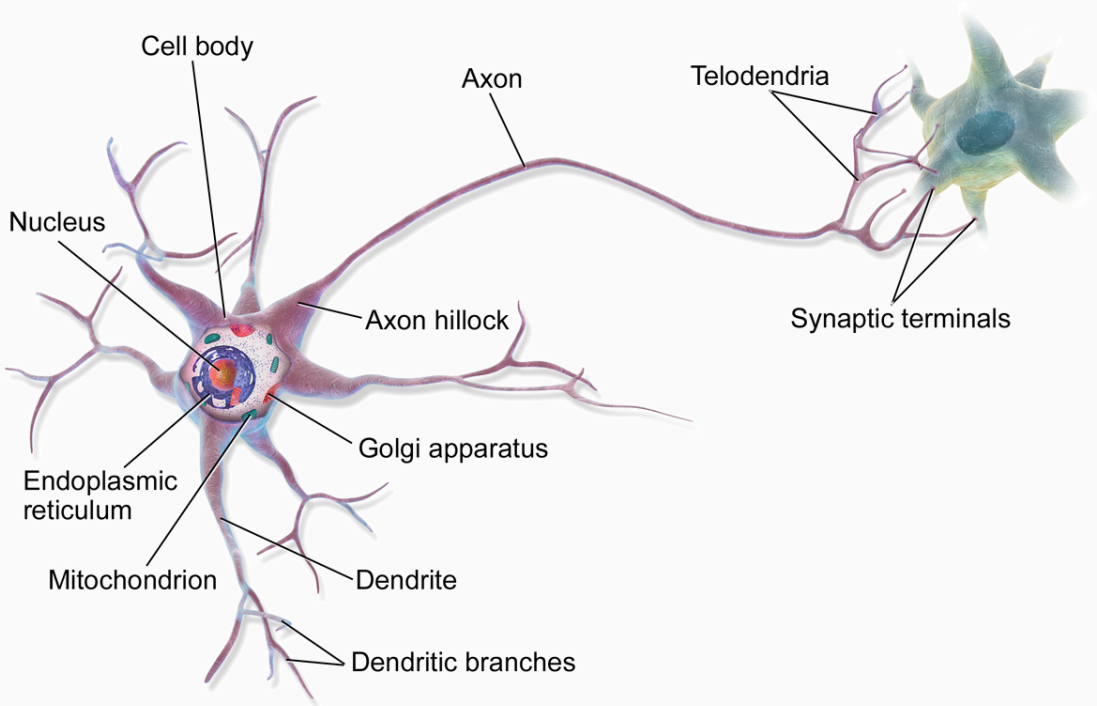
Backpropagation

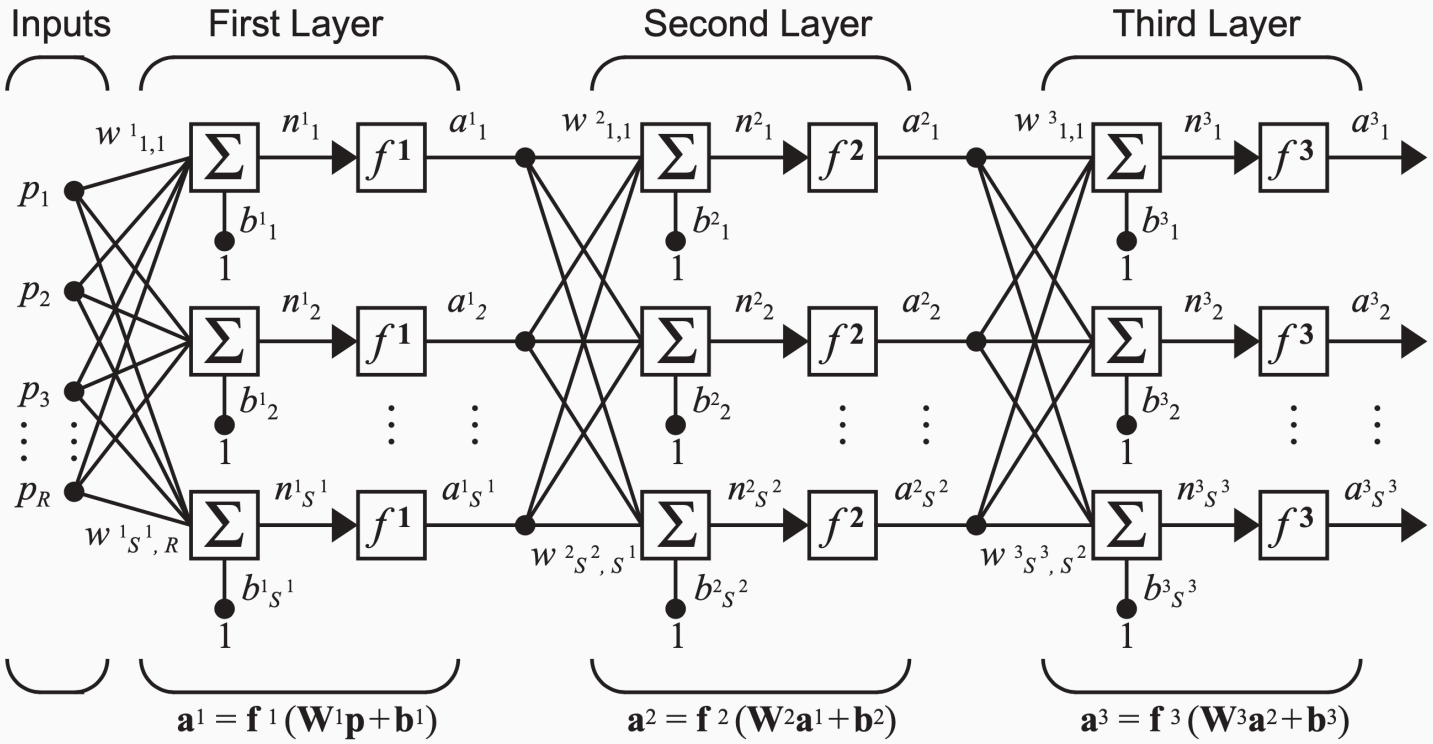
Neuron



- absztrakt neuron modell: $y = f\left(\sum_i (x_i w_i + b)\right)$
- x_i: a neuron bemenetei
- w_i: súlyzók
- b: bias
- f: aktivációs függvény
- y (vagy a): a neuron kimenete

Többrétegű előrecsatolt neuronháló

- háromrétegű előrecsatolt neuronháló:



$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

- nemlináris aktivációs függvénnyel univerzális approximátor
- a bias-t a továbbiakban implicit módon kódoltnak vesszük

Backpropagation

A backpropagation

- gradiens módszer a többrétegű előrecsatolt neuronhálók felügyelt tanítására
- a következőkben az *online* verziót tárgyaljuk

Jelölések

- $f()$ aktivációs függvény
- i, j két egymás után következő rétegben levő neuron
- w_{ij} súlyzó
- $v_i = \sum_j y_j w_{ij}$ az i neuron bemenete
- $y_i = f(v_i)$ az i neuron kimenete
- E a neurális háló hibája *egy* tanítási példára
- $G = \sum E$ a neurális háló hibája a teljes tanítási halmazra

(Általános) Delta szabály: $\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$

Lokális gradiens

- Definíció. $\delta_i \equiv \dfrac{\partial E}{\partial v_i}$ az i neuron *lokális gradiense*.
- $\dfrac{\partial E}{\partial w_{ij}} = \dfrac{\partial E}{\partial v_i}\dfrac{\partial v_i}{\partial w_{ij}} = \dfrac{\partial E}{\partial v_i}\dfrac{\partial \left(\sum_j y_j w_{ij}\right)}{\partial w_{ij}} = \dfrac{\partial E}{\partial v_i} y_j$

Lokális gradiens (2)

- ha i egy kimeneti neuron
  - $e_i = 0.5(y_i - d_i)^2$ (négyzetes hibafüggvény!)
  - $E = \sum_i e_i$
  - $\delta_i = \frac{\partial E}{\partial v_i} = \frac{\partial (\sum_i e_i)}{\partial v_i} = \frac{\partial e_i}{\partial v_i} = \frac{\partial 0.5(y_i - d_i)^2}{\partial y_i} \frac{\partial y_i}{\partial v_i} = (y_i - d_i) f'(v_i)$

- ha i az utolsó előtti rétegben levő rejtett neuron
  - $E = \sum_k e_k$
  - $e_k = 0.5(y_k - d_k)^2$
  - $\delta_i = \frac{\partial E}{\partial v_i} = \sum_k \frac{\partial e_k}{\partial v_i}$
  - $\frac{\partial e_k}{\partial v_i} = \frac{\partial \left(0.5(y_k - d_k)^2\right)}{\partial y_k} \frac{\partial f(v_k)}{\partial v_k} \frac{\partial (\sum_i y_i w_{ki})}{\partial y_i} \frac{\partial f(v_i)}{\partial v_i} = (y_k - d_k) f'(v_k) w_{ki} f'(v_i)$
  - $\delta_i = \frac{\partial E}{\partial v_i} = \sum_k \delta_k w_{ki} f'(v_i)$

- általánosan: az n-ik rétegben levő rejtett neuronok lokális gradiensei kifejezhetőek az $n+1$. réteg lokális gradiensei segítségével:
  $\delta_i = \frac{\partial E}{\partial v_i} = \sum_k \delta_k w_{ki} f'(v_i)$,
  ahol i az n. rétegben, k az $n+1$. rétegben levő neuron

Backpropagation algoritmus (online verzió)

1. (Véletlenszerű értékekkel) inicializáljuk a háló w_{ij} súlyzóit
2. Nullázzuk a hálozat *globális* hibáját: $G = 0$
3. Kiválasztunk egy $\mathbf{x}$ elemet a tanítasi halmazból
4. Forward propagation. Kiszámítjuk a hálózat y_k kimeneteit $\mathbf{x}$-re
5. Kiszámítjuk a kimeneti rétegen az e_k hibákat
6. Kiszámítjuk a hálózat négyzetes hibáját $E = \sum_k e_k$
7. Aktualizáljuk a hálózat globális négyzetes hibáját: $G = G + E$
8. Backpropagation. A rétegeken *hátulról előre haladva* minden i neuronra kiszámítjuk a lokális gradienst:

$$\delta_i = \begin{cases} (y_i - d_i) f'(v_i), \text{ ha } i \text{ egy output neuron} \\ \sum_k \delta_k w_{ki} f'(v_i), \text{ ha } i \text{ egy rejtett neuron} \end{cases}$$

9. Aktualizáljuk a súlyzókat:

$$\Delta w_{ij} = -\eta \delta_i y_j$$

10. Ha van még további elem a tanítási halmazban, akkor ugrás 3-ra
11. Ha G nem esett egy adott küszöb alá ($G > \varepsilon$), akkor ugrás 2-re

Könyvészet, ábrák

- Christopher Bishop: Pattern Recognition and Machine Learning, 2006
- Simon Haykin: Neural Networks and Learning Machines, 2009
- Martin Hagan: Neural Network Design 2nd ed, 2014
- Wikipedia