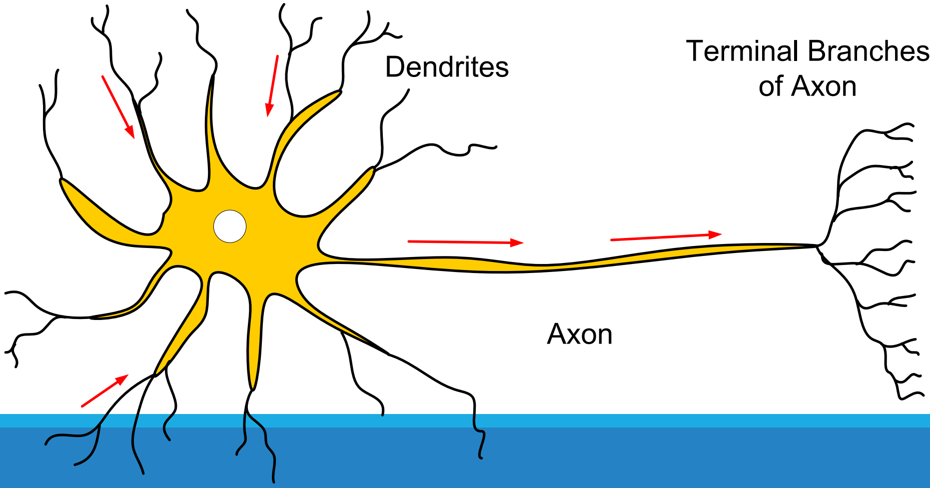
- Check reading (AT&T Bell Labs, 1995 – 2005)
- Optical character recognition (Microsoft OCR, 2000)
- Cancer detection from medical images (NEC, 2010)
- Object recognition (Google and Baidu's photo taggers, 2013)
- Speech recognition (Microsoft, Google, IBM switched in 2012)
- Natural Language Processing (NEC 2010)
- …

Neural Information Processing

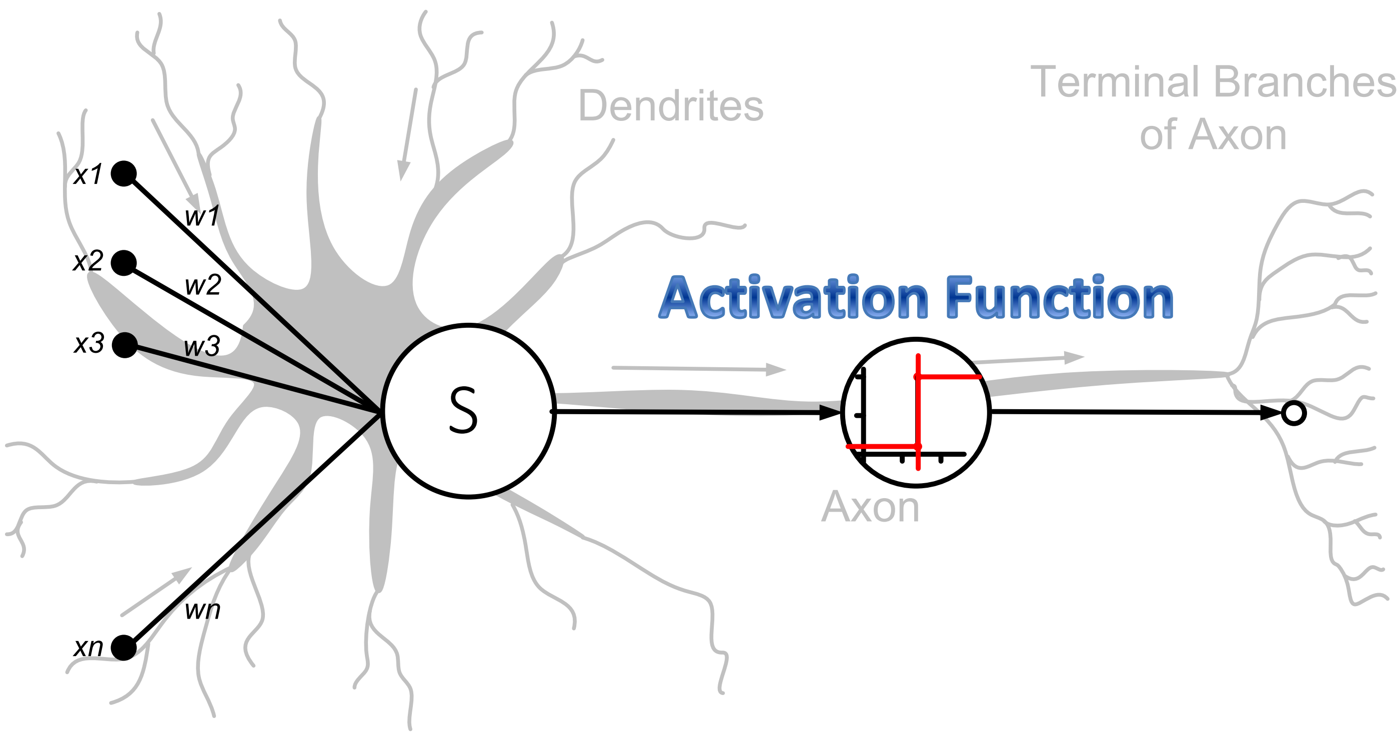
- Origins
- **Perceptron**
- Multilayer perceptron
- Convolutional neural network

Biological Neuron



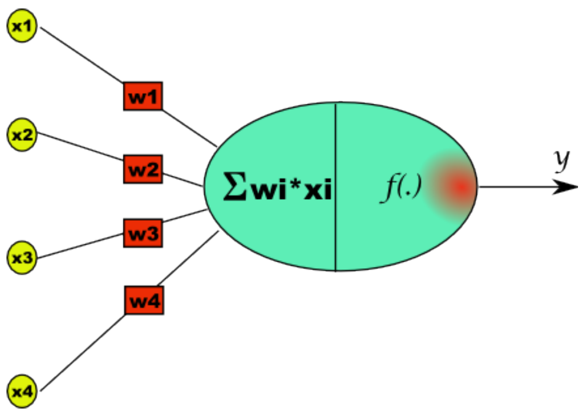


Artificial Neuron

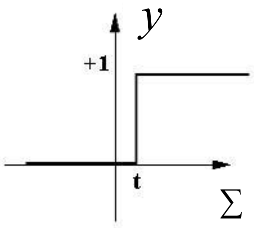


Neuron's Mathematical Model

A neuron



***f* : Transfer (Activation) Functions**



Step Function

$$f(w^T x) = \begin{cases} 1, if w^T x > t \\ 0, otherwise \end{cases}$$

Without f

$$y = \sum_{i=1}^{4} w_i x_i = w^T x$$

A neuron is a "linear equation"

With f

$$y = f(w^T x)$$

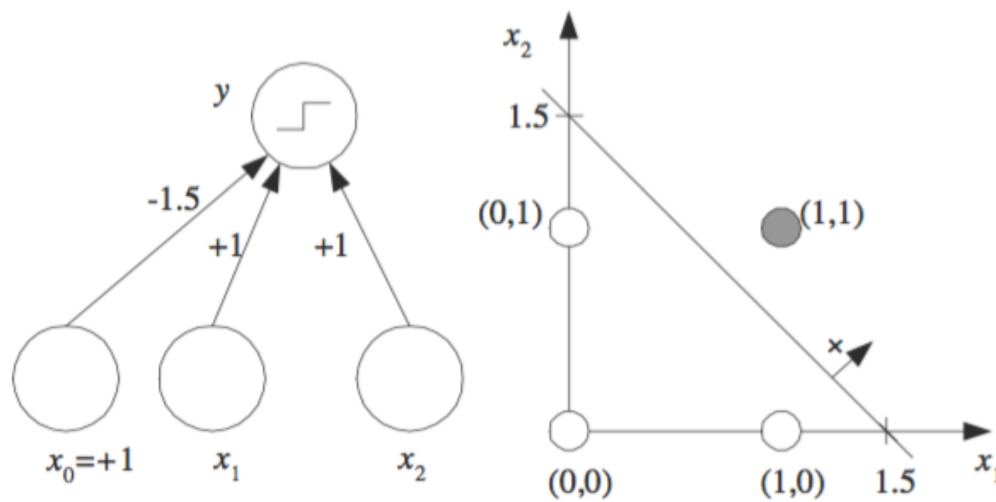
$$= \begin{cases} \text{C1 (label 1) if } w^T x > t \\ \text{C2 (label 0) if } w^T x \leq t \end{cases}$$

A neuron is a "linear classifier"

"And" function Classification

x_1	x_2	r
0	0	0
0	1	0
1	0	0
1	1	1

Input and output
for the AND function.



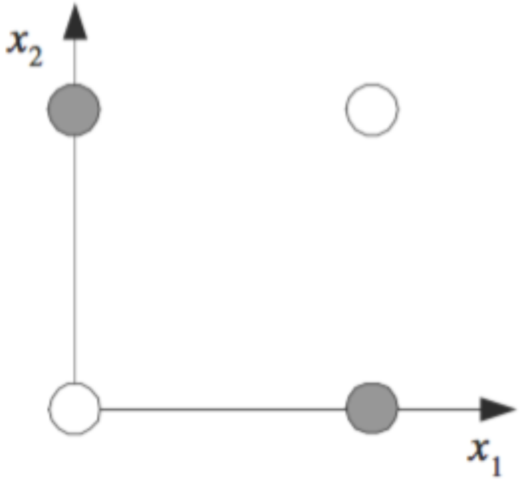
The perceptron that implements AND and
its geometric interpretation.

- The discriminant is $y = f(x1 + x2 - 1.5)$.

"XOR" Function Classification

x_1	x_2	r
0	0	0
0	1	1
1	0	1
1	1	0

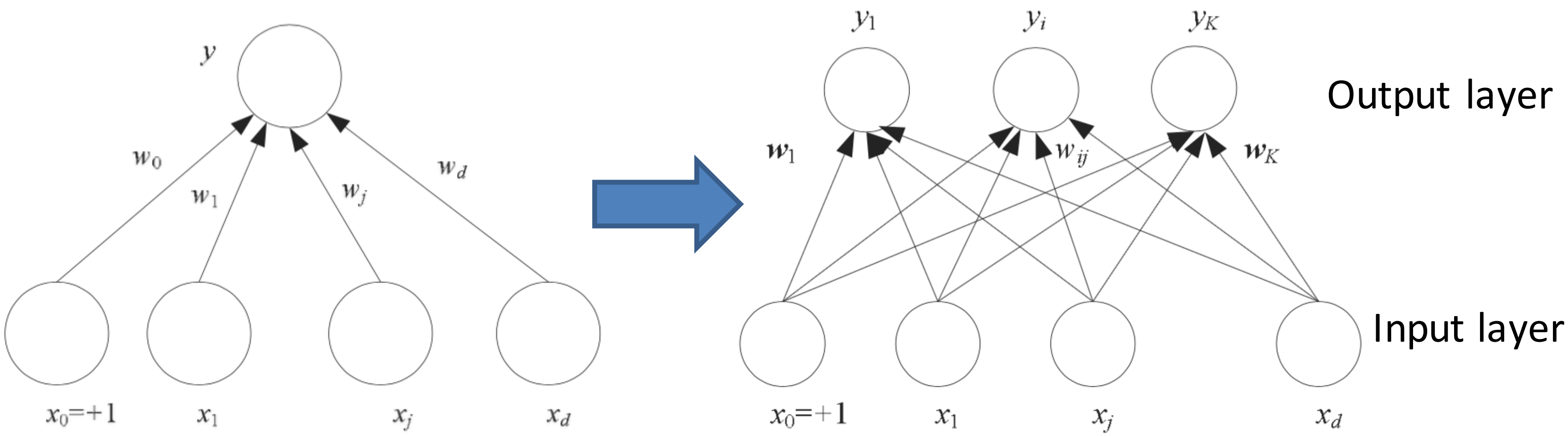
Input and output
for the XOR function



XOR problem is not linearly separable.
We cannot draw a line where
the empty circles are on one side and
the filled circles on the other side.

Perceptron can not solve the XOR classification problem

The General Perceptron Model



$$o = \sum_{j=1}^{d} w_j x_j + w_0 = w^T x$$

$$w = [w_0, w_1, \cdots, w_d]^T$$

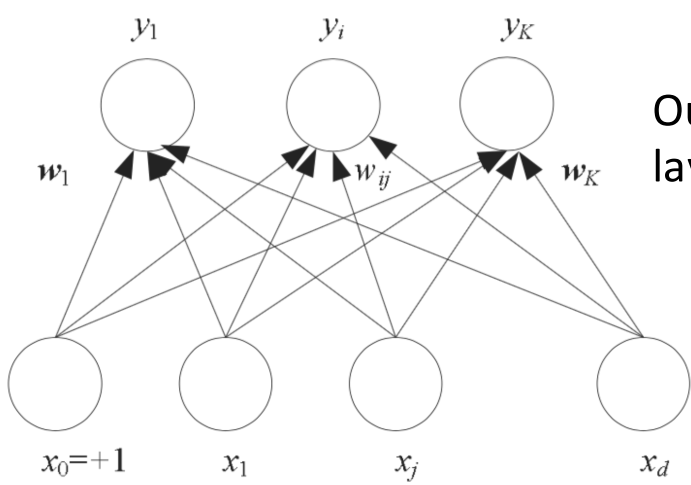
$$x = [1, x_1, \cdots, x_d]^T$$

$$y = \begin{cases} 1, & \text{if } f(o) > t \\ 0, & \text{if } f(o) \le t \end{cases}$$ (t is 0 usually)

$$o_i = w_i^T x = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

$$y_i = \begin{cases} 1, & \text{if } f(o_i) > 0 \\ 0, & \text{if } f(o_i) \le 0 \end{cases}$$

Perceptron Model for Classification



Output layer

Input layer

$$o_i = w_i^T x = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

$$y_i = \begin{cases} 1, \text{ if } f(o_i) > t_i \\ 0, \text{ if } f(o_i) \le t_i \end{cases}$$

For a K-class classification problem

- For a given x with unknown class
- $x \in$ class k, $if\ y_k = 1, y_j = 0\ for\ all\ j \neq k$
- Only one y_k can be 1

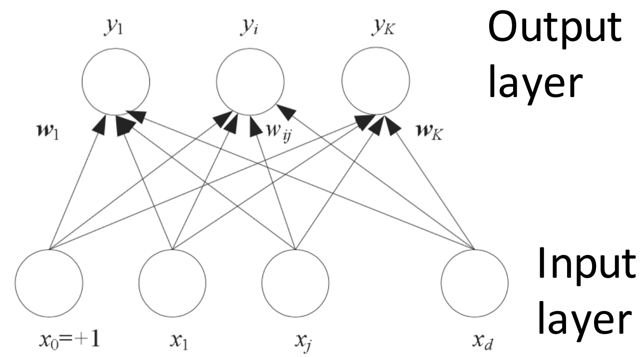
A simplified K-class classification problem

- $x \in$ class k, $if\ y_k = max_i y_i$
- $y_i = w_i^T x = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$

Training the perceptron

Testing for K-class classification problem

- For a given x with unknown class
- $x \in$ class k, $if\ y_k = max_i y_i$
- $y_i = w_i^T x = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$



That is

- A W represents a perceptron
- Given a W, then we can classify a pattern x

$$W = [w_1, w_2, \cdots, w_K] = \begin{bmatrix} w_{11} & \cdots & w_{K1} \\ \vdots & \ddots & \vdots \\ w_{1d} & \cdots & w_{Kd} \end{bmatrix}$$

A Machine Learning problem:

how to obtain the W of a perceptron

- We need a set of training patterns (X, Y)
- We need a learning algorithm to learn W by (X, Y)

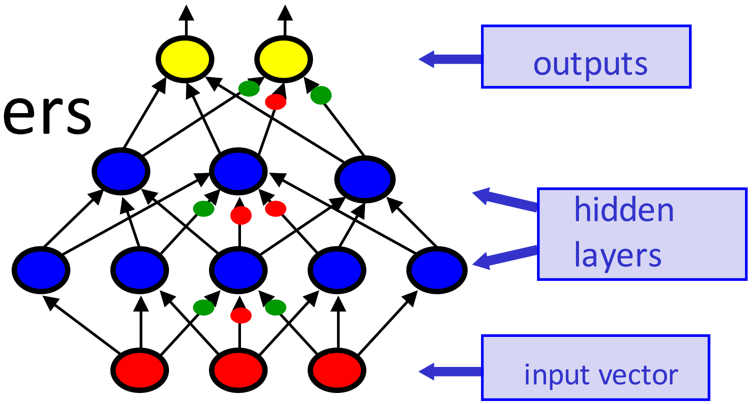
=> Perceptron learning algorithm A: $W = A(X, Y)$

Neural Information Processing

- Origins
- Perceptron
- **Multilayer perceptron (MLP)**
- Convolutional neural network

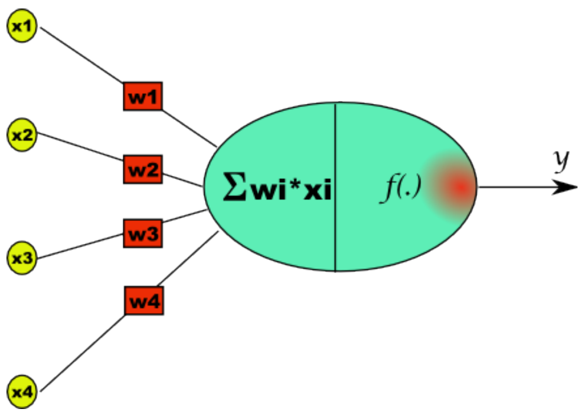
Multi-Layer Perceptron (MLP)

- Extension of perceptron from *one* layer into *multiple* layers
- Three differences compared with perceptron
  - Transfer(activation) function
    - From step function to sigmoid function
  - Layer
    - Add hidden layer
  - Training algorithm
    - Backpropagation learning algorithm to learn the weights



Neuron's Mathematical Model

Neuron:



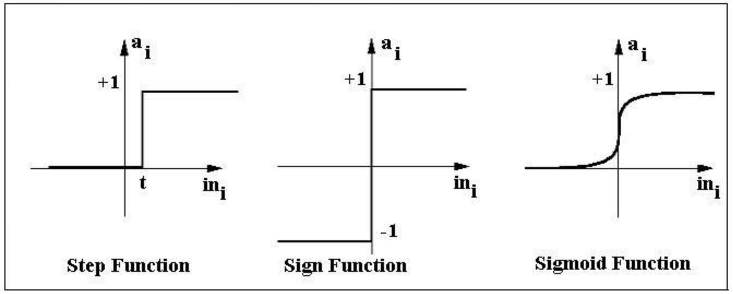
f is a *step* function

$$y = f(w^T x)$$

$$= \begin{cases} \text{C1 (label 1) if } w^T x > t \\ \text{C2 (label 0) if } w^T x \le t \end{cases}$$

A neuron is a "linear classifier"

***f*: Transfer (Activation) Functions**



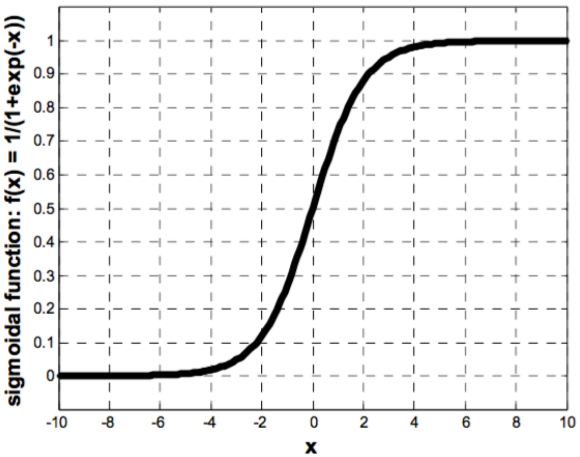
f is a *sigmoid* function

$$y = f(w^T x)$$

$$= \begin{cases} \text{C1 iff } (w^T x) > 0.5 \\ \text{C2 iff } (w^T x) \le 0.5 \end{cases}$$

A neuron is still a "linear classifier"

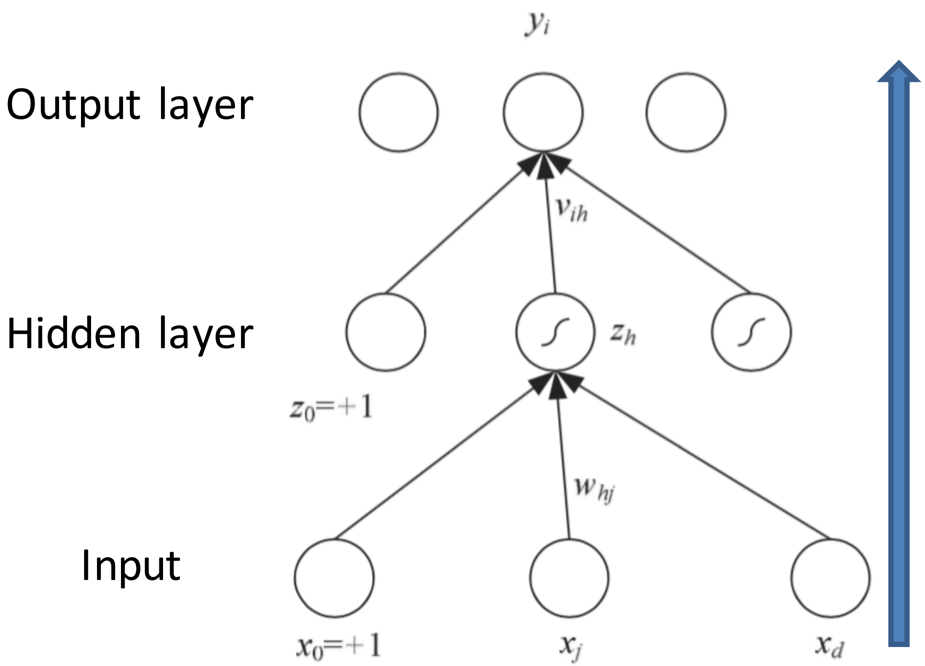
Transfer functions : Sigmoid



$$y = f(x) = \frac{1}{1+e^{-x}}$$

$$y = f(w^T x)$$
$$= \frac{1}{1+e^{-w^T x}}$$

Propagation from input to output



$$y_i = v_i^T z = \sum_{h=1}^{H} v_{ih} z_h + v_{i0}$$

$$z_h = sigmoid\left(w_h^T x\right)$$
$$= \frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{exp}\left[-\left(\sum_{j=1}^{d} \boldsymbol{w}_{hj}\boldsymbol{x}_j+\boldsymbol{w}_{h0}\right)\right]}$$

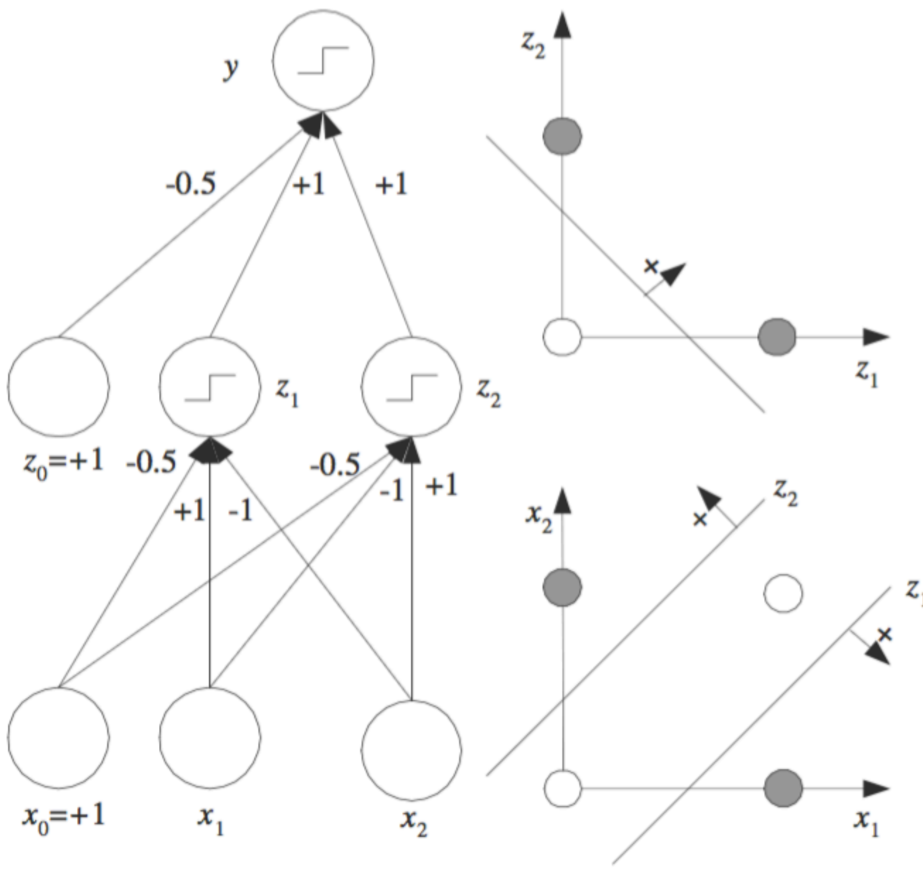
$$h = 1, \dots, H$$

MLP as a universal approximator

- The MLP that solves the XOR problem

$$x_1 \text{ XOR } x_2 = (x_1 \text{ AND } \sim x_2) \text{ OR } (\sim x_1 \text{ AND } x_2)$$

- The hidden units and the output have the threshold activation function with threshold at 0



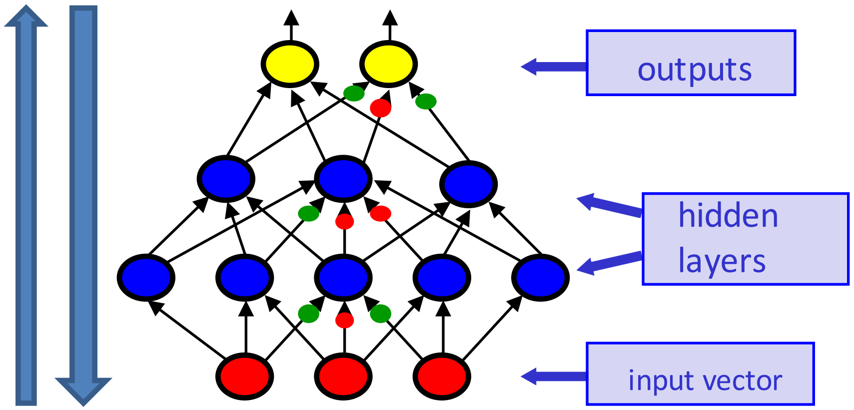
Training the MLP: Backpropagation

Testing for K-class classification problem

- For a given x with unknown class
- $x \in$ class k, $if\ y_k = max_i y_i$
- $y_i = v_i^T z = \sum_{h=1}^{H} v_{ih} z_h + v_{i0}$

That is

- A W represents a MLP
- Given a W, then we can classify a pattern x



$$W = [w_1, \cdots, w_K, v_1, \cdots, v_H]$$

A Machine Learning problem:

how to obtain the W of a MLP

- We need a set of training patterns (X, Y)
- We need a learning algorithm to learn W by (X, Y)

=> Backpropagation learning algorithm B: $W=B(X, Y)$

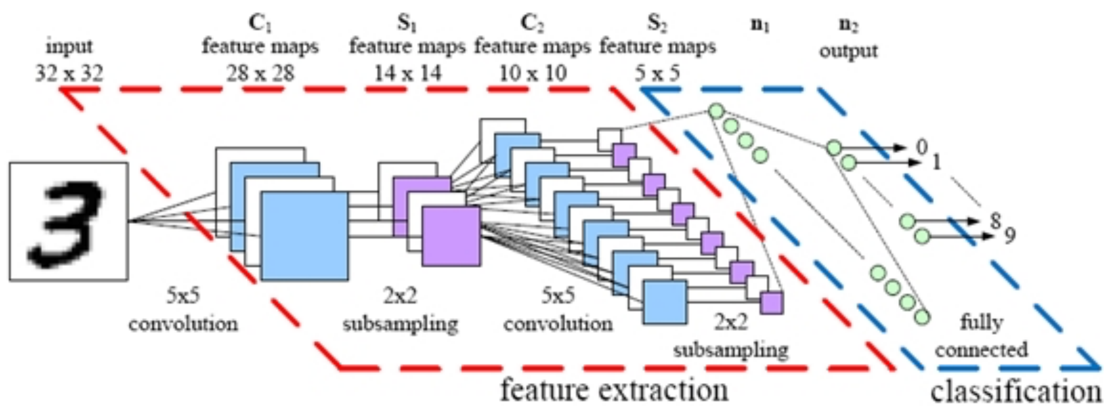
Neural Information Processing

- Origins
- Perceptron
- Multilayer perceptron network
- **Convolutional neural networks (CNN)**

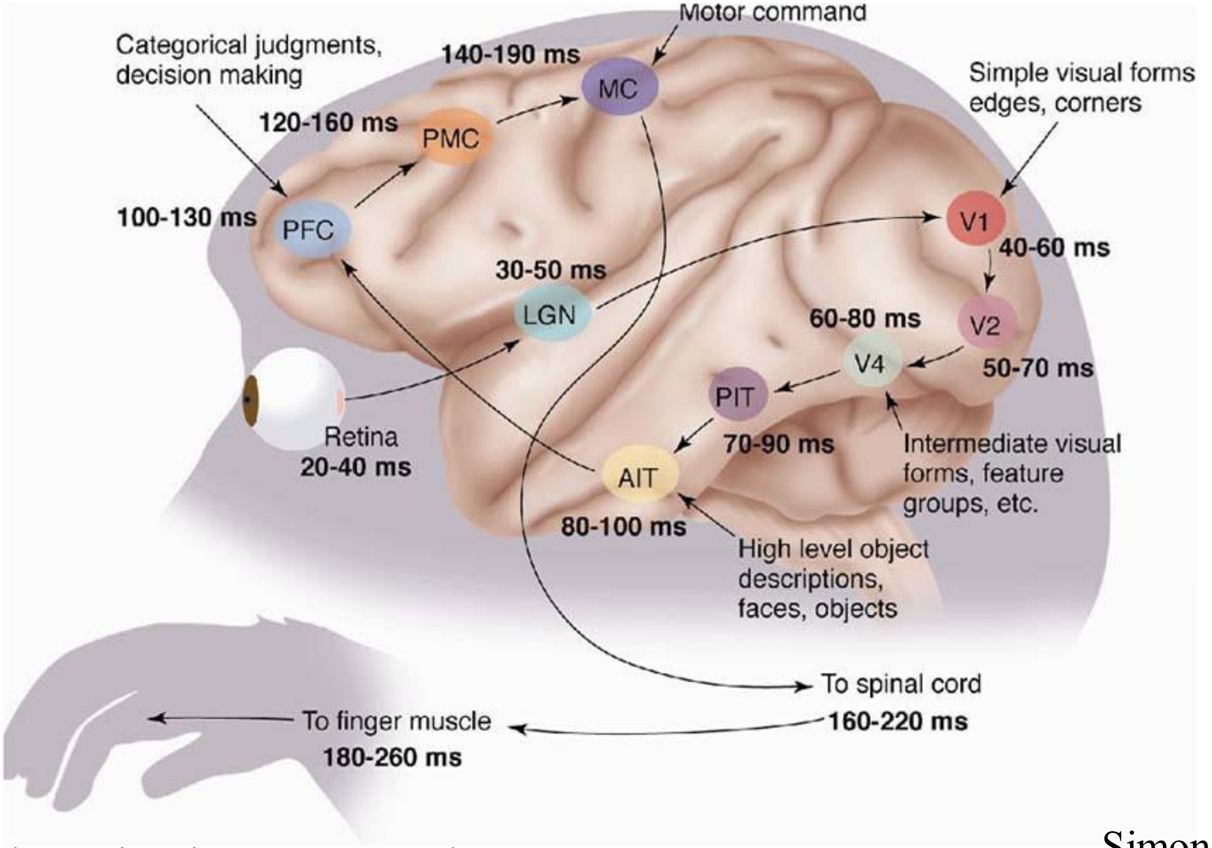
What Is CNN

- An extension of MLP
  - *More hidden layers* to extract *features* of signal (image, speech)
  - Depth of network is "deep"
- CNN is one famous model of **deep neural networks (DNN)**

CNN



Why CNN? Human vision is fast



Experiments show that humans can detect animals in a scene in less than 150 ms.

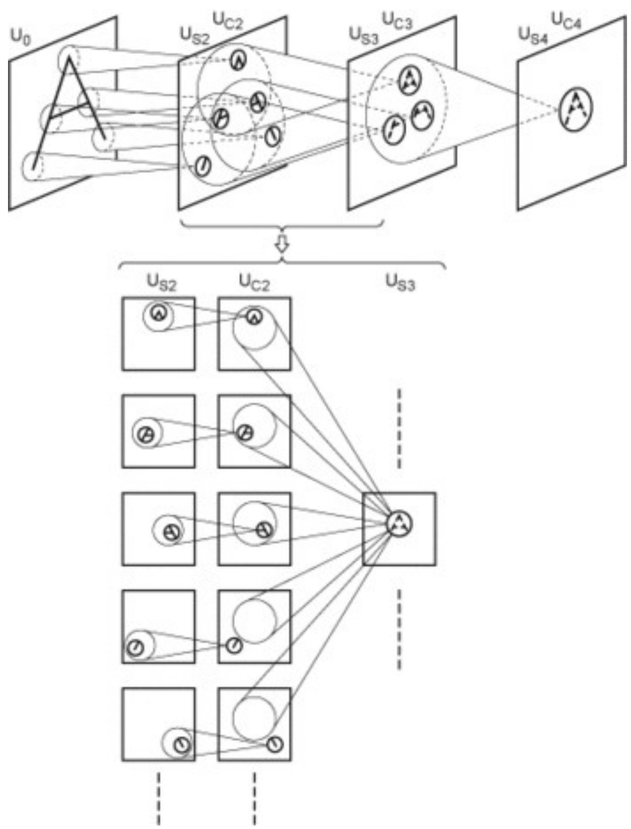
Simon Thorpe et al., "Seeking Categories in the Brain," Neuroscience 2001.

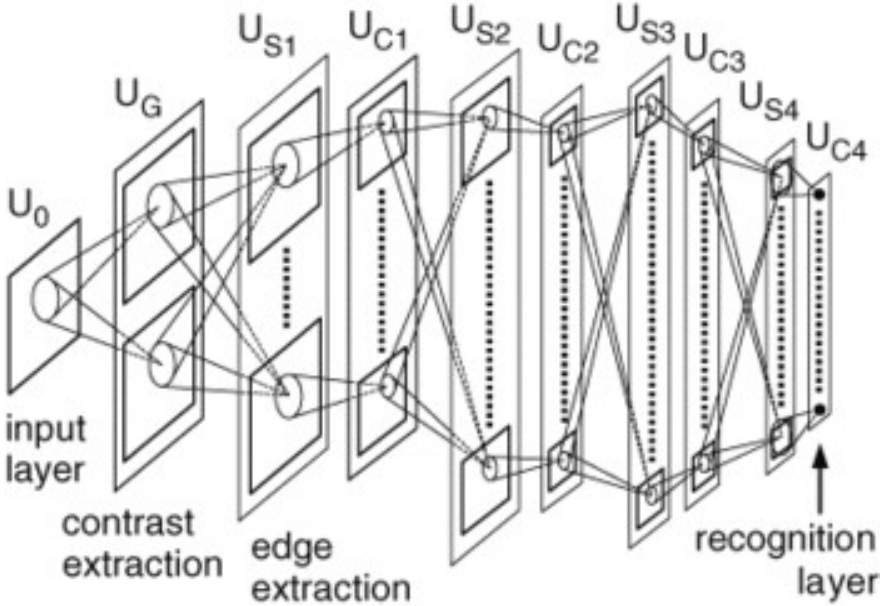
Hubel & Wiesel (1962)

Insights about early image processing in the brain

- Simple cells detect local features
- Complex cells pool local features in a retinotopic neighborhood

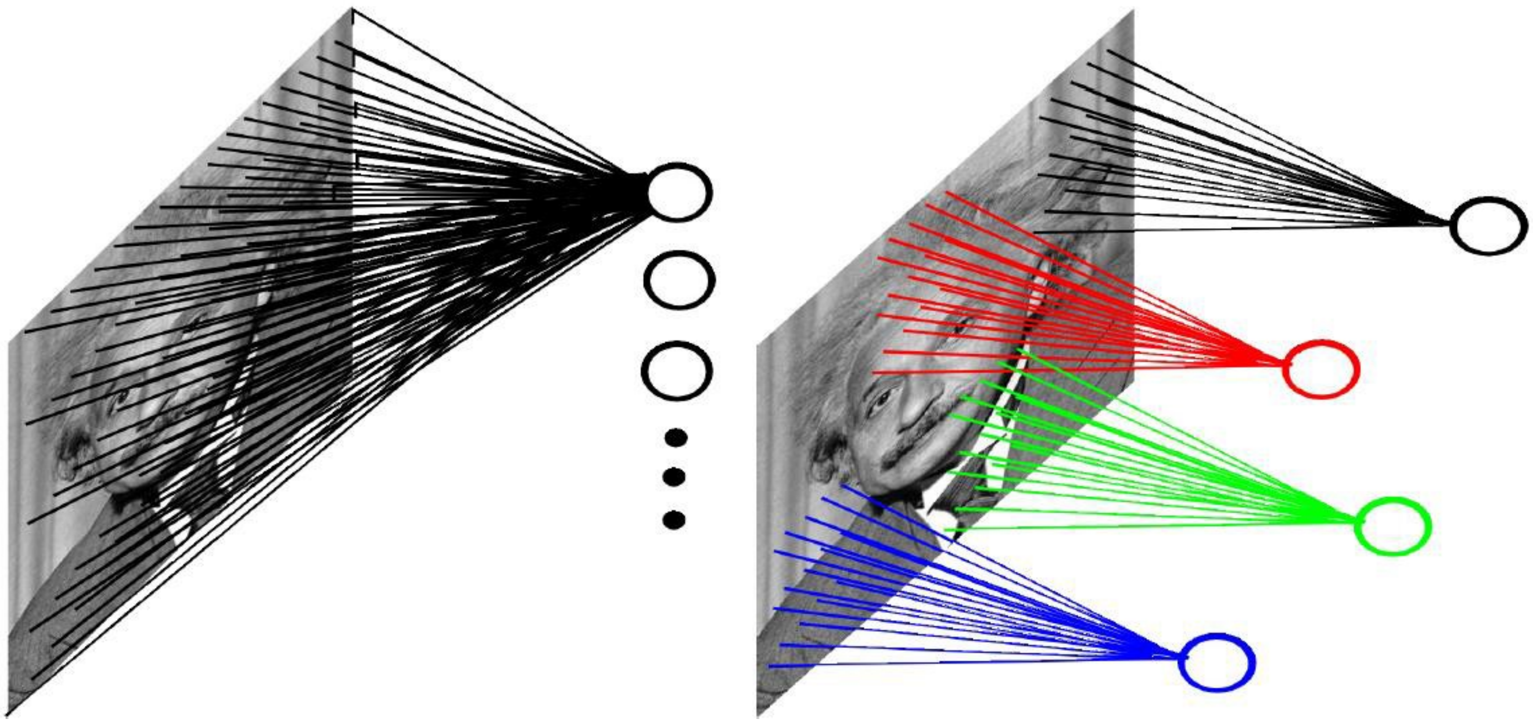
The Neocognitron



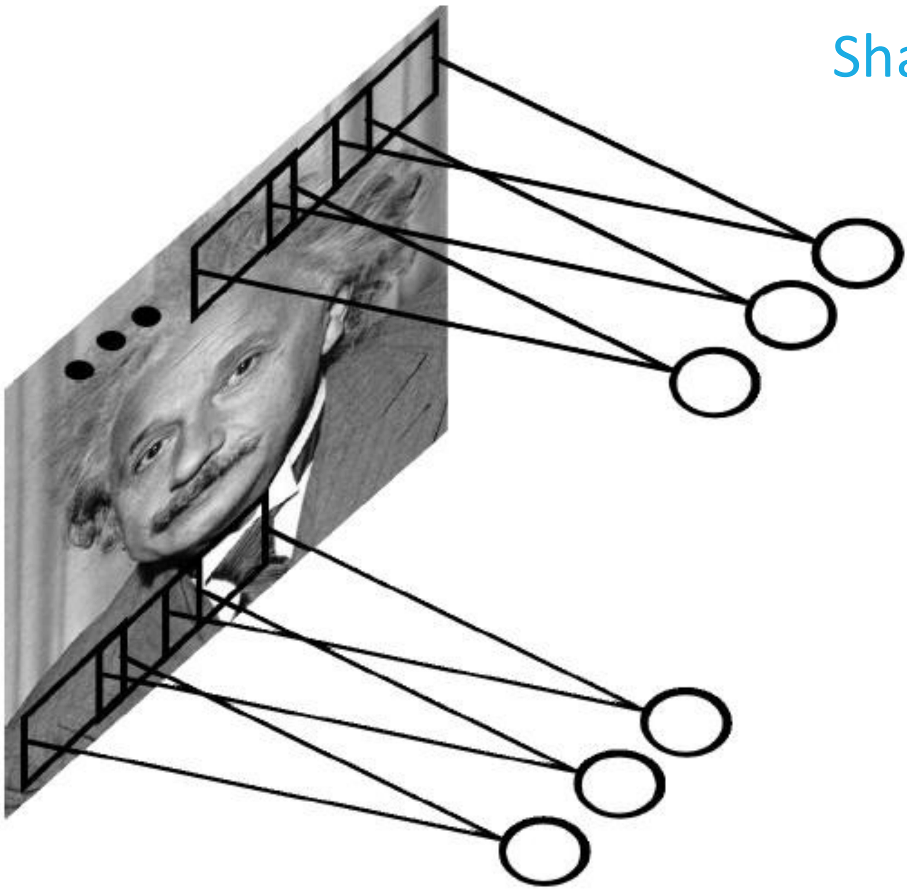


(Fukushima 1974-1982)

Local connections

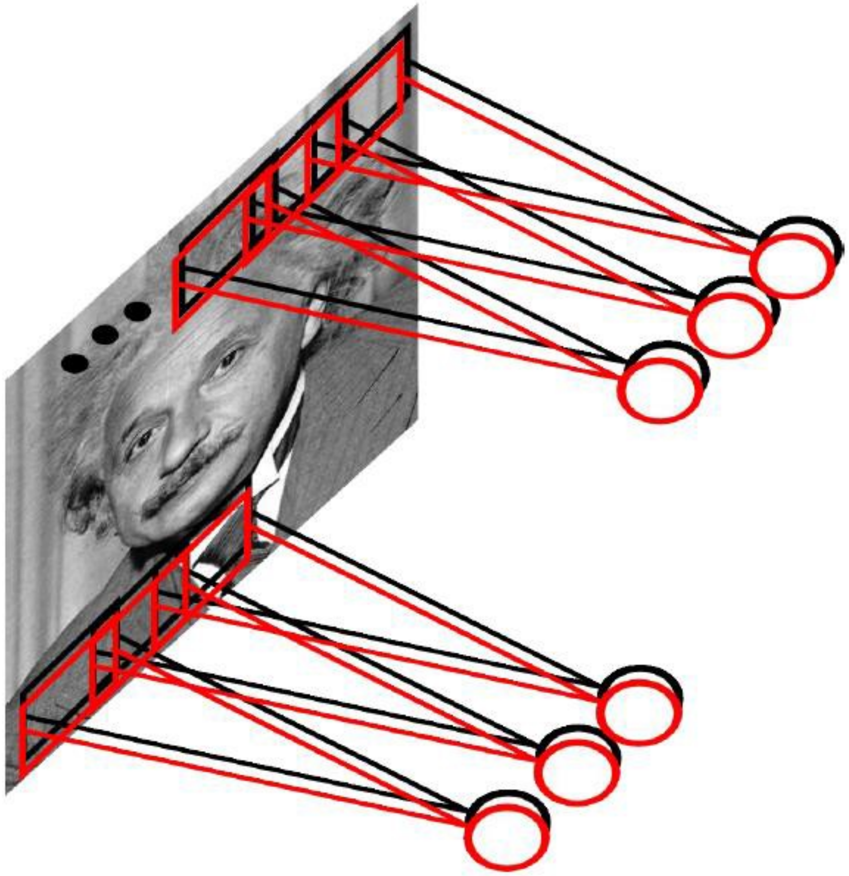


Convolution



Shared weights

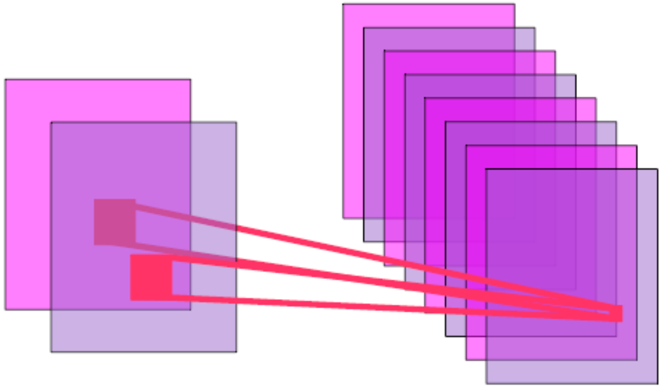
Multiple convolutions



CNN:

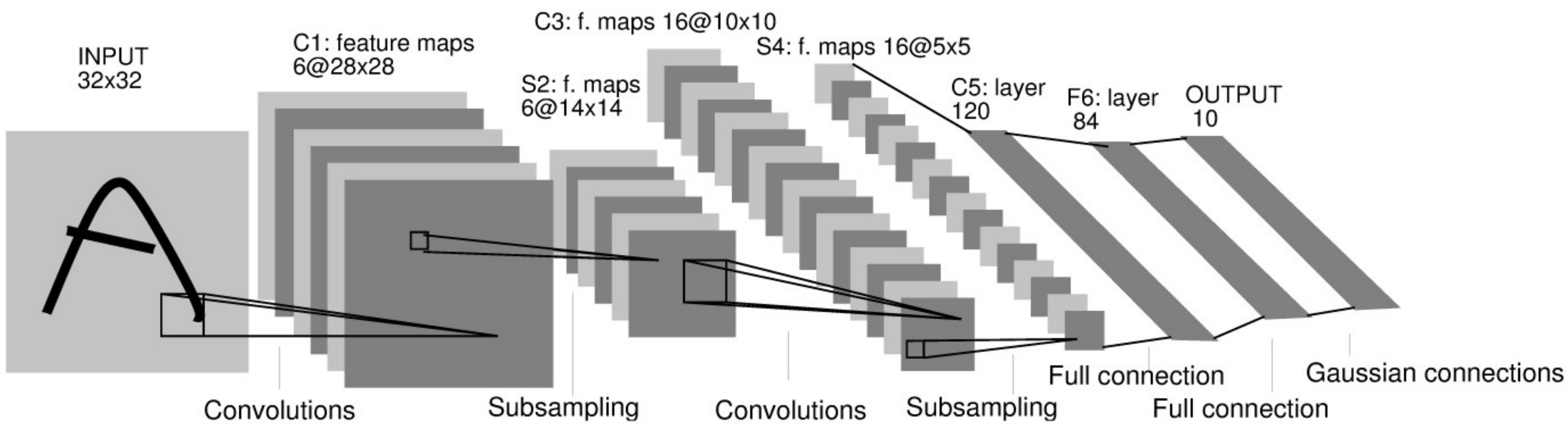
Convolutional neural network

ConvNet

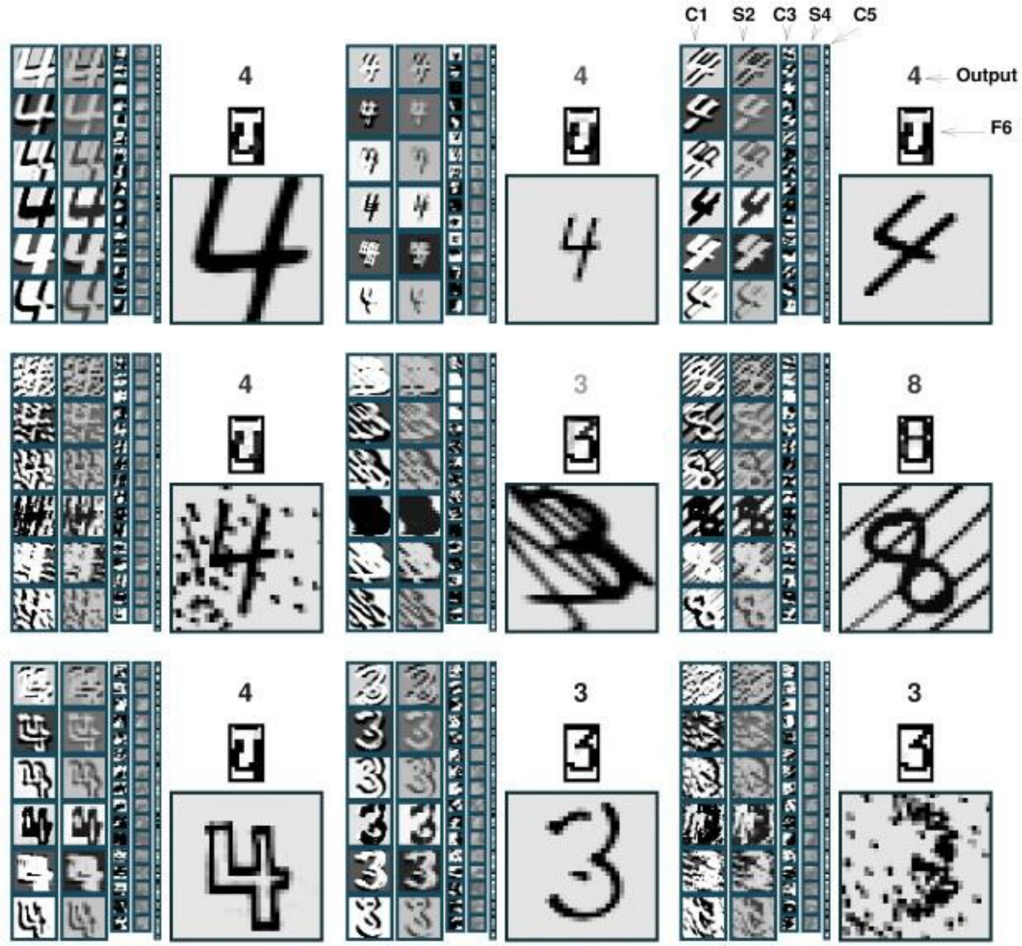


CNNs in the 1990s

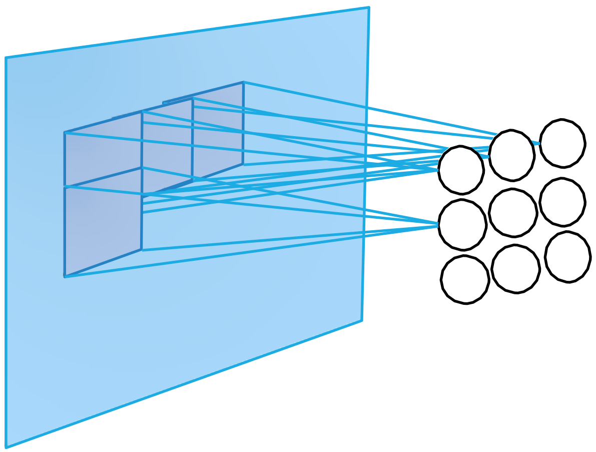
- 1989 Isolated handwritten character recognition (AT&T Bell Labs)
- 1991 Face recognition. Sonar image analysis. (Neuristique)
- 1993 Vehicle recognition. (Onera)
- 1994 Zip code recognition (AT&T Bell Labs)
- 1996 Check reading (AT&T Bell Labs)



Convnets in the 1990s



Pooling



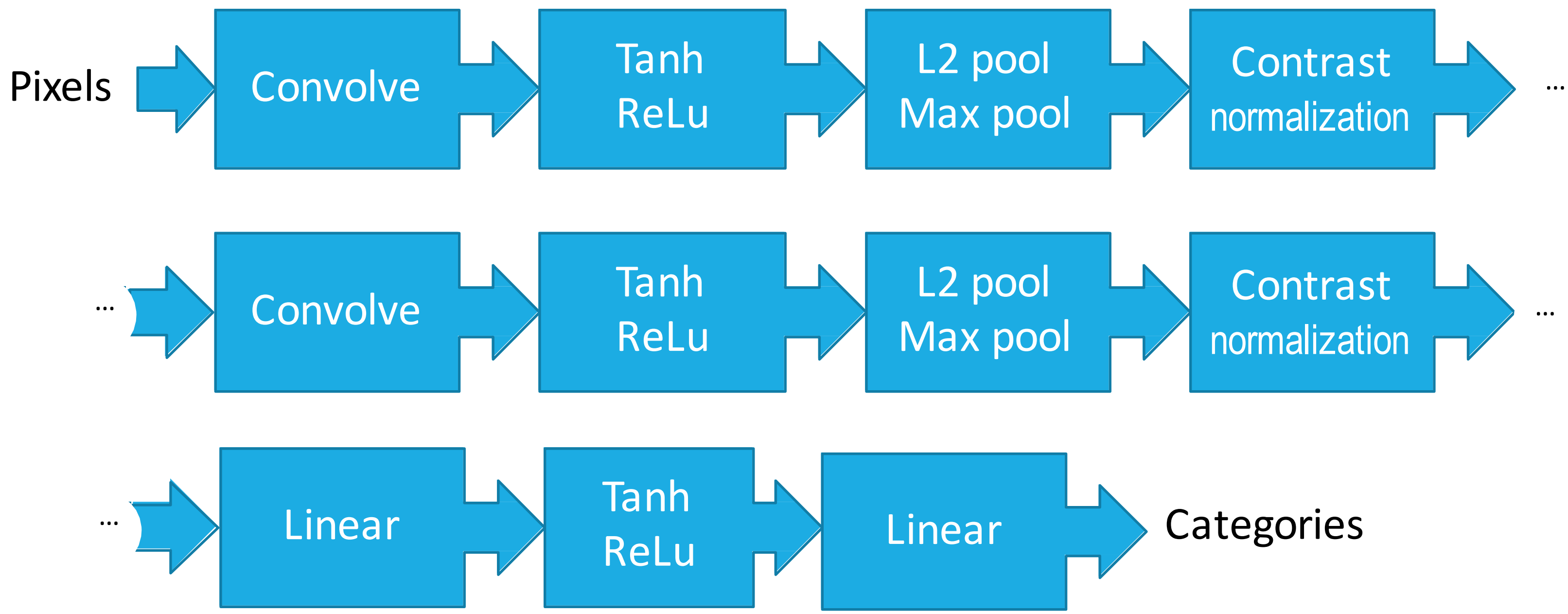
Name	Pooling formula
Average pool	$\frac{1}{s^2}\sum x_i$
Max pool	$\max\{x_i\}$
L2 pool	$\sqrt{\frac{1}{s^2}\sum x_i^2}$
L_p pool	$\left(\frac{1}{s^2}\sum \lvert x_i\rvert^p\right)^{\frac{1}{p}}$

Contrast Normalization

Contrast normalization

- Subtracting a low-pass smoothed version of the layer
- Just another convolution in fact (with fixed coefficients)
- Lots of variants (per feature map, across feature maps, …)
- Divisive normalization

CNNs in the 2010s

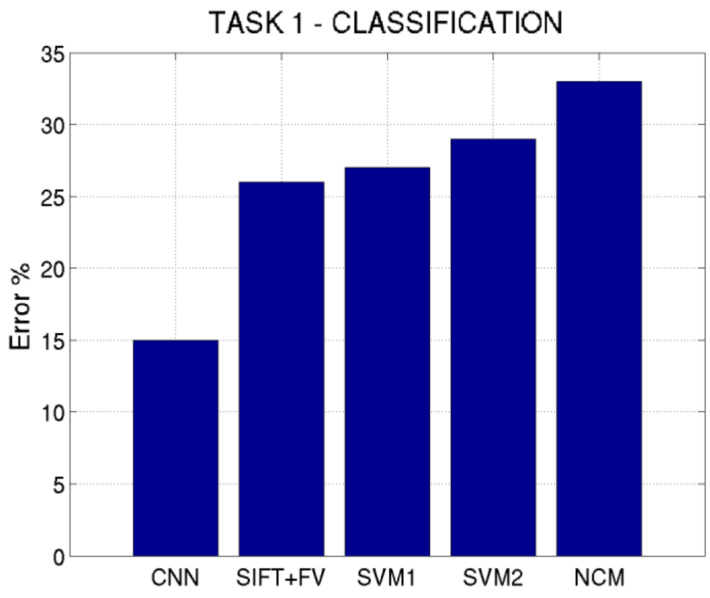


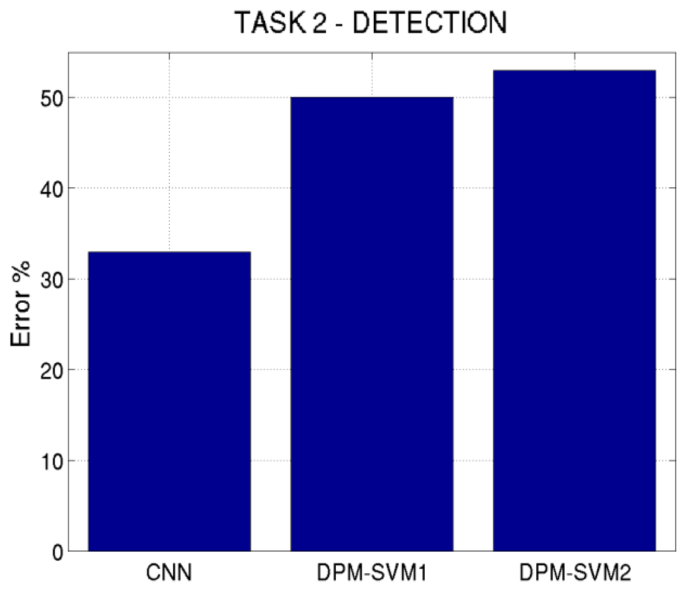
Convnets in the 2000s

- OCR in natural images [2011]. Streetview house numbers (NYU)
- Traffic sign recognition [2011]. GTRSB competition (IDSIA, NYU)
- Pedestrian detection [2013]. INRIA datasets (NYU)
- Volumetric brain segmentation [2009]. Connectomics (MIT)
- Human action recognition [2002,2011]. Smartcatch (NEC), Hollywood II (SF)
- Object recognition [2004,2012]. Norb (NEC), ImageNet (UofT)
- Scene parsing [2010-2012]. Stanford bldg, Barcelona (NEC, NYU)
- Medical image analysis [2008]. Cancer detection (NEC)

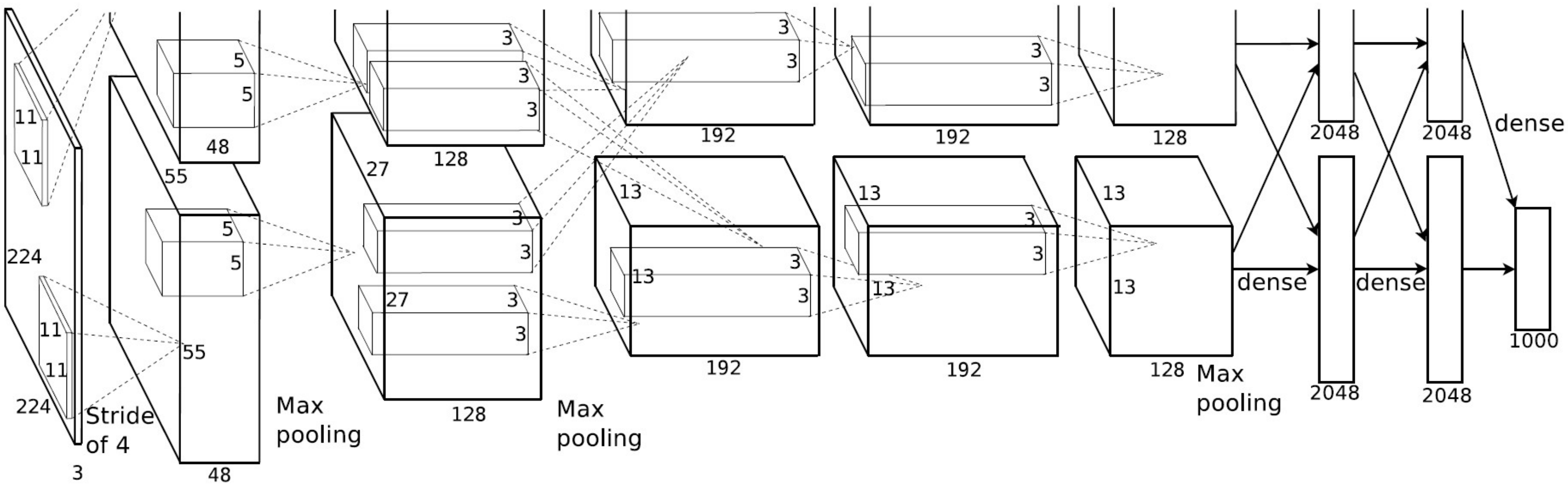
ImageNet 2012 competition

- Object recognition. 1000 categories. 1.2M examples





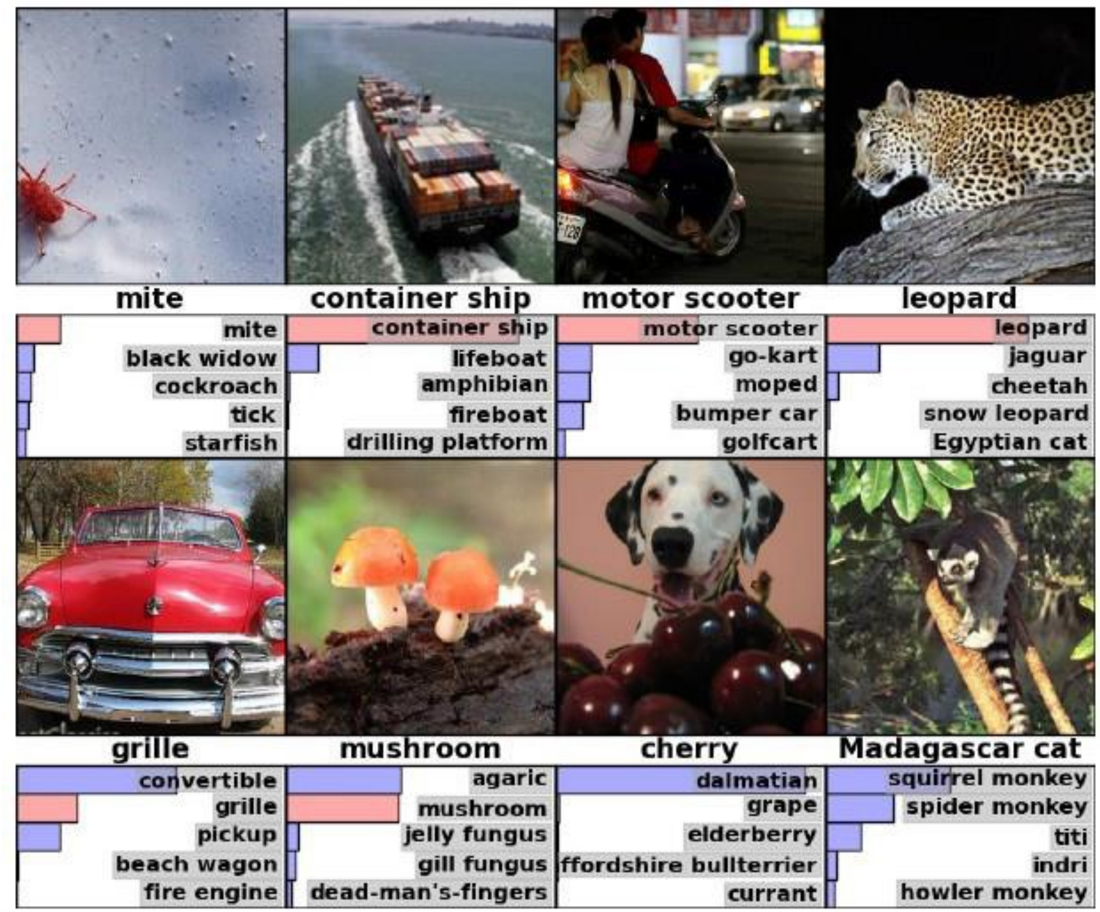
ImageNet CNN



- Structure (conv-relu-maxpool-norm)^3-linear-relu-linear-relu-linear
- Very good implementation, running on two GPUs.
- ReLU transfer function. Dropout trick.
- Also trains on full ImageNet (15M images, 15000 classes)

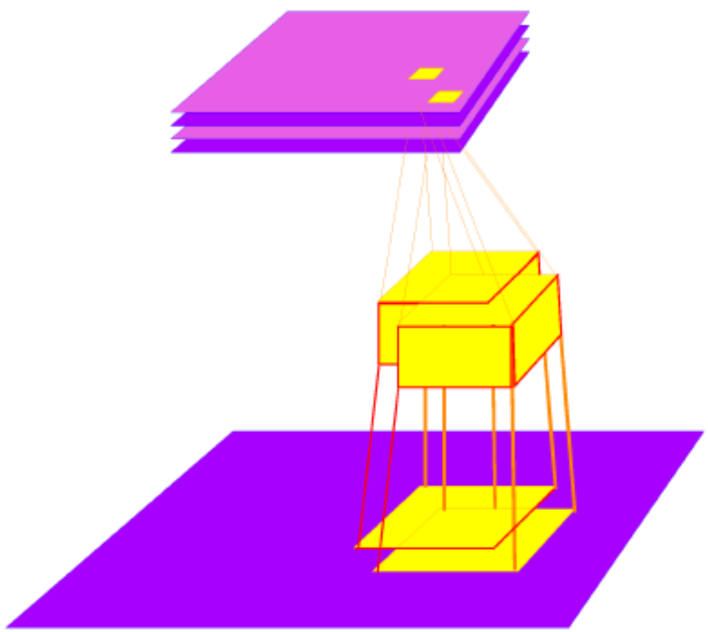
(Kirzhevsky, Sutskever, Hinton, 2012)

ImageNet CNN

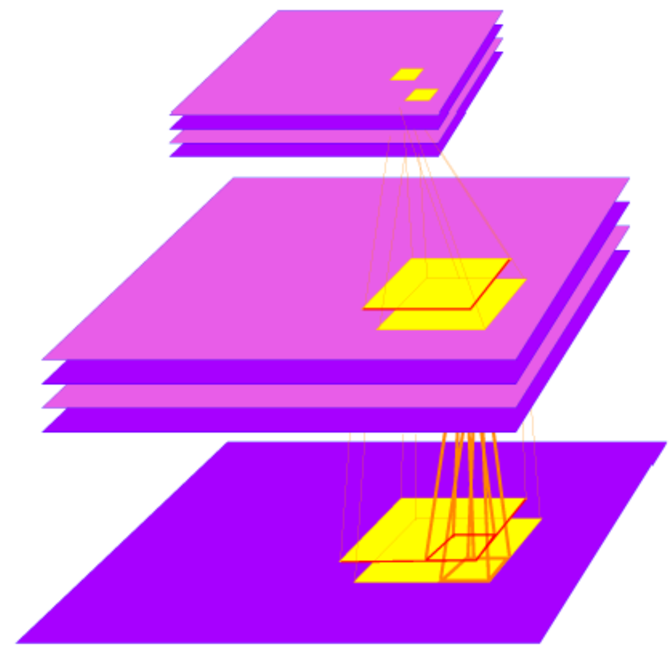


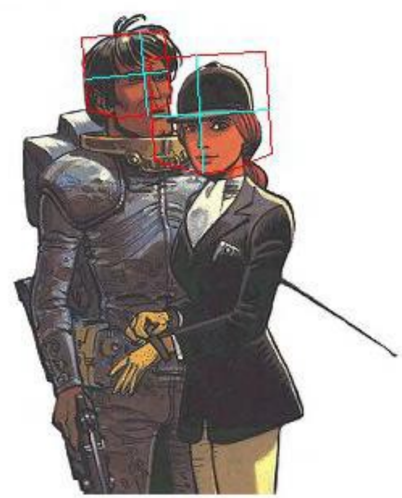
Replicated CNNs

Wrong way



Right way





Today :
Neural Information Processing

- Origins
- Perceptron
- Multilayer perceptron (MLP)
- Convolutional neural network (CNN)

Next :

Training multilayer networks

- Perceptron learning
- Optimization basics
- Stochastic gradient descent
- Backpropagation learning algorithms