

AI — Some Theoretical Basics

Dipl.-Inf. Markus Sing
markus.sing@gmx.ch

April 26th, 2026

Special thanks to Christoph Haberer

Brief History

AI research starting in the fifties of last century

2 main directions:

- “copy” human brain → neural networks
 - ⚠ needs very much computing power
 - short coming up, AI winter, big coming back ...
- logic-based → expert systems
 - hype in the late eighties of last century

Logic-based (excursus)

Prolog (programming language)

```
% clauses: facts
mother_child(trude, sally).
father_child(tom, sally).
father_child(tom, erica).
father_child(mike, tom).

% clauses: rules
parent_child(X, Y) :-
    father_child(X, Y).

parent_child(X, Y) :-
    mother_child(X, Y).

sibling(X, Y) :-
    parent_child(Z, X),
    parent_child(Z, Y),
    not(X = Y).

ancestor(X, Y) :-
    parent_child(X, Y).

ancestor(X, Y) :-
    parent_child(X, Z),
    ancestor(Z, Y).

% queries
?- sibling(sally, erica).
Yes
?- mother_child(trude, X).
X = sally
```

programs work in both directions :-)

Example based on <https://en.wikipedia.org/wiki/Prolog>

Basic Perceptron

Perceptron (Frank Rosenblatt, 1957): simplified artificial neuronal network

basic version: one artificial neuron

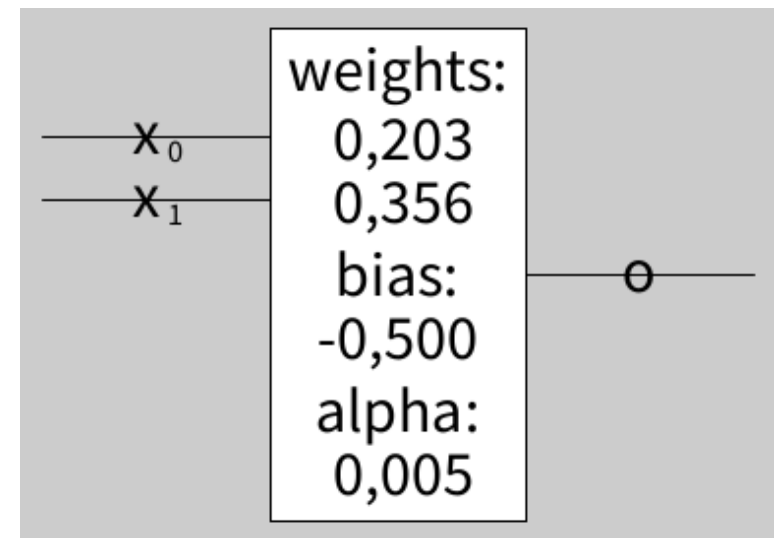
$$o = \begin{cases} 1, & \sum_i w_i \cdot x_i + b > 0 \\ 0, & \text{else} \end{cases}$$

w_i : weights, x_i : inputs, b : bias, o : output

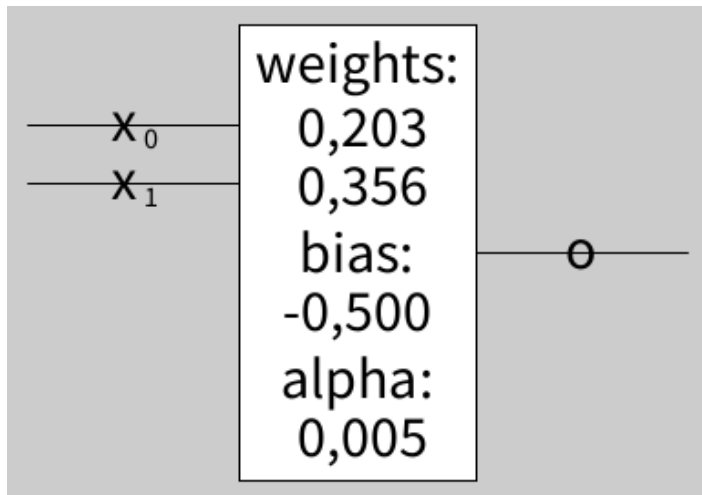
learning = “correctly” adjusting weights

$$w_i^{\text{new}} = w_i^{\text{old}} + \underbrace{\alpha \cdot (t - o) \cdot x_i}_{\Delta w_i}$$

α : learning rate, t : target (correct output)

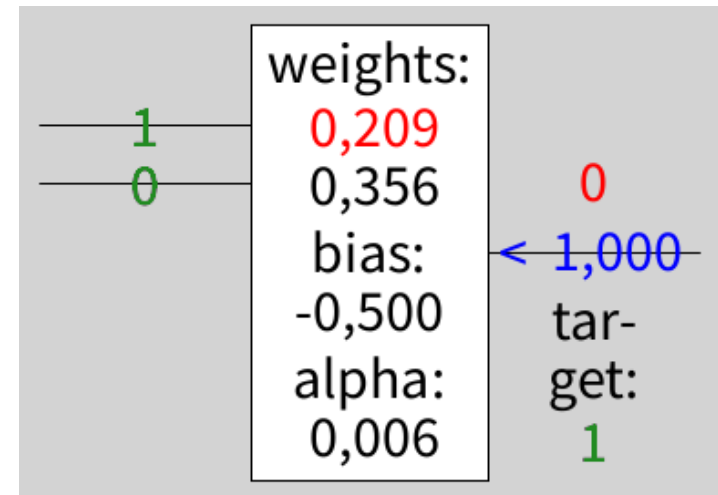


Train it (1)



OR:

x_1	x_0	t
0	0	0
0	1	1
1	0	1
1	1	1



Train it (2) / limitations of the basic perceptron

AND:

x_1	x_0	t
0	0	0
0	1	0
1	0	0
1	1	1

XOR:

x_1	x_0	t
0	0	0
0	1	1
1	0	1
1	1	0

?

⚠ sometimes it does not converge, instead it oscillates

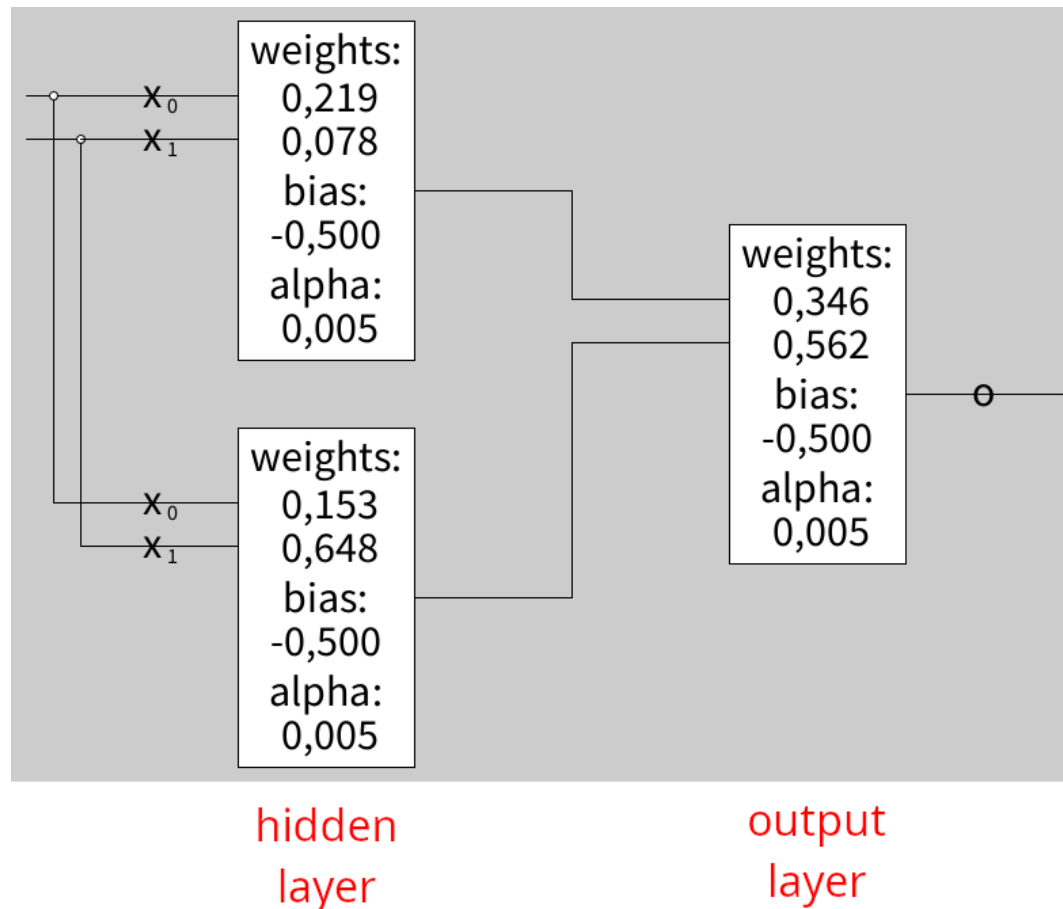
The following might help:

- sending training data not round robin
- varying alpha

But not in all cases — esp. XOR. :-)

Multi-Layer-Perceptron

several artificial neurons in multiple layers



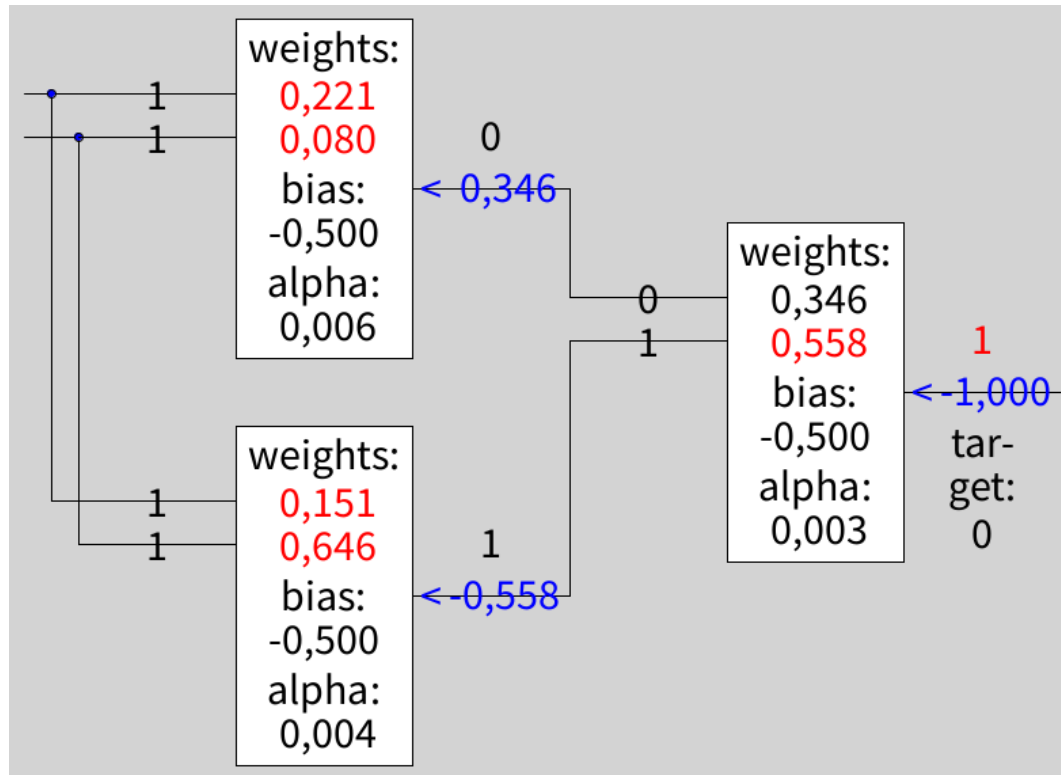
Train it:

$$w_i^{\text{new}} = w_i^{\text{old}} + \underbrace{\alpha \cdot (t - o)}_{\Delta w_i} \cdot x_i$$

– output layer: :-)

– hidden layer(s): t ?

Backpropagation



- use input, calculate forward
- calculate error behind (and adapt weights there)
- propagate error in a clever way back to front (and adapt weights there)

Multi-Layer: different network types

- feedforward neural network
output can't be input of ahead layers
⚠ required for backpropagation
- recurrent neural network
output can be input of ahead layers
private crazy idea: make sound with this?

Training data: it's getting great ...

⚠ for each input combination, there is a target value

- 2 inputs → 4 input combinations (→ 16 target functions)

possible input com- binations	each column: possible target function (training uses only one of them)						
0, 0	0	0	...	0	0	...	1
0, 1	0	0	...	1	1	...	1
1, 0	0	0	...	1	1	...	1
1, 1	0	1	...	0	1	...	1
		AND		XOR	OR		

- 3 inputs → 8 input combinations (→ 256 target functions)
- 4 inputs → 16 input combinations (→ 65536 target functions)
- n inputs → 2^n input combinations (→ $2^{(2^n)}$ target functions)

text input, just 50 letters, ...

Supervised vs. Unsupervised Learning

as above, with well known target function: supervised learning

“no” explicit target function: unsupervised learning:

search for even unknown correlations between input data

“target”: optimization based on some metric

- segment into clusters
- find outliers
- find principal components (to simplify by dropping the rest)

Unsupervised Learning: Find Principal Components

ship	length	width	origin
Queen Mary 2	345m	41m	2003
AidaBlu	253m	32m	2010
Mein Schiff 5	295m	35m	2016
Icon of the Seas	395m	48m	2023
⋮	⋮	⋮	⋮

→ just one of them

Principal Component Analysis (PCA)

- from statistics (Pearson, 1901)
- uses also a lot of linear algebra

Principal Component Analysis (PCA)

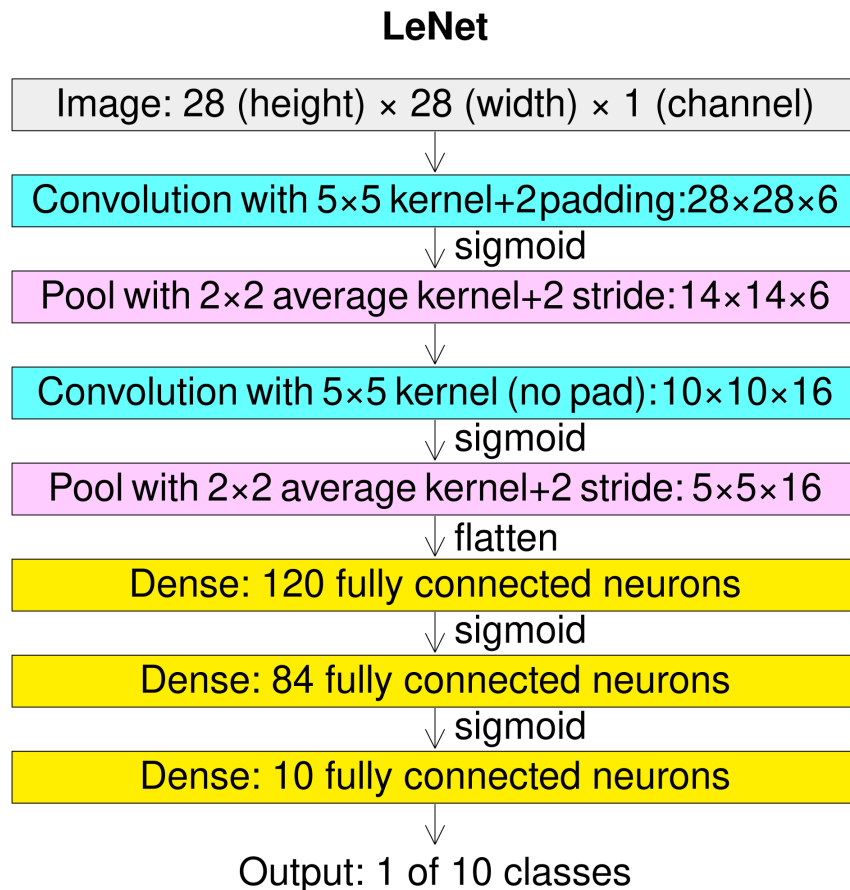
n points in a coordinate system of dimension p (e.g. data of n ships concerning p characteristics)

3 steps:

- translation: new coordinate system with origin = “middle” of data
- rotation: new coordinate system with
 - along 1. axis largest variance
 - along 2. axis 2nd largest variance
 - \vdots(via diagonalizing covariance matrix)
- projektion: reduce dimensions

Convolutional Neural Network

advanced architecture with specialized feedforward subnetworks with specialized aggregation of their results

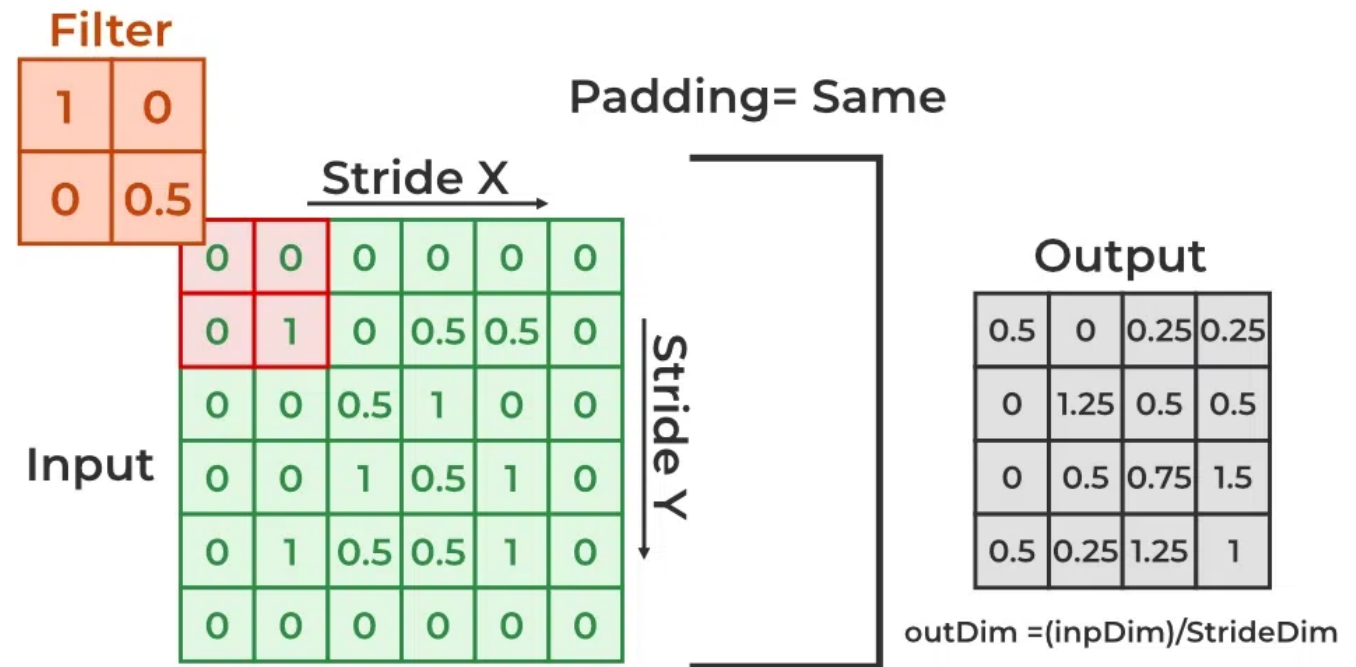


- **convolution** layers:
combine 2 functions to 1 function
 - **pooling** layers:
reduce dimensions of data
(take max or average of a cluster)
 - **dense** layers:
fully connected neurons
(sigmoid function)
- ⚠ anything with multiple layers:
deep learning (DL)

picture source: https://en.wikipedia.org/wiki/Convolutional_neural_network

Convolution with Kernel and Padding

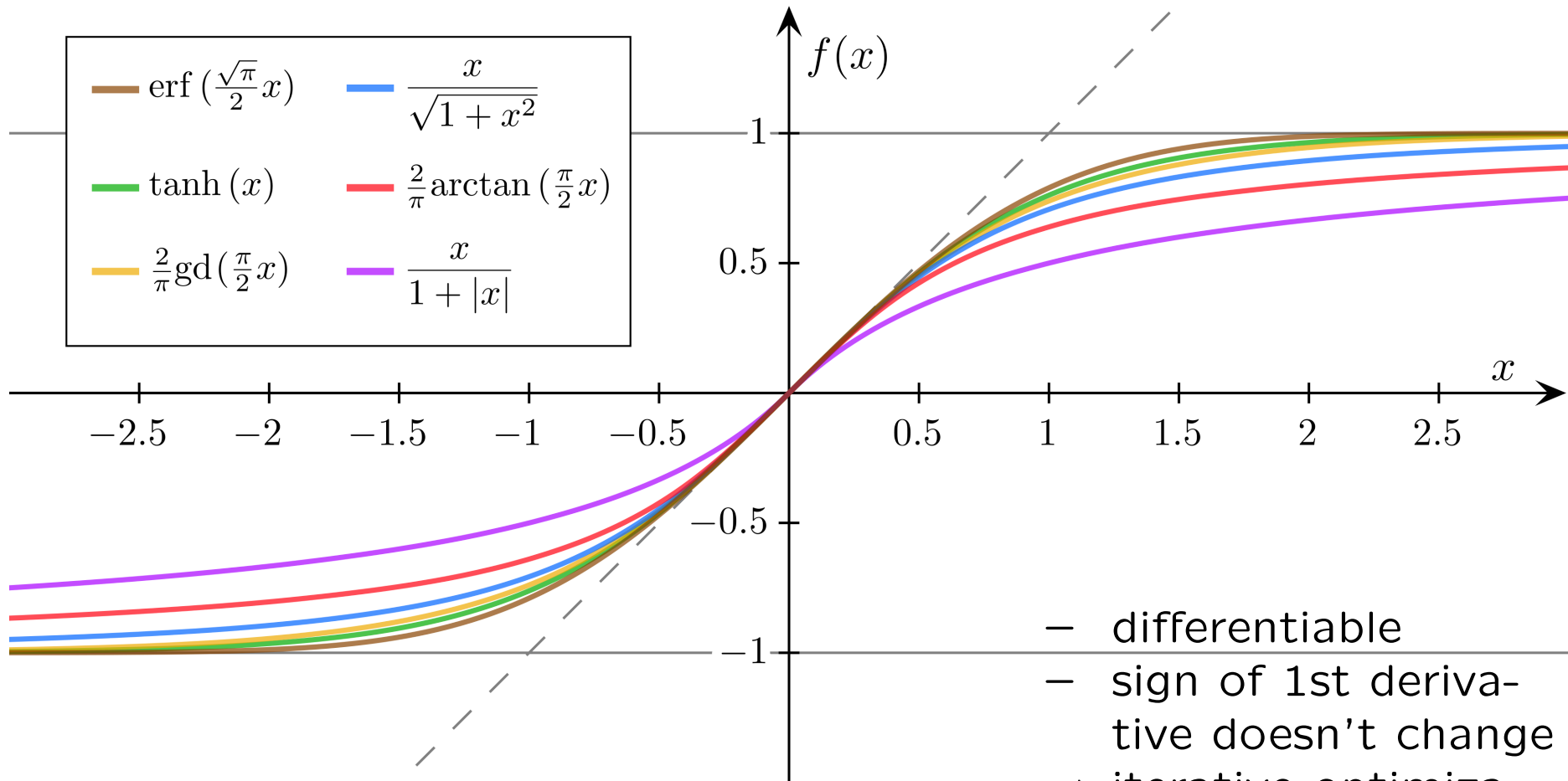
- input: 4×4
- padding:
adding the zeros around input
- convolution:
applying 2×2 kernel (= filter), i.e. sum of all products (input, filter at corresponding position)
- output: 4×4



picture source:

<https://www.geeksforgeeks.org/machine-learning/cnn-introduction-to-padding/>

Sigmoid functions

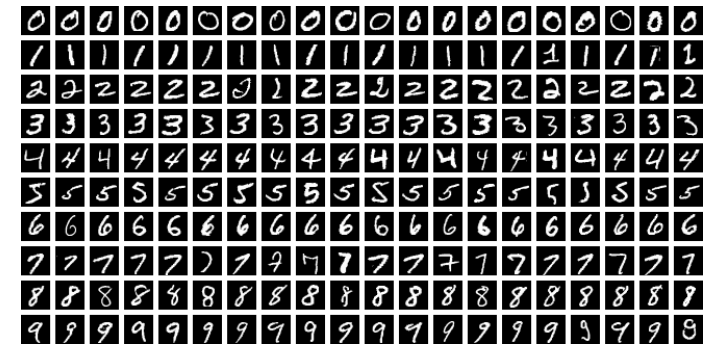


picture source: <https://de.wikipedia.org/wiki/Sigmoidfunktion>

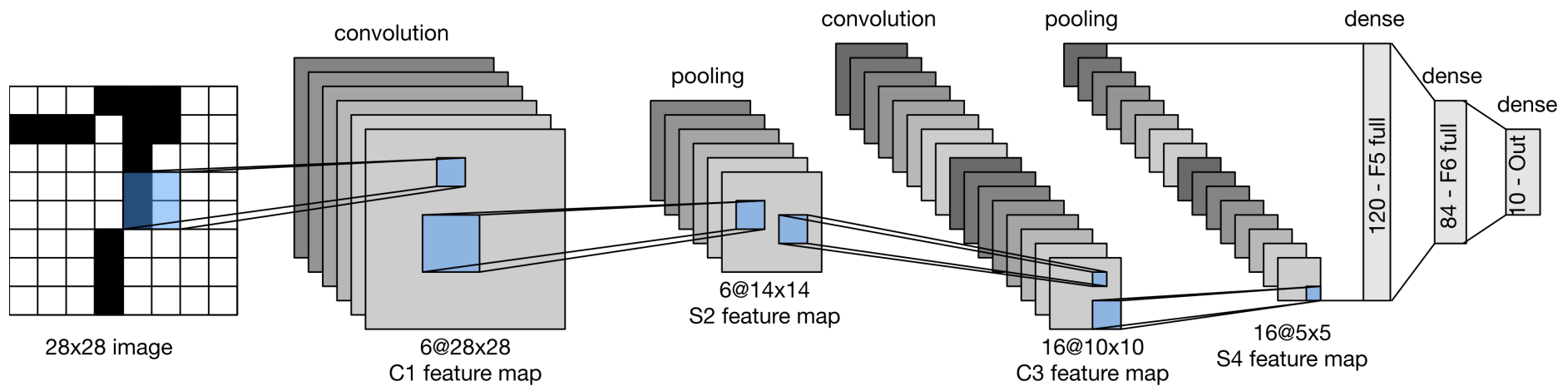
LeNet

LeNet:

- 1988-1998, AT&T Bell Laboratories
- to identify handwritten digits and letters, esp. in banking applications



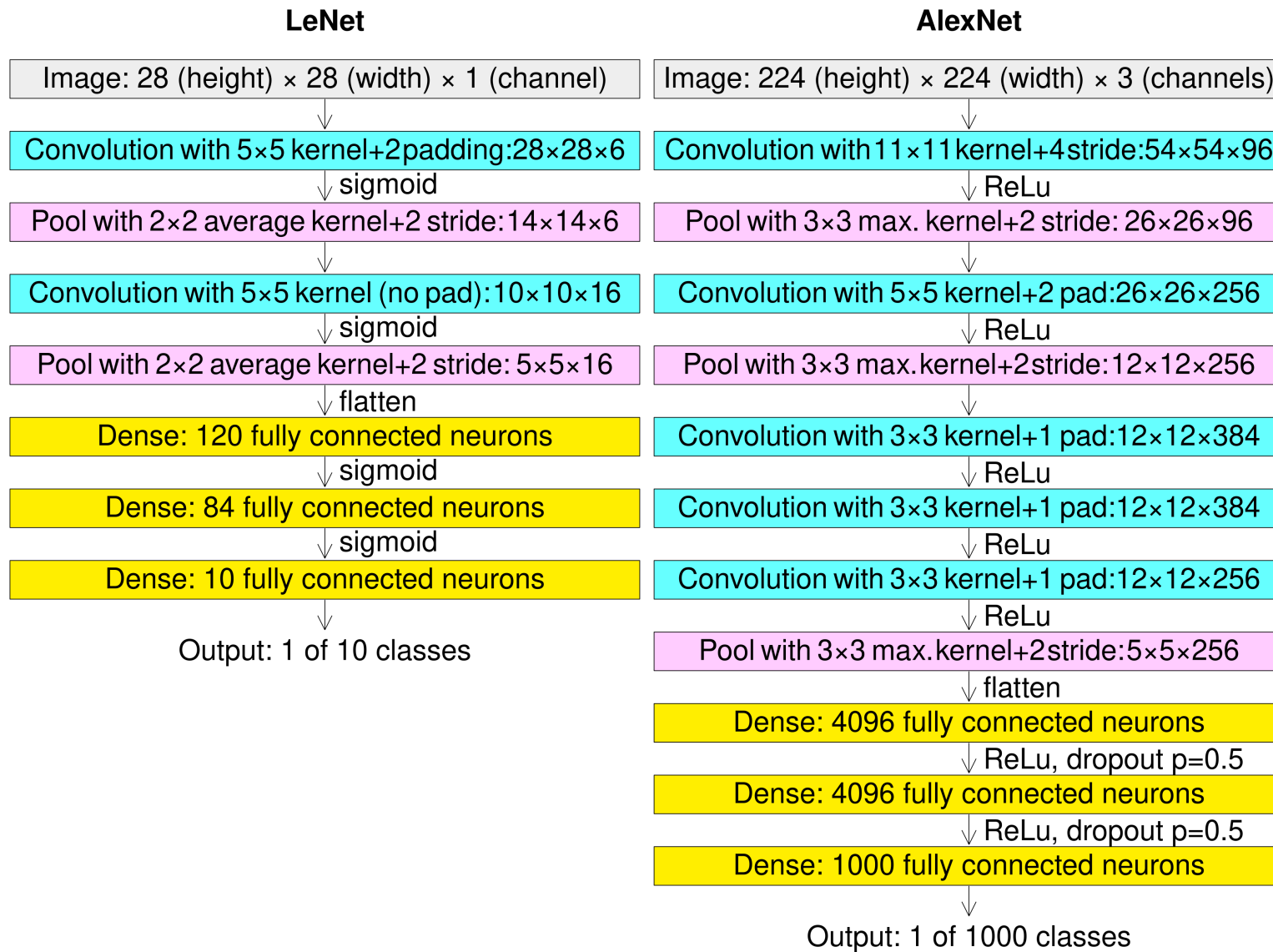
Training data



Architecture

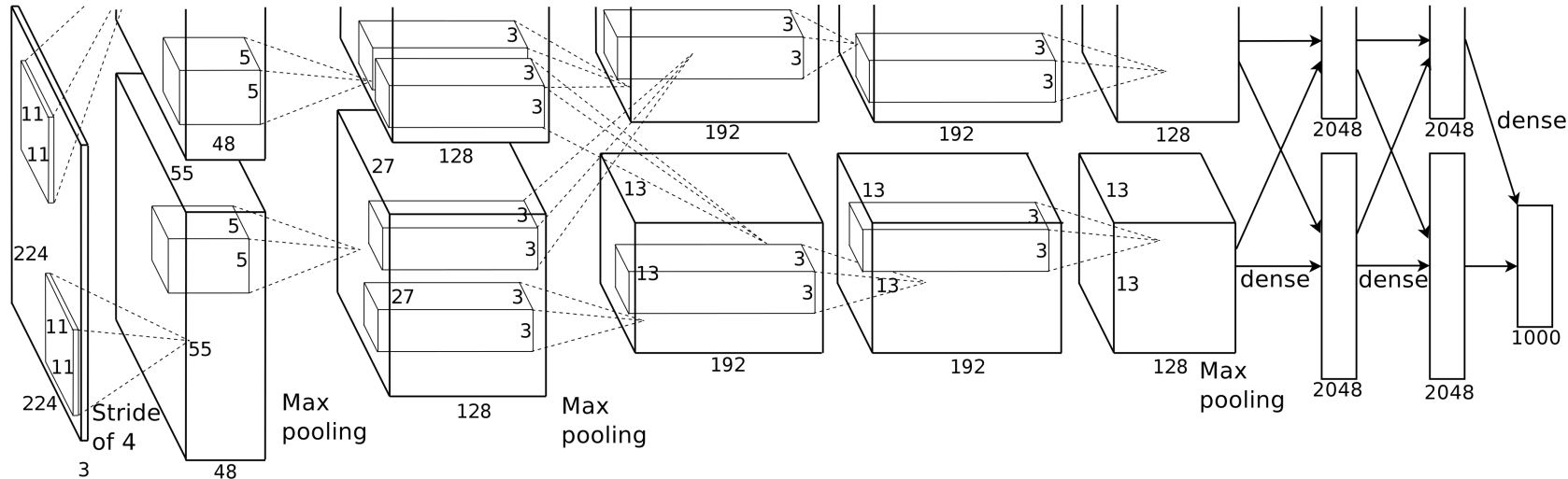
picture sources: <https://en.wikipedia.org/wiki/LeNet>

LeNet (1995) / AlexNet (2012)



source: https://en.wikipedia.org/wiki/Convolutional_neural_network

AlexNet

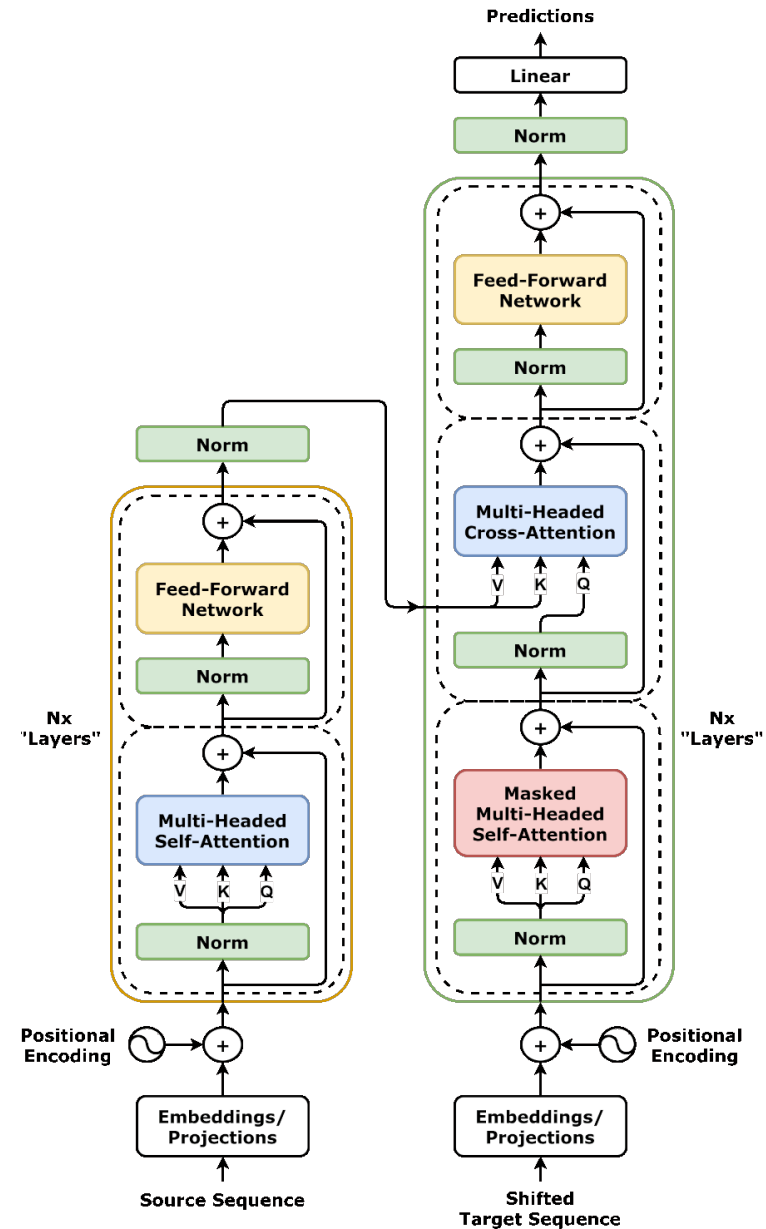


picture source: <https://en.wikipedia.org/wiki/AlexNet>

- 2012, University of Toronto
- for image classification tasks, wins a contest and influenced a large number of subsequent work
- training set: 1.2 million images, training for 90 epochs, 5 to 6 days
2 Nvidia GPUs (split up due to memory requirements)
each forward pass of AlexNet required approximately 1.43 GFLOPs
(theoretically 2,200 forward passes per second possible)

Large Language Model (LLM)

- for natural language processing tasks
- trained on a vast amount of text (→ copyright?)
- based on transformer architectures



Picture source:

[https://en.wikipedia.org/wiki/Transformer_\(deep_learning\)](https://en.wikipedia.org/wiki/Transformer_(deep_learning))

transformer architecture