EMBEDDED VISION DESIGN 3

ARTIFICIAL NEURAL NETWORKS

JEROEN VEEN

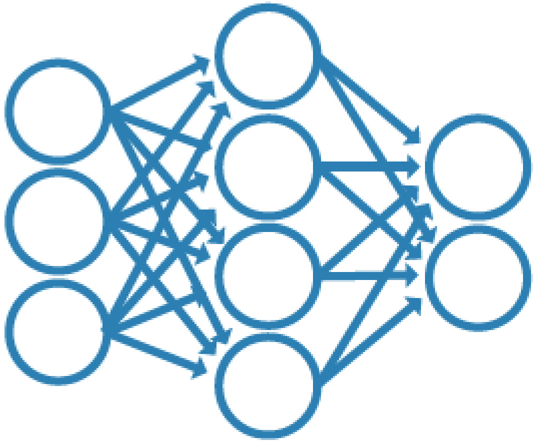
HAN_UNIVERSITY OF APPLIED SCIENCES

DEADLINES

Code	Deadline	Title	LO	Assessment indicators	Acceptance criteria
resit	Monday 22-11	ESEVML-EVD3-P ML portfolio resit	1,2,3		
A3	Monday 29-11	ESEVML-EVD3-P DL portfolio pre-submission, Ch. 1-3	4,5	Introduction and problem statement Data augmentation and preprocessing	DL relation to EVD is discussed. Personal interests and learning objectives in the context of are discussed. SMART problem definition. List or requirements and priotization Data augmented, and method argued. Preprocessing pipeline argued and implemented.
A4	Monday 25-01	ESEVML-EVD3-P full DL portfolio, Ch. 4-6	4, 5, 6	CNN architecture design and training Deploy, test and conclude	Architecture is designed and argued. Data is split into stratified subsets and checked. CNN is trained, cross-validated, and fine-tuned. Performance is evaluated using appropriate methods. Transormation of images is visualized. Net is deployed. Test plan present and test results documented. Results are concluded. Generalization performance discussed. ML and DL application are compared.
resit	Monday 17-01	ESEVML-EVD3-P DL portfolio resit	4, 5, 6		

CONTENTS

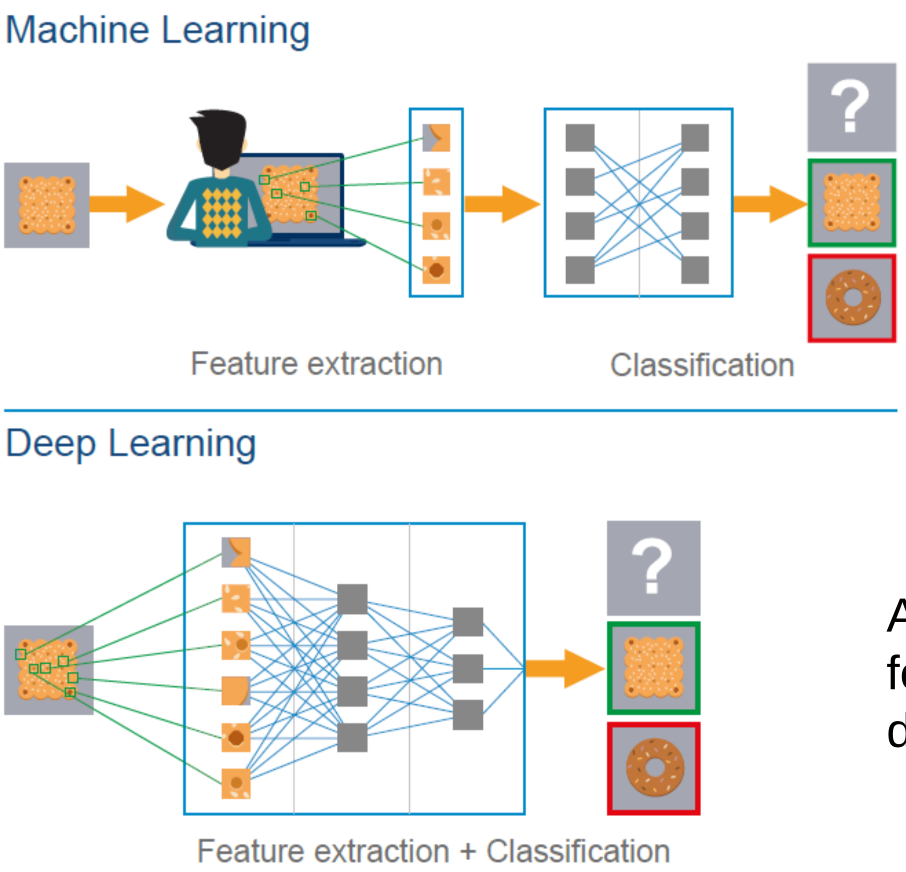
- Machine learning vs deep learning
- Biological neuron
- Perceptron
- Multi-layer perceptron (MLP)
- Backpropagation
- Regression and classification MLP



BACKGROUND MATERIAL

- https://deeplizard.com/learn/playlist/PLZbbT5o_s2xq7LwI2y8_QtvuXZedL6tQU
- https://www.3blue1brown.com/topics/neural-networks

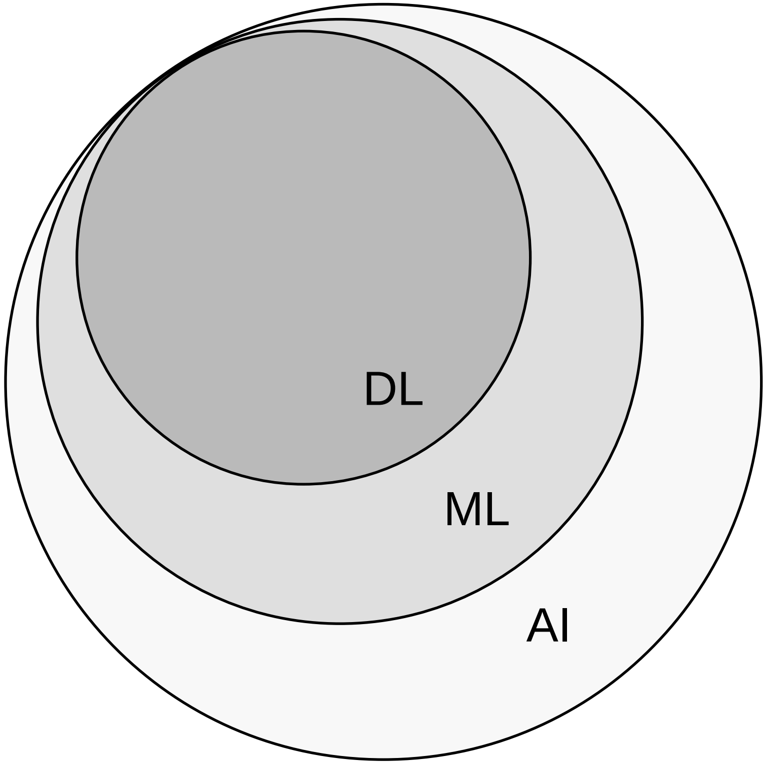
MACHINE LEARNING VS DEEP LEARNING



Autonomous feature definition

Source: Tilmann Zuper, Artificial Intelligence in Image Processing: Deep Learning Compared with Conventional Methods, Basler AG, Ahrensburg, Germany.

DEFINING AI, DL & ML



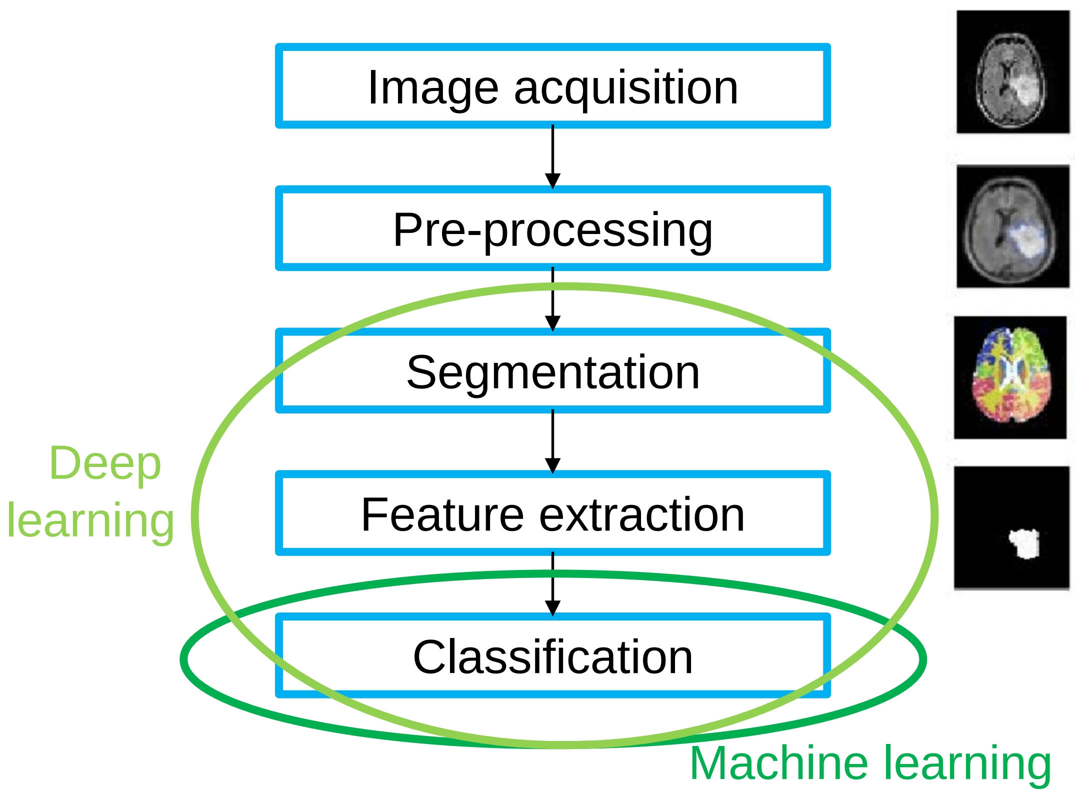
- Strong AI vs Applied AI
- Cognitive replication
- Rational process

Machine learning

- Performs predictive analysis
- Just fancy math & pattern matching

MACHINE LEARNING APPLIED TO VISION

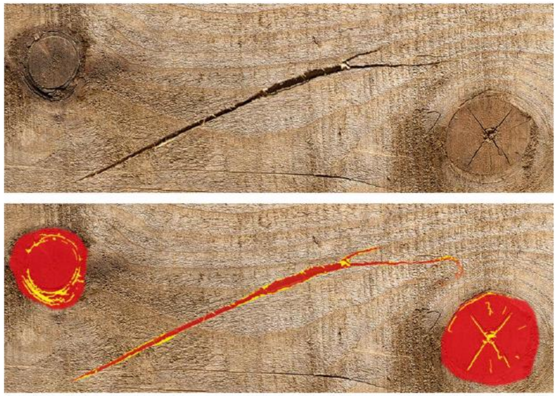
- Classical image processing

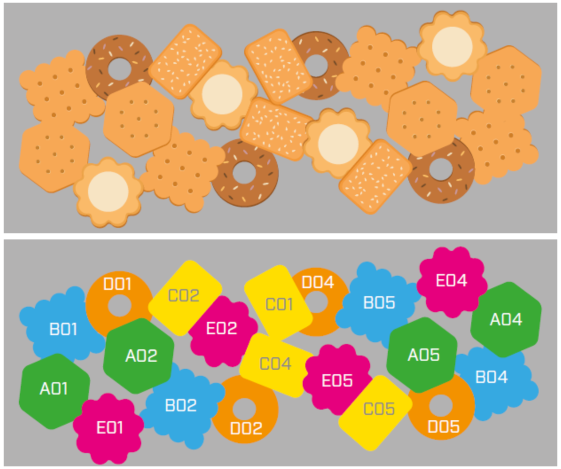


APPLICATION AREAS OF DEEP LEARNING

- Anomaly detection, image classification, image segmentation and object recognition.

- Higher precision and greater flexibility compared to conventional image analysis methods.





Source: Tilmann Zuper, Artificial Intelligence in Image Processing: Deep Learning Compared with Conventional Methods, Basler AG, Ahrensburg, Germany.

HAN_UNIVERSITY OF APPLIED SCIENCES

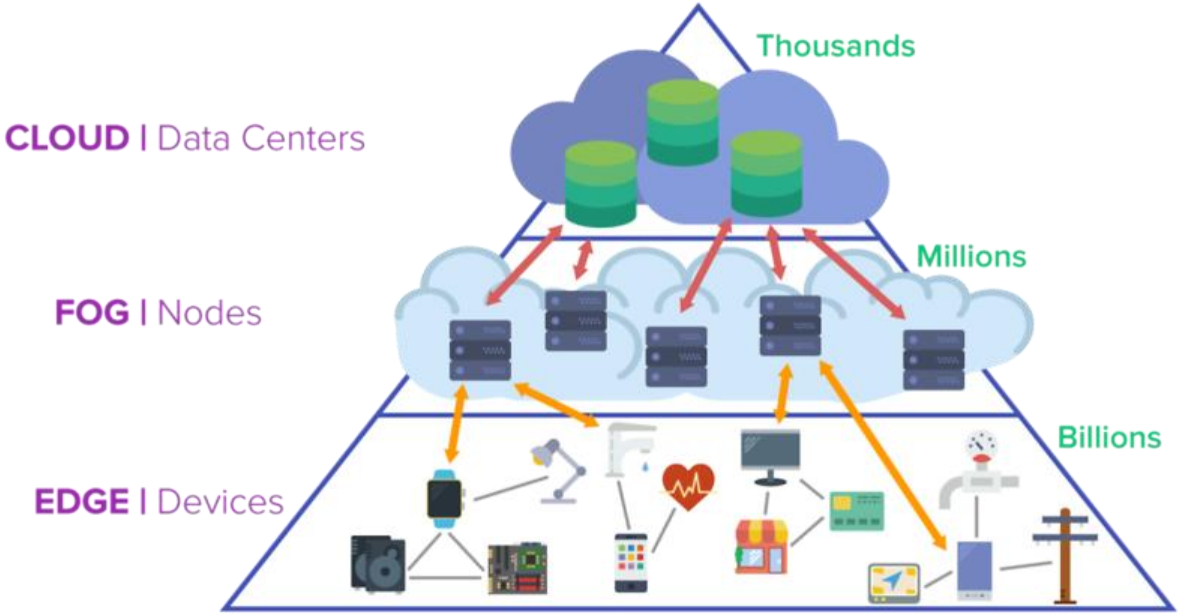
COSTS OF DEEP LEARNING

- Additional hardware
  Large memory and computing capacity is required, typically outsourced to e.g. GPUs (graphic cards).

- Power consumption:
  Large memory and computing capacity increase power consumption and thus the heat generation. This can be problematic for embedded systems.

- High amount of training data:
  Large number of training images required, which is sometimes difficult in the development of a Machine Vision application.

ON THE EDGE



Source: https://medium.com/da-labs/edge-ai-the-future-of-ai-d954ebc40a46

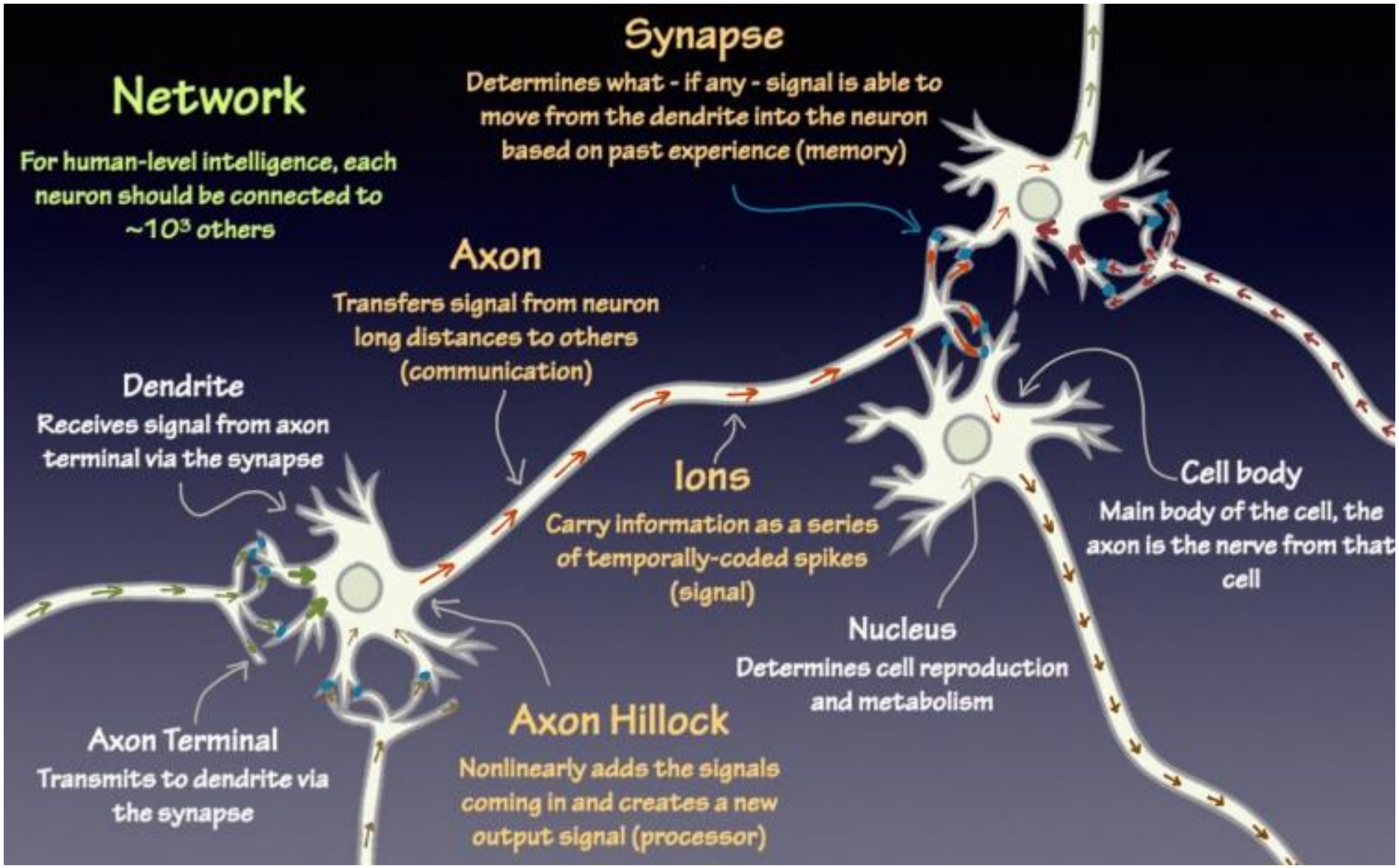
HYBRID APPROACH

- High performance with low memory and power requirement
- Image preprocessing with conventional methods.

- An artificial neural network then delivers the desired results with the preprocessed data.

- DL mingled with expert systems

BIOLOGICAL NEURONS

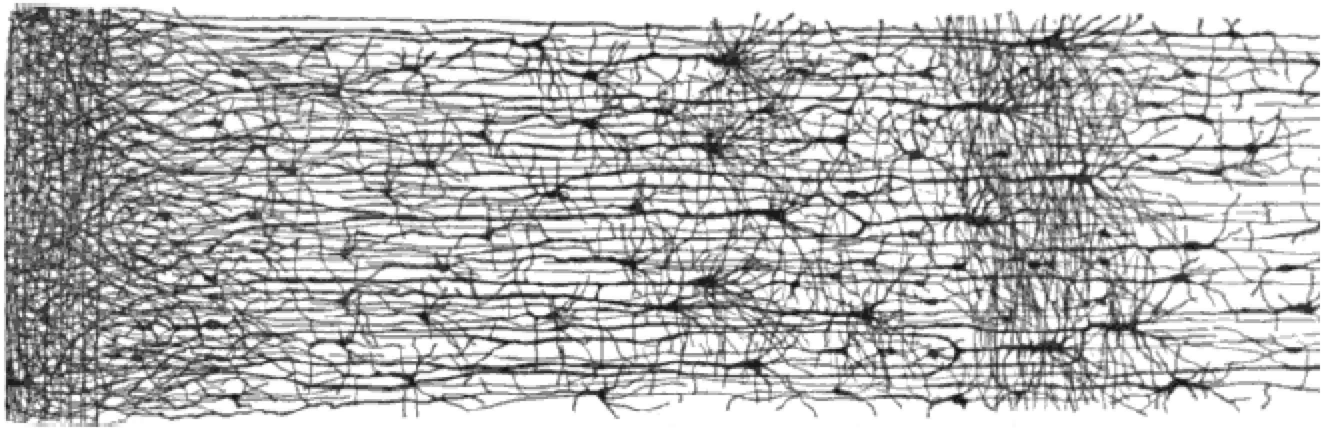


Source: S. Bains, The Promise and Pitfalls of Neuromorphic Computers, EETimes, 2020

NEURAL CIRCUITS

- Population of neurons interconnected by synapses to carry out a specific function when activated

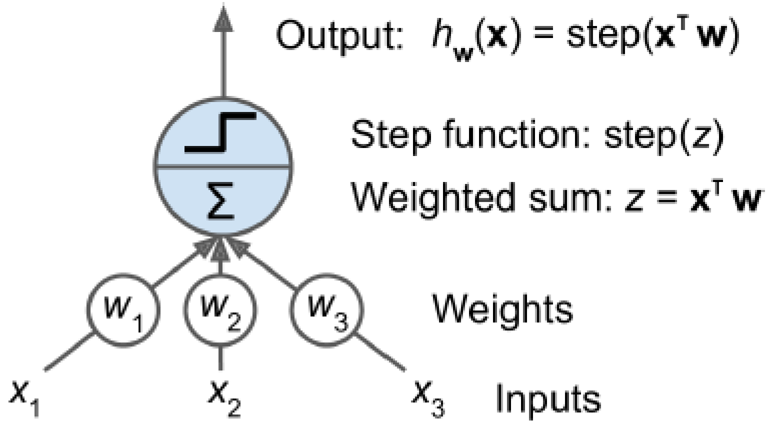
- Highly complex computations can be performed by a network of fairly simple neurons



Source: Géron, ISBN: 9781492032632

THRESHOLD LOGIC UNIT (TLU)

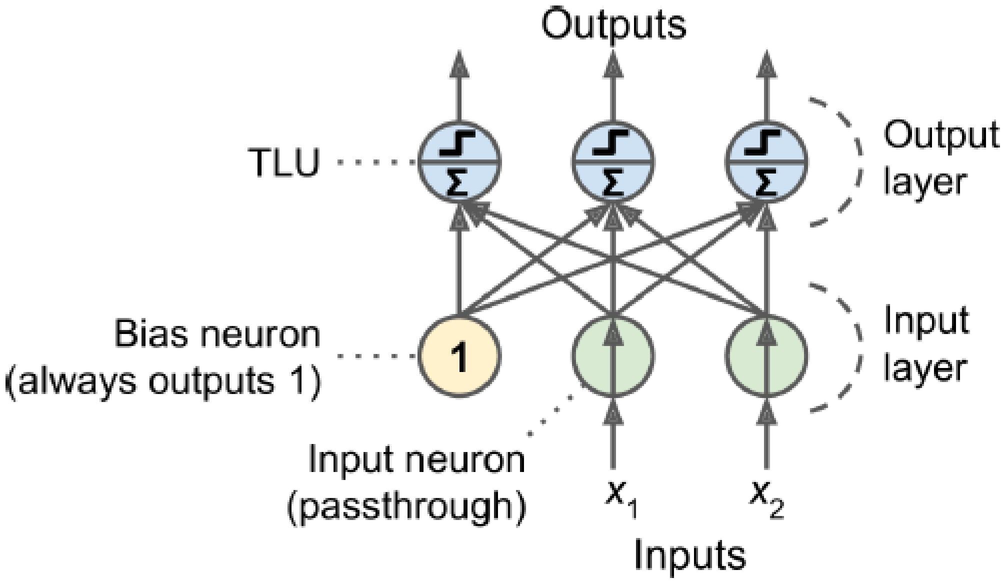
- Elementary unit of an ANN
- Simplified model of a biological neuron
- Dot product followed by a non-linear function
- Performs linear binary classification



Source: Géron, ISBN: 9781492032632

PERCEPTRON

- Single layer of TLUs
- Multioutput classifier
- Connection weights

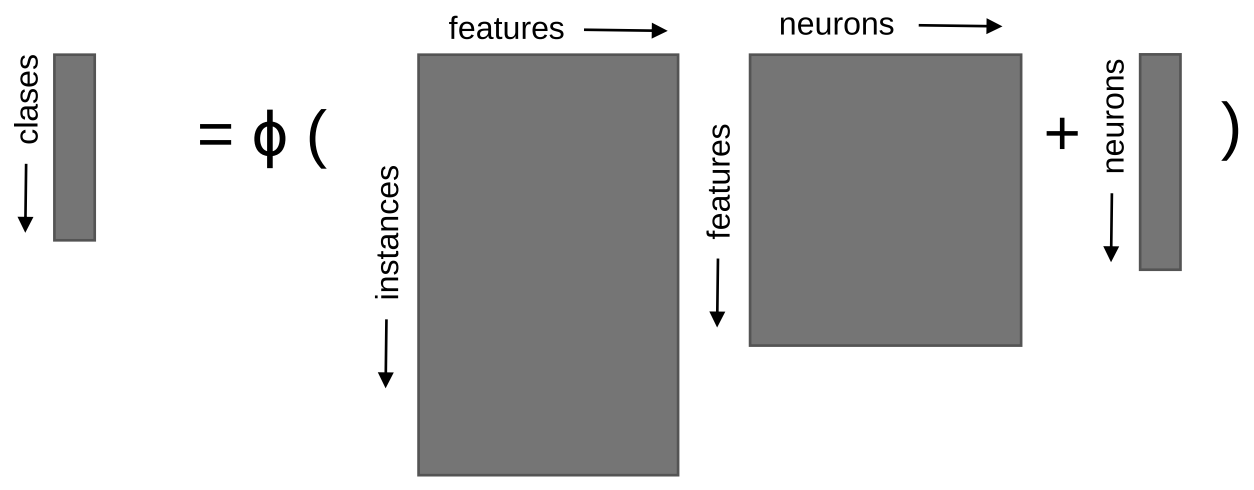


Source: Géron, ISBN: 9781492032632

OUTPUT COMPUTATION

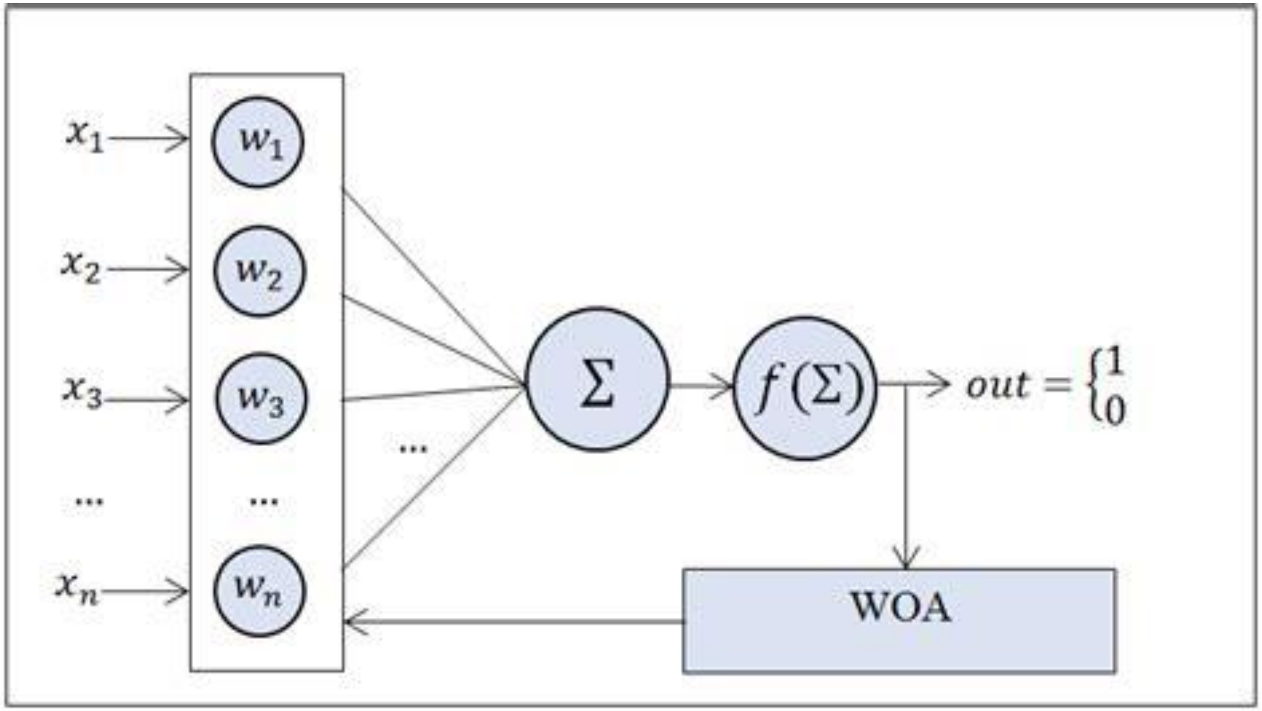
$$h_{\mathbf{W},\,\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{X}\mathbf{W} + \mathbf{b})$$

Output vector   Matrix of input features   Activation function   Weight matrix   Bias vector



HOW TO FIND THE OPTIMAL WEIGHTS?

- Optimization
- Cost function



Source: : Creative Commons Attribution 4.0 International

PERCEPTRON TRAINING ALGORITHM

- Multi-dimensional optimization problem
- Gradient descent

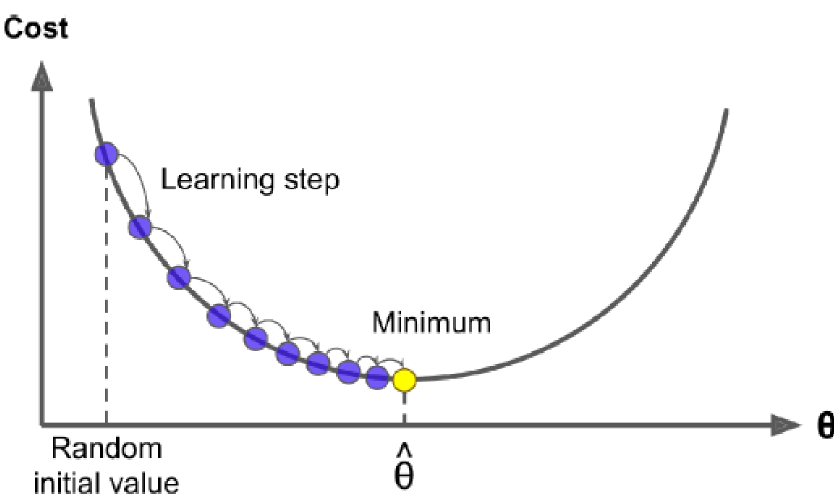
Learning rate

Input value

$$w_{i,j}^{(\text{next step})} = w_{i,j} + \eta \boxed{\left(y_j - \hat{y}_j\right)} x_i$$

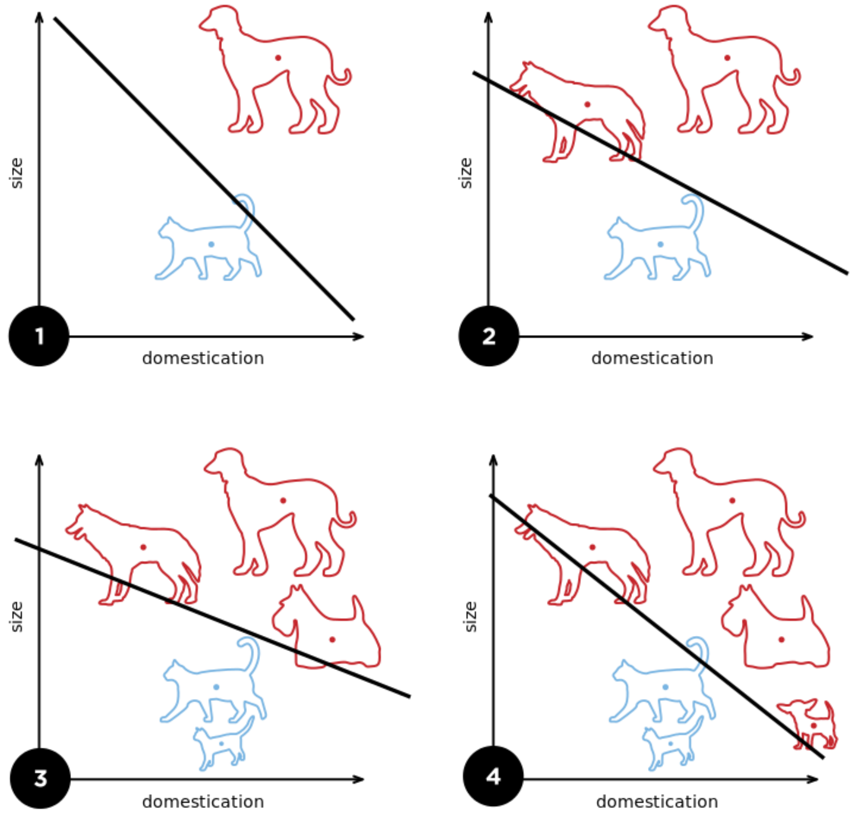
Connection weights

error



HAN_UNIVERSITY
OF APPLIED SCIENCES

EXAMPLE OF ITERATIVE UPDATING



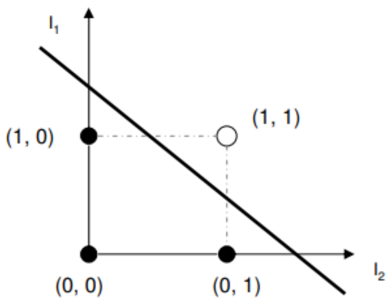
Source: https://en.wikipedia.org/wiki/Perceptron

HAN_UNIVERSITY OF APPLIED SCIENCES

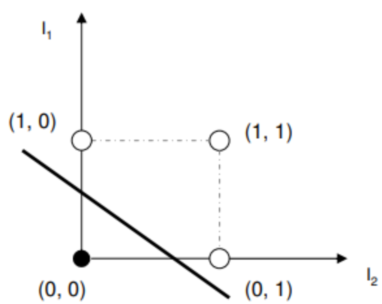
PERCEPTRON LIMITATIONS

- Linear decision boundary
- Incapable of learning complex patterns

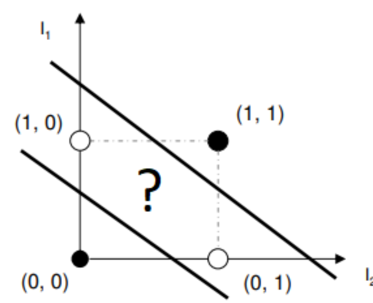
AND		
I_1	I_2	out
0	0	0
0	1	0
1	0	0
1	1	1

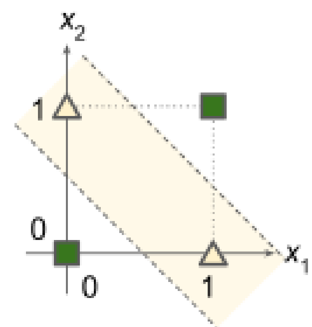


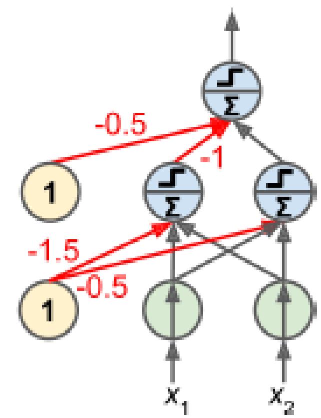
OR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	1



XOR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	0



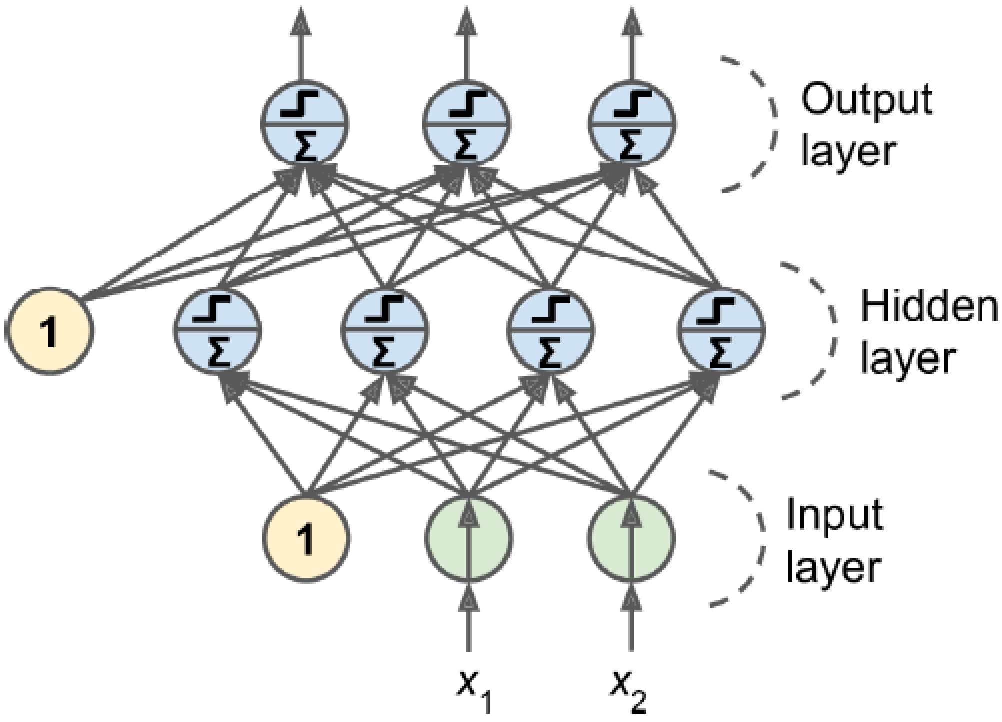




Source: https://mc.ai/solving-xor-with-a-single-perceptron/

MULTILAYER PERCEPTRON

- Feedforward neural network



Source: Géron, ISBN: 9781492032632

BACKPROPAGATION

Let’s now watch

3BLUE1BROWN SERIES S3 • A3
What is backpropagation really doing?
https://www.youtube.com/watch?v=Ilg3gGewQ5U

BACKPROPAGATION

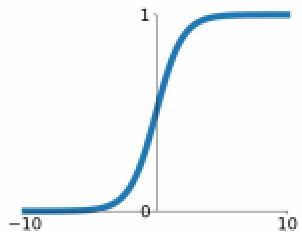
- for each training instance, the backpropagation algorithm first makes a prediction (forward pass) and measures the error,
- then goes through each layer in reverse to measure the error contribution from each connection (reverse pass),
- and finally tweaks the connection weights to reduce the error (Gradient Descent step).

break the symmetry: randomly initialize weights and biases

ACTIVATION FUNCTIONS

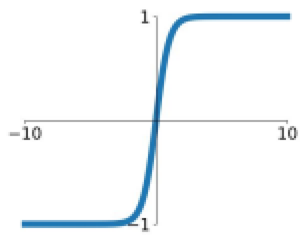
Sigmoid

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



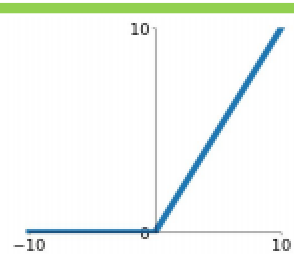
tanh

$\tanh(x)$



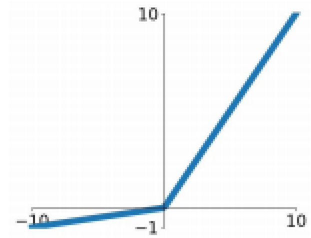
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

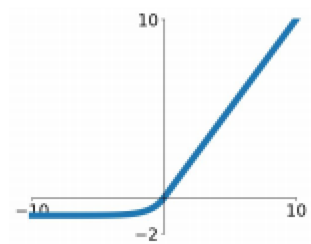


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

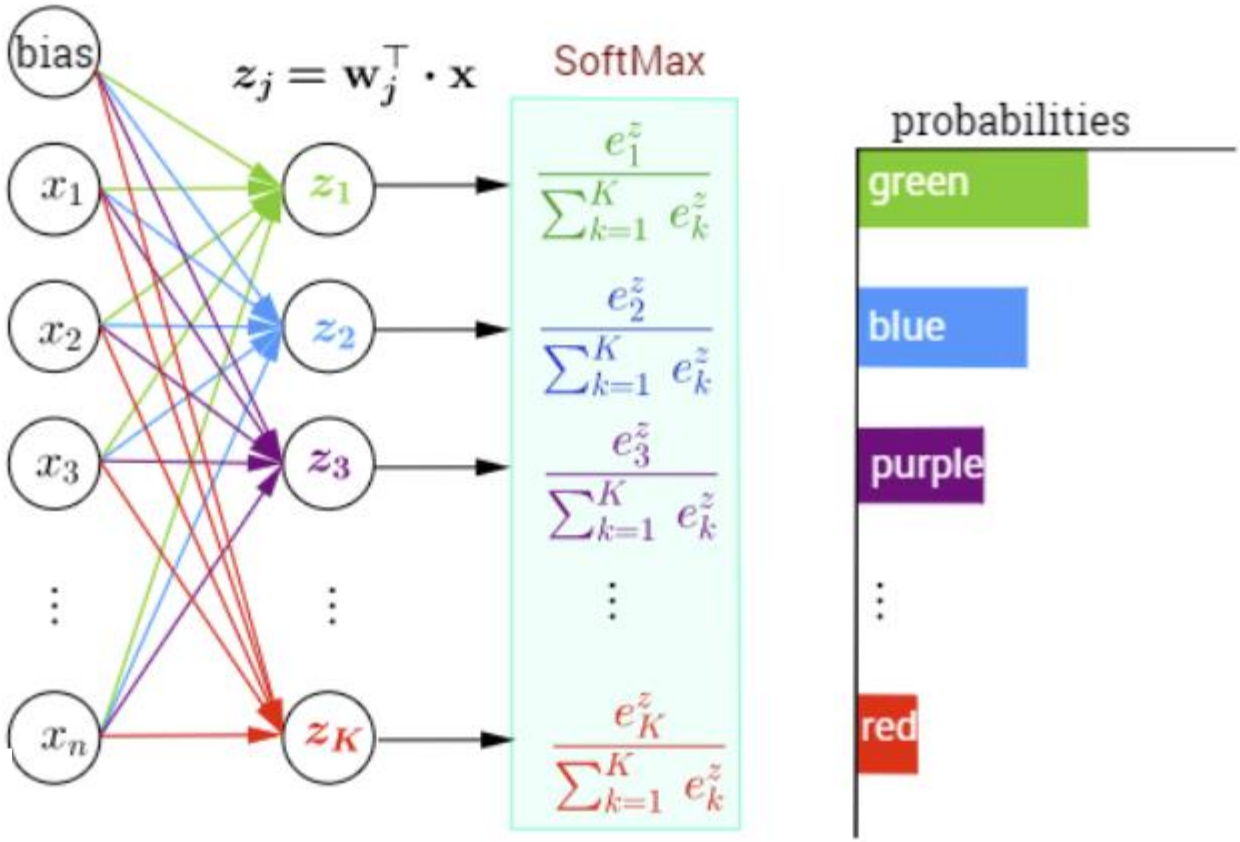
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Source: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture6.pdf

CLASSIFICATION MLP

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_K \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \\ \mathbf{w}_2^\top \\ \mathbf{w}_3^\top \\ \vdots \\ \mathbf{w}_K^\top \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$



$$\sigma(j) = \frac{\exp(\mathbf{w}_j^\top \mathbf{x})}{\sum_{k=1}^{K} \exp(\mathbf{w}_k^\top \mathbf{x})} = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)}$$

Source: http://rinterested.github.io/statistics/softmax.html

REGRESSION MLP

- No activation function for output neurons required
- Use functions to bound outputs, e.g. relu, softplus, logistic function

Table 10-1 summarizes the typical architecture of a regression MLP.

Table 10-1. Typical regression MLP architecture

Hyperparameter	Typical value
# input neurons	One per input feature (e.g., 28 x 28 = 784 for MNIST)
# hidden layers	Depends on the problem, but typically 1 to 5
# neurons per hidden layer	Depends on the problem, but typically 10 to 100
# output neurons	1 per prediction dimension
Hidden activation	ReLU (or SELU, see Chapter 11)
Output activation	None, or ReLU/softplus (if positive outputs) or logistic/tanh (if bounded outputs)
Loss function	MSE or MAE/Huber (if outliers)

Source: Géron, ISBN: 9781492032632

EXERCISE

- https://developers.google.com/machine-learning/crash-course/reducing-loss/playground-exercise

- How did the lower learning rate impact convergence?
- Can you find a learning rate too slow to be useful?
- Better website: https://playground.tensorflow.org

NEXT TIME

- Hands-on: Tensorflow & Keras 2
  - Tensorboard
  - Fine-tuning neural network hyperparameters
  - Exercise: tune hyperparameters
    (Make sure you have completed this week's exercise!)

- Theory: Training deep neural networks
  - Vanishing and exploding gradients
  - Transfer learning
  - Learning rate scheduling
  - Regularization