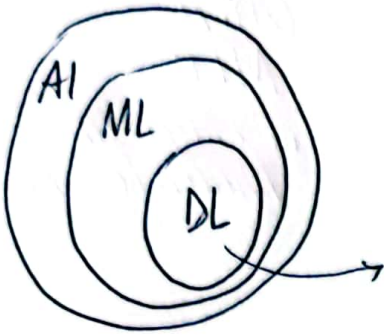
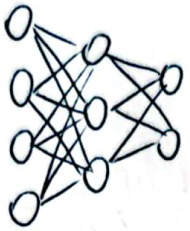
# NEURAL NETWORKS CRASH COURSE

Advanced topic - Programming Wise 23-24

Luna Planesi

## ▷ WHAT IS DEEP LEARNING?

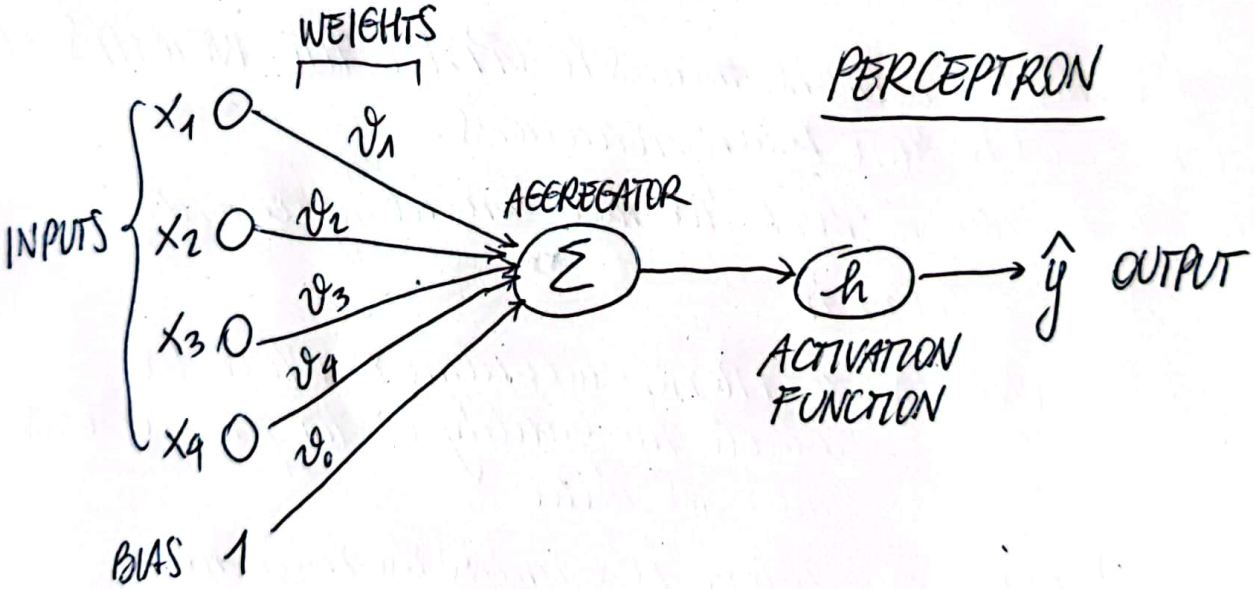


Deep learning is a subset of machine learning focusing on artificial neural networks. "Deep" refers to the many layers constituting neural networks.

## ▷ WHAT IS A NEURAL NETWORK?

A neural network is a NONLINEAR, PARAMETERISED FUNCTION WITH RESTRICTED OUTPUT RANGE.



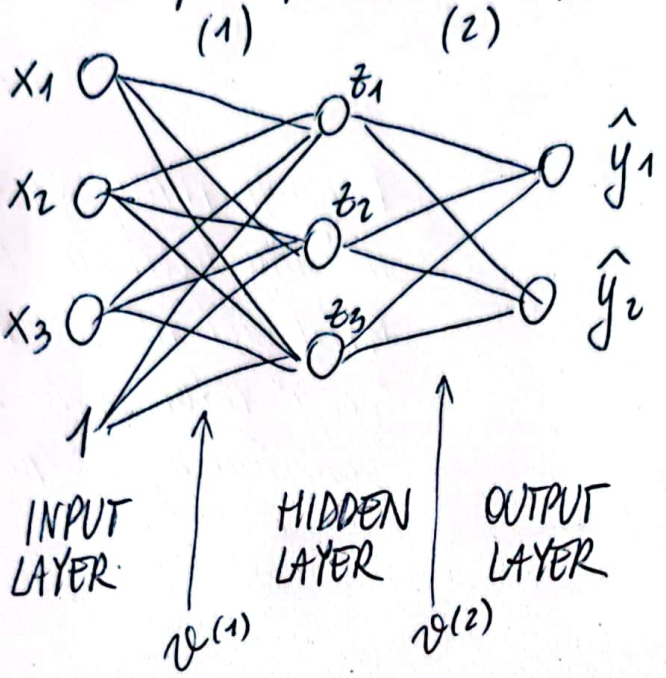

Neural network components:

- INPUTS: images, numbers, text, ...
- WEIGHTS: the parameters governing the function
- BIAS: added constant to offset results
- AGGREGATOR: function to aggregate computations
- ACTIVATION FUNCTION: non-linear function through which the input has to pass

This version of neural network is called PERCEPTRON.

## ▷ MULTI-LAYER PERCEPTRON (MLP)

We can combine more perceptrons to "deepen" our network:




SINGLE LAYER MLP (we only count hidden layers)

MLPs can approximate more complex functions than perceptrons.

## ▷ TRAINING A NEURAL NETWORK

Training a neural network means to UPDATE THE WEIGHTS of the network so as to have better predictions.

To train a neural network we need several ingredients:

- DATA: a lot of it
- LOSS FUNCTION: a criterion according to which we evaluate the quality of the predictions that we make
- OPTIMIZER: a strategy to optimize the training
- BACKPROPAGATION: a way to exploit the errors that we make during training to "teach" the network to perform better

## BACKPROPAGATION:

To "back propagate" the error signal obtained during training to update the weights of the network.

### GENERAL TECHNICAL FORMULATION OF BACKPROPAGATION:

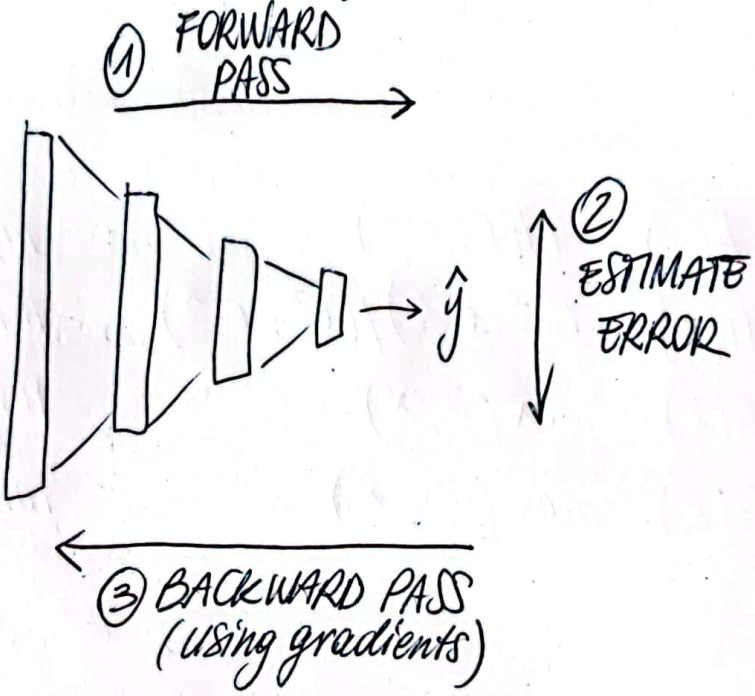
Let $W_{i \rightsquigarrow j}$ be the weights from node $i$ to node $j$.

Then:

$$\frac{\partial L}{\partial W_{i \rightsquigarrow j}} = \delta_j z_i$$

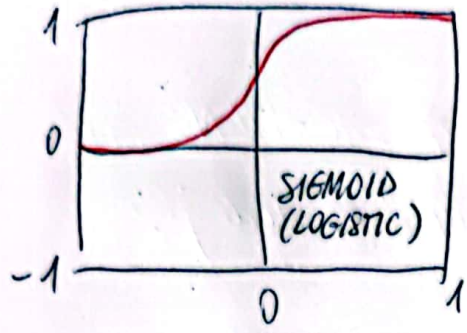
where $\delta_j$ is the LOCAL ERROR (going from $j$ backwards) and $z_i$ is the LOCAL INPUT coming from $i$.

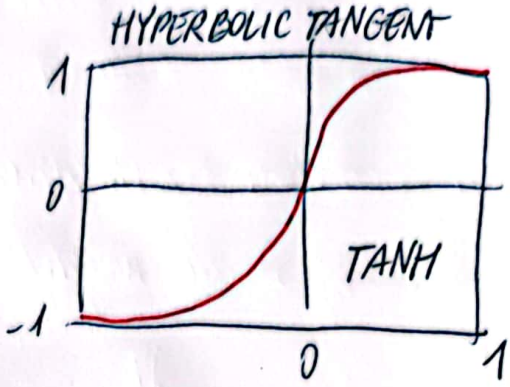
To train a network means to repeat steps ①, ② & ③ over and over again for a certain number of times, called EPOCHS.

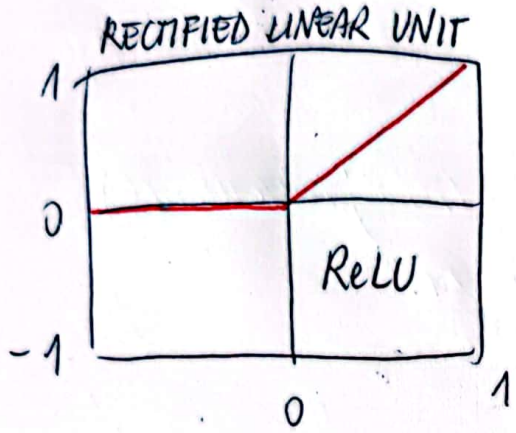


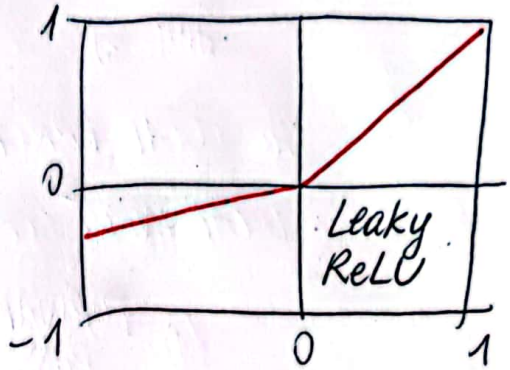

Now let's see some of the elements that we mentioned earlier in more detail.

## ▷ ACTIVATION FUNCTIONS




| | | |
|---|---|---|
| SIGMOID | $h(x) = 1/(1+e^{-x})$ | range : $[0,1]$ |
| TANH | $h(x) = (e^{x}-e^{-x})/(e^{x}+e^{-x})$ | range : $[-1,+1]$ |
| ReLU | $h(x) = \max(0,x)$ | range: $[0,+\infty)$ |
| L. ReLU | $h(x) = \max(\alpha x, x)$ | range : $(-\infty,+\infty)$ |

## ▷ LOSS FUNCTIONS

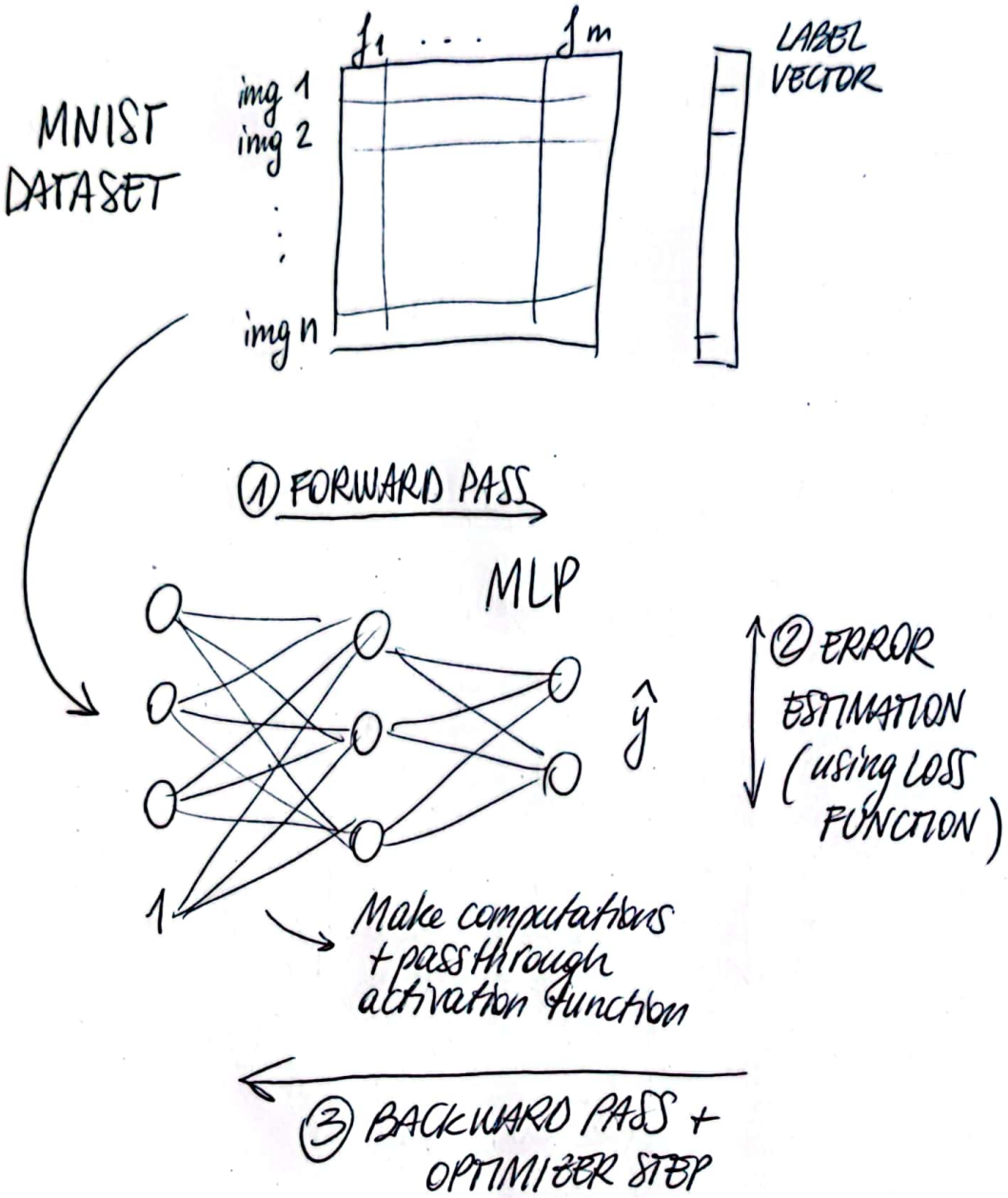
- CROSS-ENTROPY $H(P,Q) = -\sum_{x \in X} p(x) \log q(x)$

- L1 LOSS $l_1(p,q) = \|p-q\|_T = \sum_{i=1}^{n} |p_i - q_i|$

## ▷ OPTIMIZER

- GRADIENT DESCENT
  - STOCHASTIC (SGD)
  - BATCH (BGD)

▷ NOW LET'S PUT EVERYTHING TOGETHER FOR A COMPUTER VISION TASK (e.g. DIGIT RECOGNITION)




Train everything for e epochs.