Advanced Machine Learning
Lecture 5: Neural networks

Sandjai Bhulai
Vrije Universiteit Amsterdam

s.bhulai@vu.nl
19 September 2023

VU VRIJE UNIVERSITEIT AMSTERDAM
Faculty of Science

Neural networks

Advanced Machine Learning

Neural networks

- Linear models studies so far suffer from the curse of dimensionality
- Solution 1: Define basis functions centered on training data points and then select (SVM)
- Solution 2: Fix the basis functions in advance but allow them to be adaptive (NN)

Neural networks

- So far, we have seen

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \varphi_j(x)\right)$$

- Goal: series of functional transformations
- Construct M linear combinations of $x_1, \ldots, x_D$

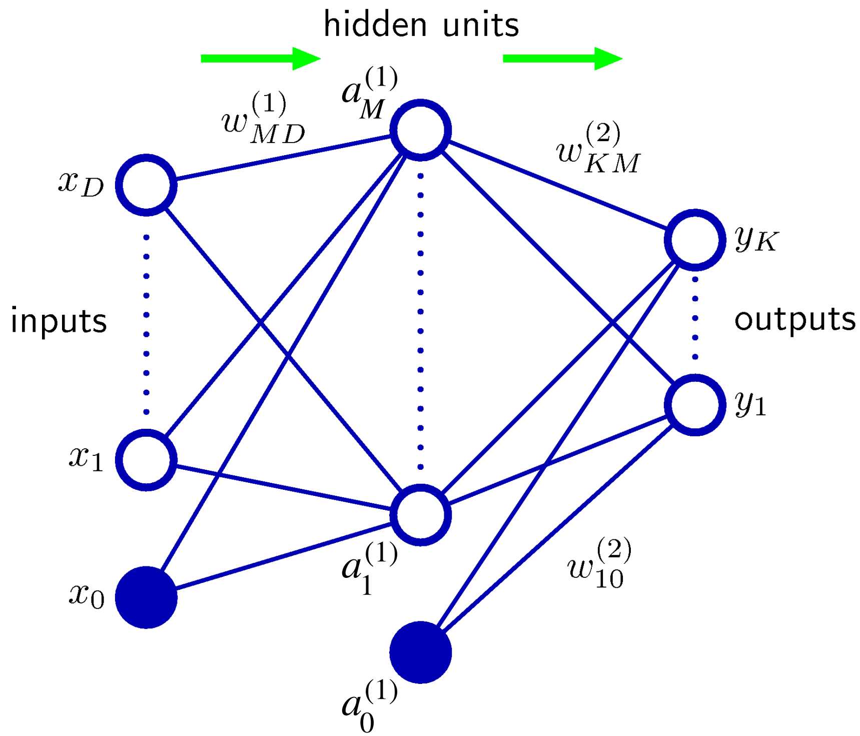
$$z_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$$

- Transform linear combination z_j by $a_j = h(z_j)$

Neural networks

- Construct new K linear combinations

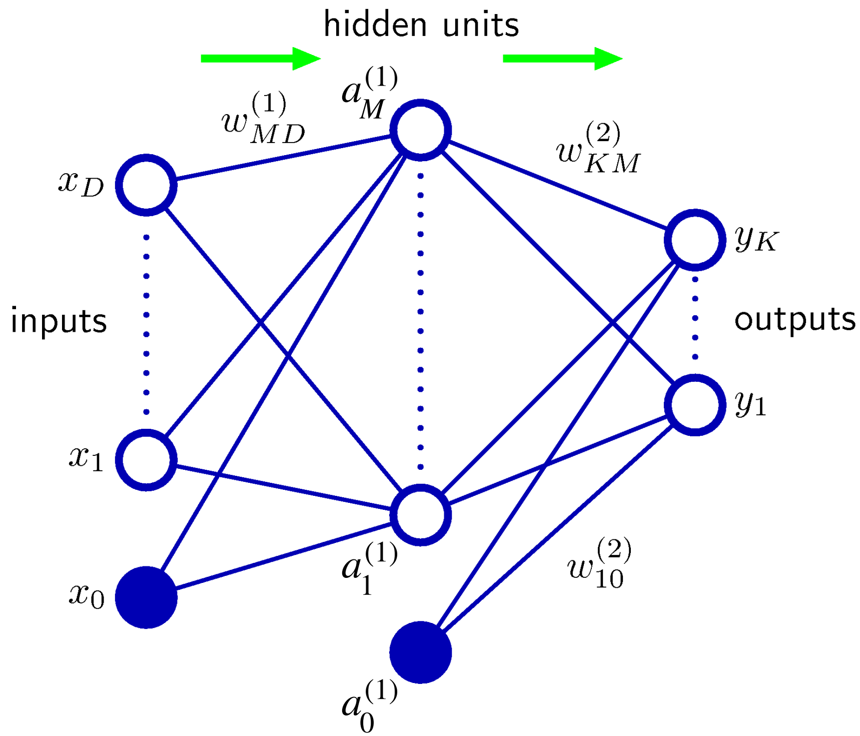
$$z_k = \sum_{j=1}^{M} w_{kj}^{(2)} a_j + w_{k0}^{(2)}$$



Neural networks

- Transform to outputs by activation function σ

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$



Neural networks

- How to choose σ?
  - σ is identity function for regression
  - σ is sigmoid function for binary classification
  - σ is softmax function for multi class problems

- How to choose h?
  - Linear function gives a linear model related to PCA
  - Sigmoid, tanh
  - ReLu, leaky ReLU, …

Neural networks

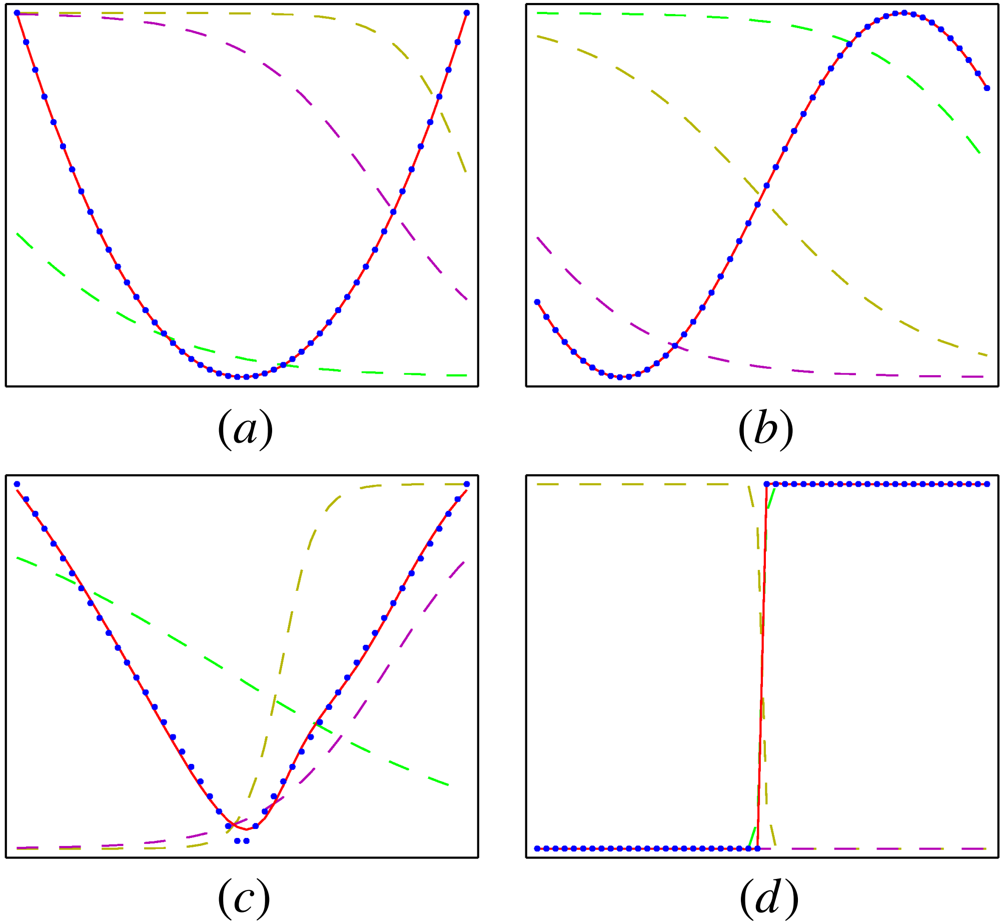
- Neural networks are universal approximations

  a) $f(x) = x^2$

  b) $f(x) = \sin(x)$

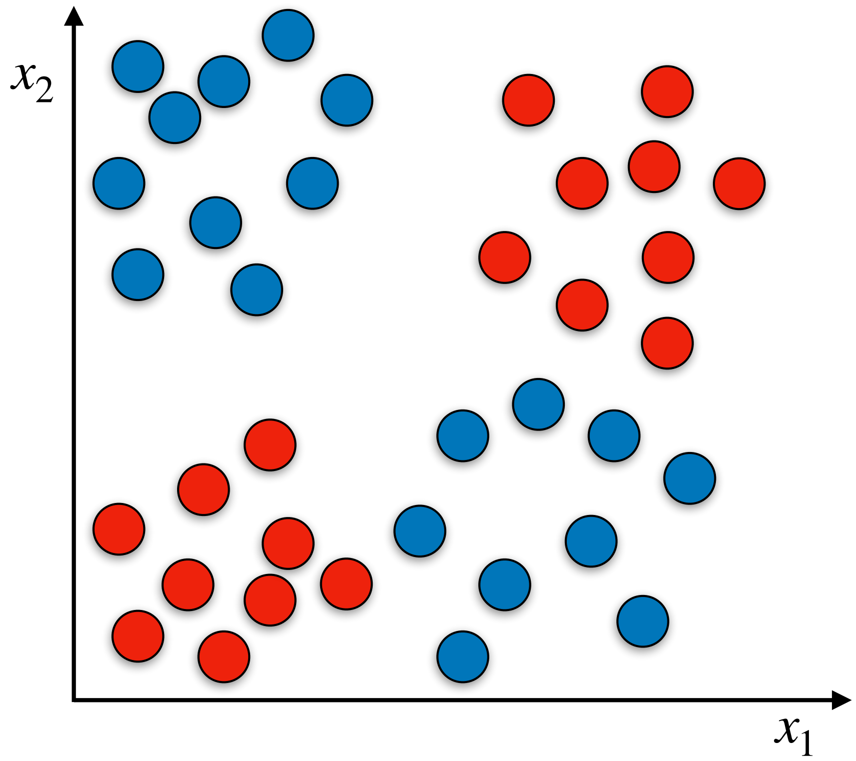
  c) $f(x) = |x|$

  d) $f(x) = H(x)$



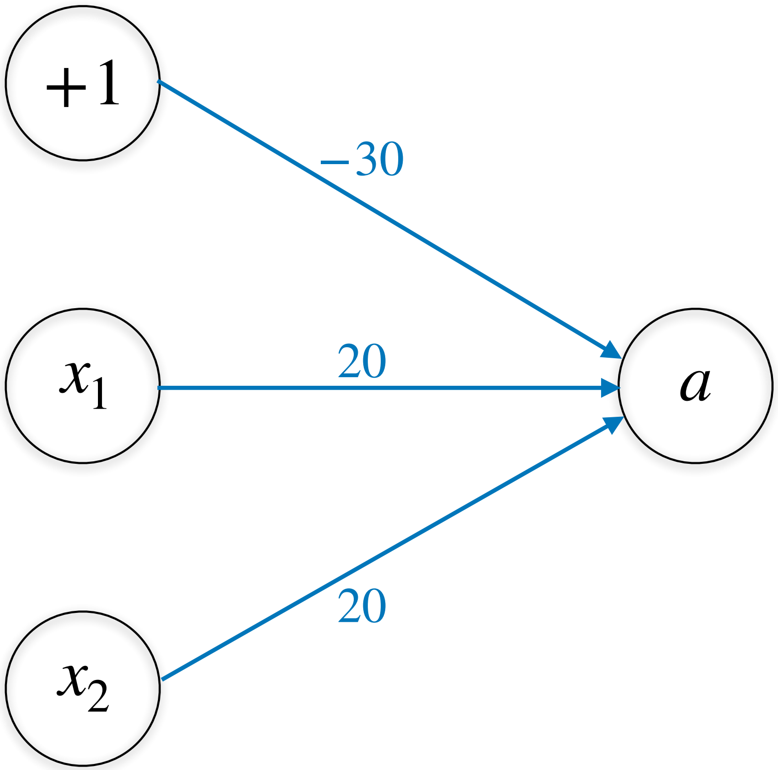
- $N = 50$
- 2-layer network
- 3 hidden tanh units

Neural networks - intuition



Neural networks - intuition

$$g(-30 + 20x_1 + 20x_2)$$

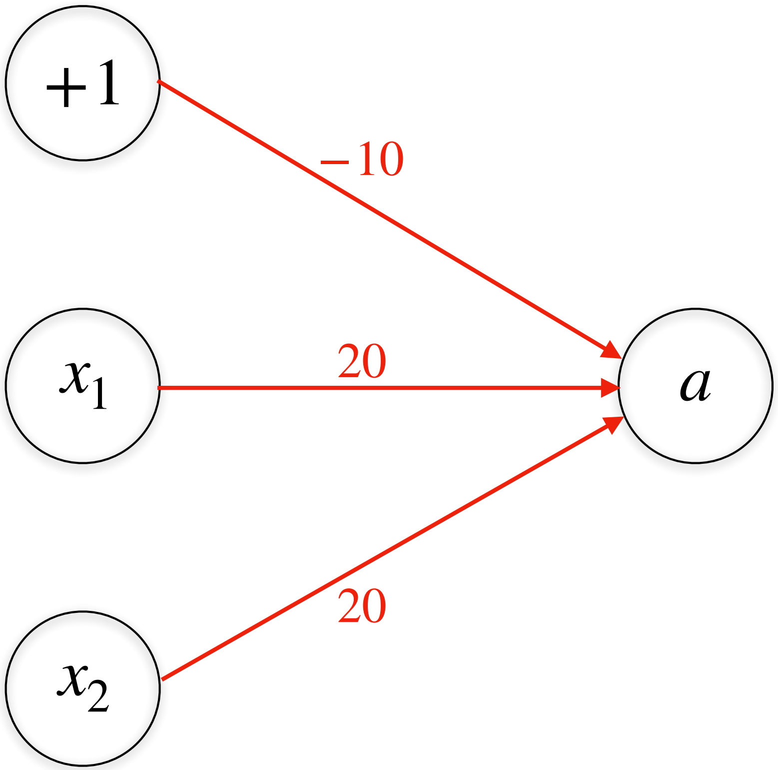


Model: AND gate

x_1	x_2	a	$g(a)$
0	0	-30	0
0	1	-10	0
1	0	-10	0
1	1	10	1

Neural networks - intuition

$$g(-10 + 20x_1 + 20x_2)$$

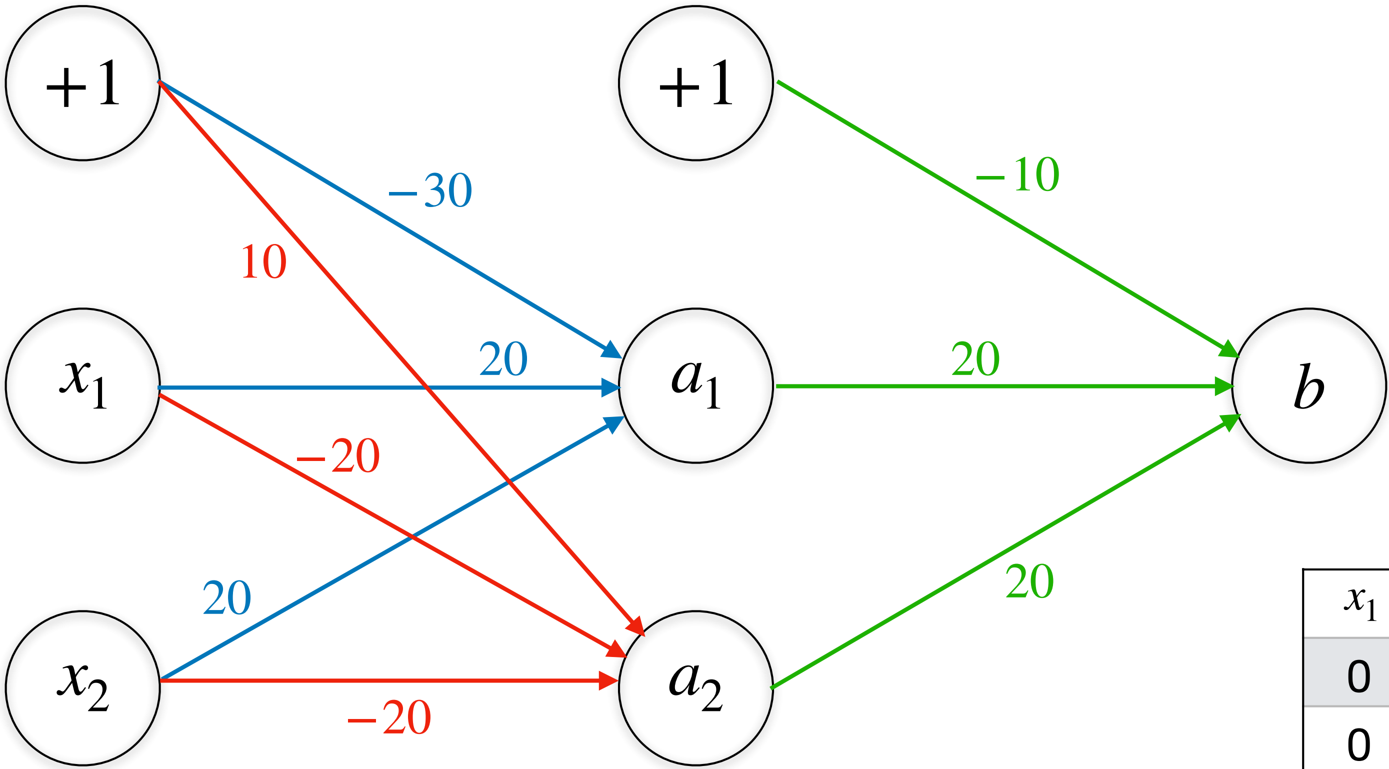


Model: OR gate

x_1	x_2	a	$g(a)$
0	0	-10	0
0	1	10	1
1	0	10	1
1	1	30	1

Neural networks - intuition

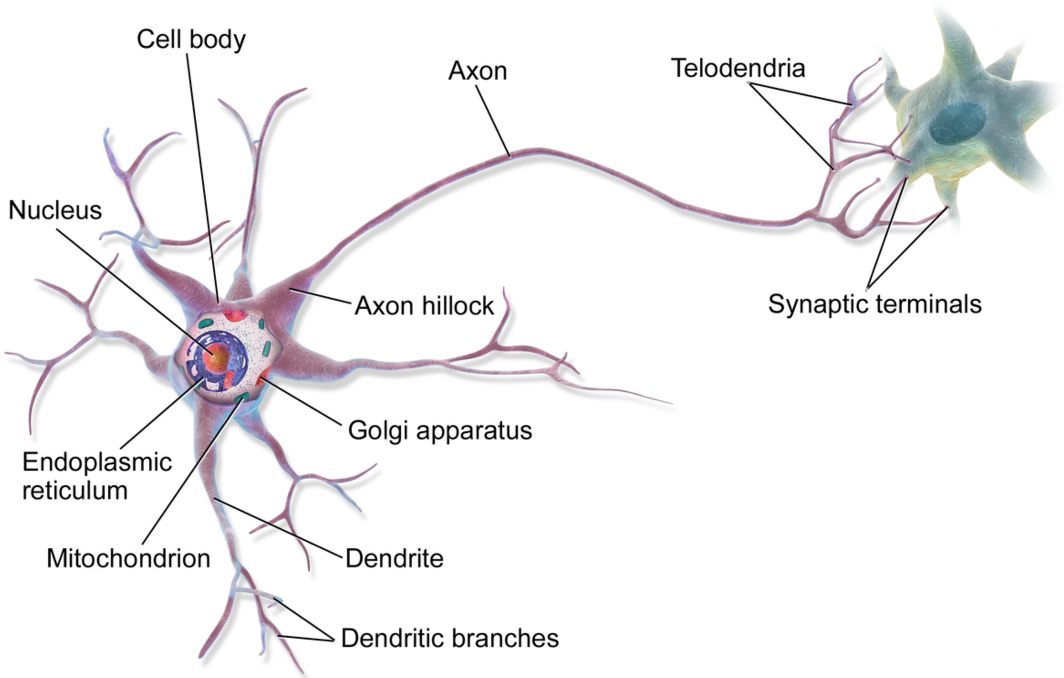
$$g\Big(-10 + 20g(-30 + 20x_1 + 20x_2) + 20g(10 - 20x_1 - 20x_2)\Big)$$



x_1	x_2	$g(a_1)$	$g(a_2)$	$g(b)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

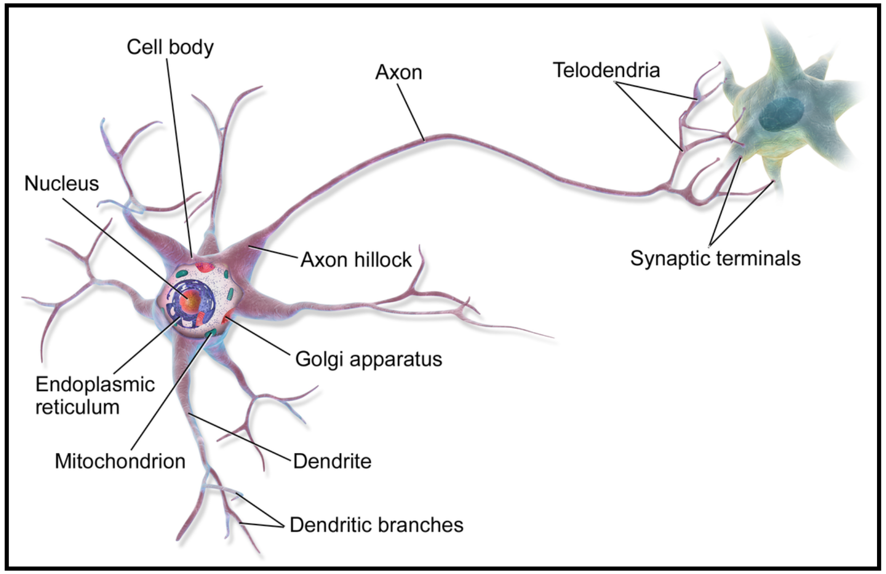
Neural networks - intuition

- The human brain:
  - 10 billion neurons
  - 60 trillion connections



Neural networks - intuition

- How does a neuron work?
  - It receives input signals through its dendrites
  - The input signals generate difference of electrical potential on the cell membrane
  - The difference is propagated to the axon hillock
  - A train of electrical impulses is generated along the axon
  - The impulses on the axon generate the release in the synaptic space of some neurotransmitters



Neural networks - history

The roots of neural networks are in:

- **Neurobiological studies (more than one century ago)**:
  - How do nerves behave when stimulated by different magnitudes of electric current? Is there a minimal threshold needed for nerves to be activated? Given that no single nerve cell is long enough, how do different nerve cells communicate among each other?

- **Psychological studies**:
  - How do animals learn, forget, recognize and perform other types of tasks?

- **Psycho-physical experiments**:
  - Understand how individual neurons and groups of neurons work

Neural networks - history

- **Pitts and McCulloch (1943)**:
  - First mathematical model of biological neurons
  - All boolean operations can be implemented by these neuron-like nodes (with different threshold and excitatory/inhibitory connections)
  - Origin of automaton theory

- **Hebb (1949)**:
  - Hebbian rule of learning: increase the connection strength between neurons and whenever both and are activated
  - Or increase the connection strength between nodes and whenever both nodes are simultaneously ON or OFF

Neural networks - history

Early booming (50's – early 60's)

- **Rosenblatt (1958)**:
  - Perceptron: network of threshold nodes for pattern classification. Perceptron learning rule – first learning algorithm
  - Perceptron convergence theorem:
    - Everything that can be represented by a perceptron can be learned
- **Widrow and Hoff (1960, 1962)**:
  - Learning rule that is based on minimization methods
  - Minsky's attempt to build a general purpose machine with Pitts and McCulloch units

Neural networks - history

The setback (mid 60’s – late 70’s)

- **Minsky and Papert** publish a book “Perceptrons” (1969):
  - Single layer perceptrons cannot represent (learn) simple functions such as XOR
  - Multi-layer of non-linear units may have greater power but there was no learning rule for such networks
  - Scaling problem: connection weights may grow infinitely
- US defense / government stops funding research on artificial neural networks (ANNs)

Neural networks - history

Renewed enthusiasm and flourish (80's – 90's)

- **New techniques**:
  - Backpropagation learning for multi-layer feed forward nets (with non-linear, differentiable node functions)
  - Physics inspired models (Hopfield net, Boltzmann machine, etc.)
  - Unsupervised learning

- Impressive applications (character recognition, speech recognition, text-to-speech transformation, process control, associative memory, etc.)

But:

- Criticism from statisticians, neurologists, biologists, ordinary users, …
- Lots of ad-hoc solutions, "wild creativity"
- A lot of rubbish is produced …

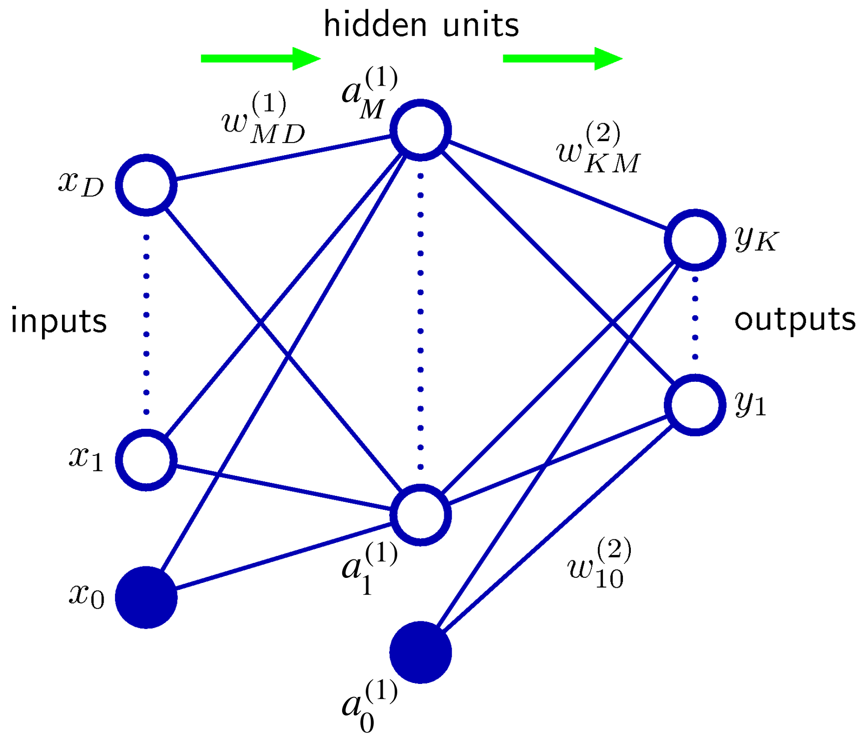
Neural networks - history

- **Revolution**: next to neural networks new techniques offer increasingly promising results (90’s – now)
  - Support Vector Machines (Vapnik)
  - Kernel methods (Vapnik, Scholkopf, …)
  - Ensemble methods (Breiman, Hasti, Tibshirani, Friedman, …)
  - Bagging
  - Boosting
  - Stacking
  - Deep Learning
  - Transparency (LIME, SHAP, …)

Neural networks

- Transform to outputs by activation function σ

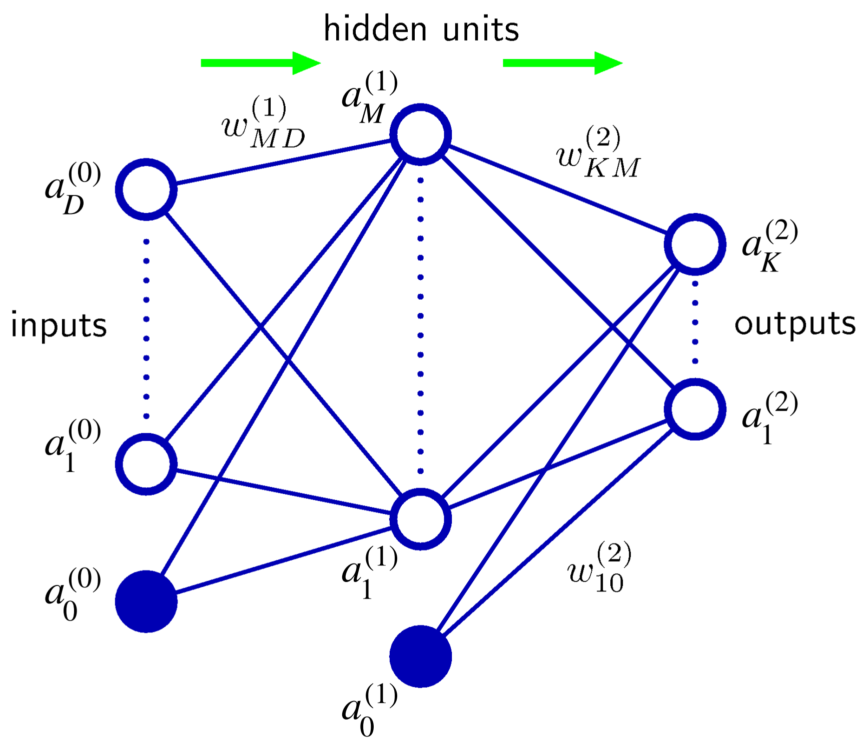
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$



Neural networks

- Transform to outputs by activation function σ

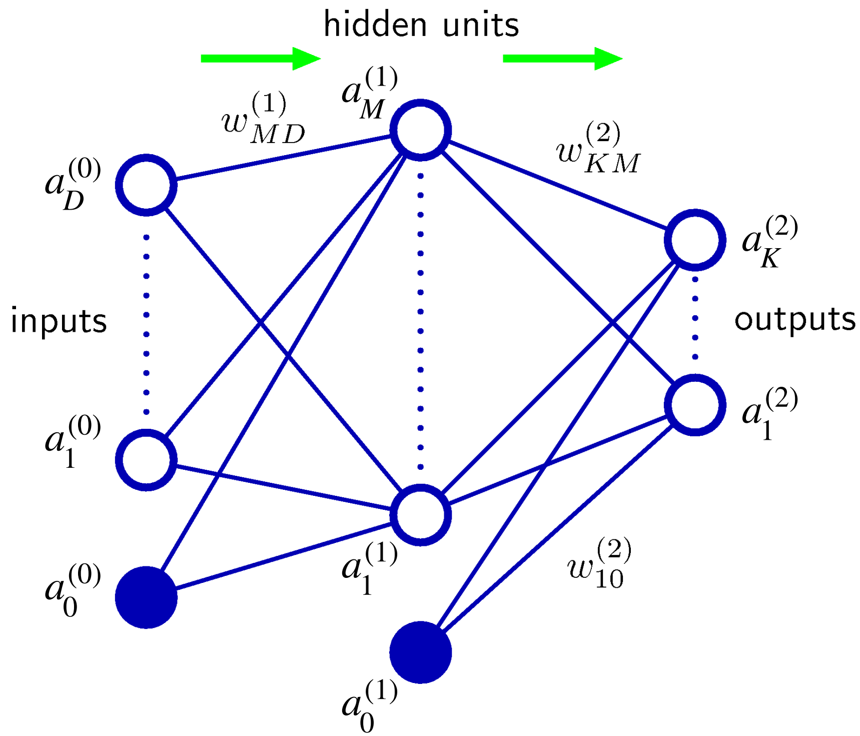
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$



Neural networks

- Transform to outputs by activation function σ

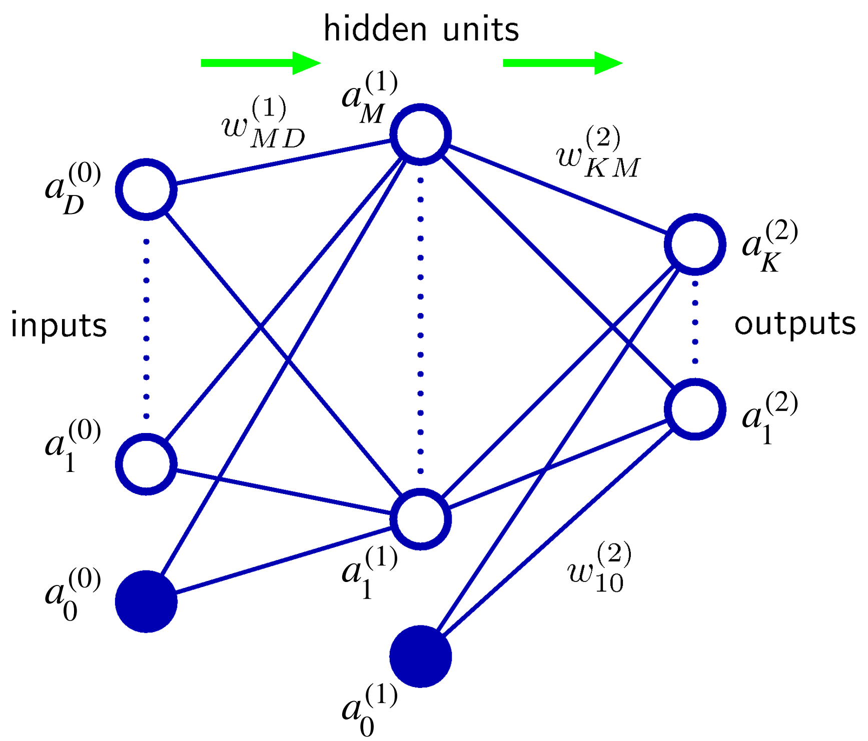
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} a_i^{(0)} + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right)$$



Neural networks

- Transform to outputs by activation function σ

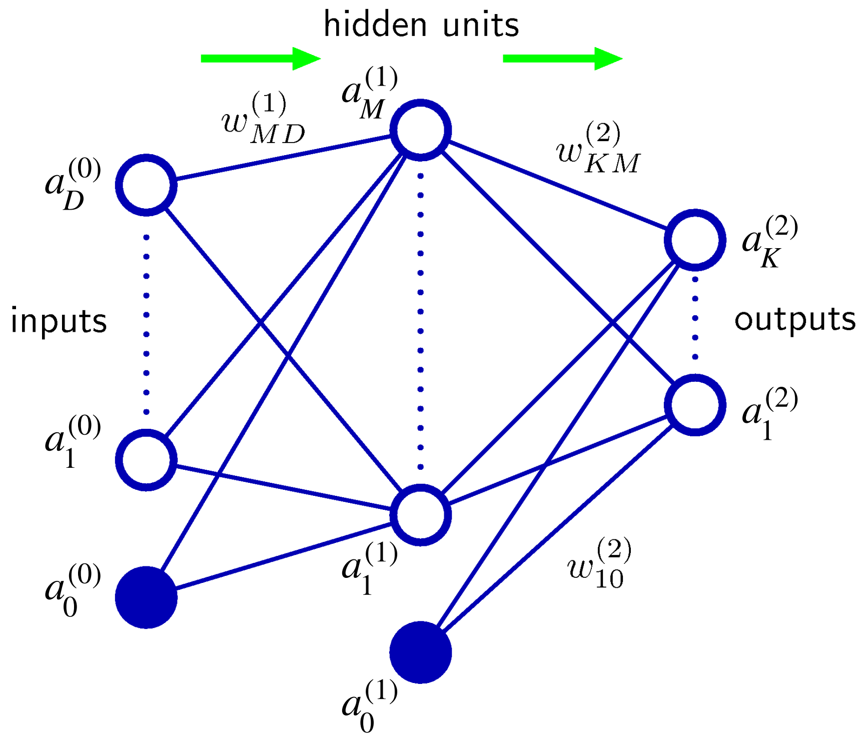
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h(z_j^{(1)}) + w_{k0}^{(2)}\right)$$



Neural networks

- Transform to outputs by activation function σ

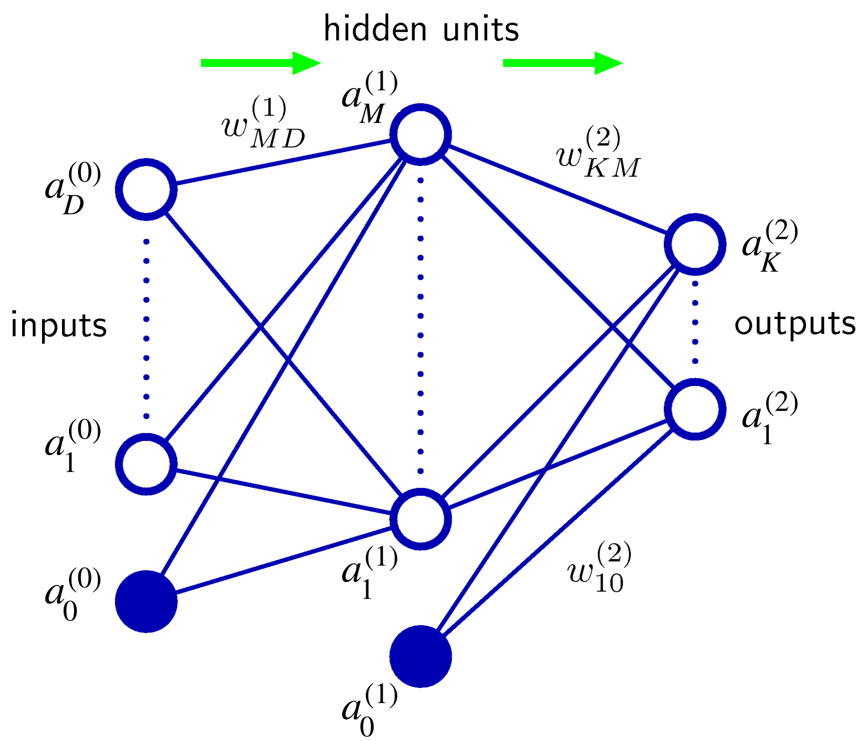
$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} a_j^{(1)} + w_{k0}^{(2)}\right)$$



Neural networks

- Transform to outputs by activation function σ

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma(z_K^{(2)})$$



Neural networks

- Transform to outputs by activation function σ

$$y_k(\mathbf{x}, \mathbf{w}) = a_k^{(2)}$$