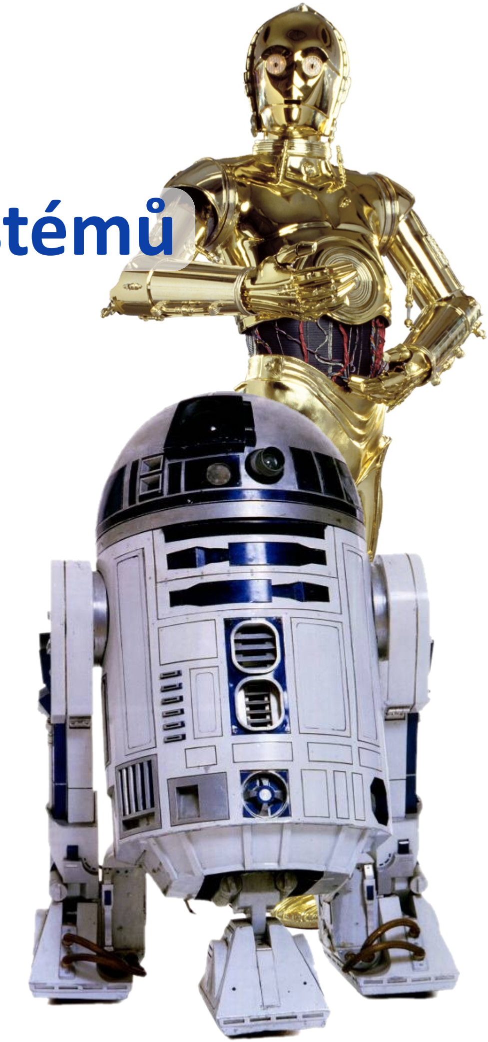
Teorie kognitivních systémů

09 Umělé neuronové sítě

- Historie, biologický předobraz
- Matematické modely neuronu
- Neuronová síť, vrstevnaté sítě
- Klasifikace neuronovou sítí
- Cenová funkce
- Optimalizace cenové funkce
- Učení neuronových sítí
- Algoritmus Backpropagation



Umělé neuronové sítě
Úvod

Umělá neuronová síť (*Artificial Neural Network*) – matematický simulační model informačního procesoru inspirovaný biologickými neuronovými sítěmi, zejména lidskou nervovou soustavou a mozkem.

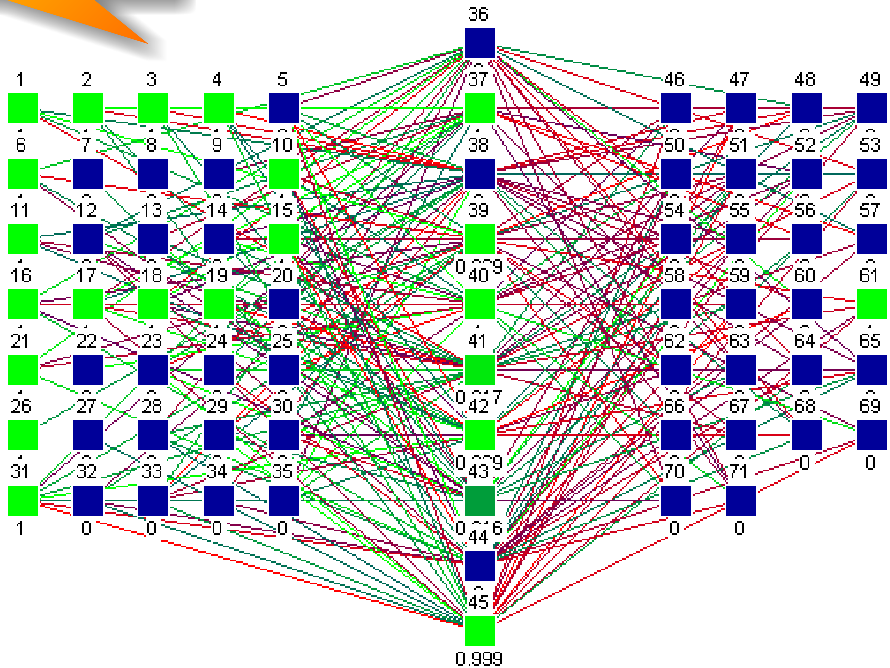
Neuronová síť se skládá z množiny **umělých neuronů**, které mohou být uspořádány do **vrstev** (podle účelu) a navzájem propojeny **synapsemi**, jejichž významnost určují **synaptické váhy**.

Neuronová síť je **adaptivní systém**, který mění svoji strukturu během fáze učení. Modeluje komplexní relace mezi vstupy a výstupy, popř. hledá ve vstupních datech vzory...

Umělé neuronové sítě
Ilustrační příklad

Neuronová síť je matematický model definující funkci

$$f\colon \mathbf{x} \in \mathbf{R}^n \rightarrow \mathbf{y} \in \mathbf{R}^m$$



Umělé neuronové sítě
Historický vývoj

- biologický předobraz: Jan Evangelista Purkyně, Ramon y Cajal
- vizionářský článek Američana W. S. McCullocha z roku 1921 o umělém napodobování biologických neuronových sítí
- W. S. McCulloch & W. Pitts představili v roce 1943 první reálně použitelný matematický model neuronu
- D. Hebb, 1949: zákon učení sítí neuronů, konstrukce prakticky použitelných učících algoritmů
- B. Widrow: vrstevnaté sítě ADALINE, MADALINE
- F. Rosenblatt: učící algoritmus pro vrstevnaté sítě s dopředným přenosem signálu
- M. Minsky & S. Papert, 1969: kniha *Perceptrons: An Introduction to Computational Geometry* “dokazuje” omezení jednovrstvé lineární neuronové sítě (perceptronu) → útlum zájmu o problematiku až do konce 80. let 20. stol.

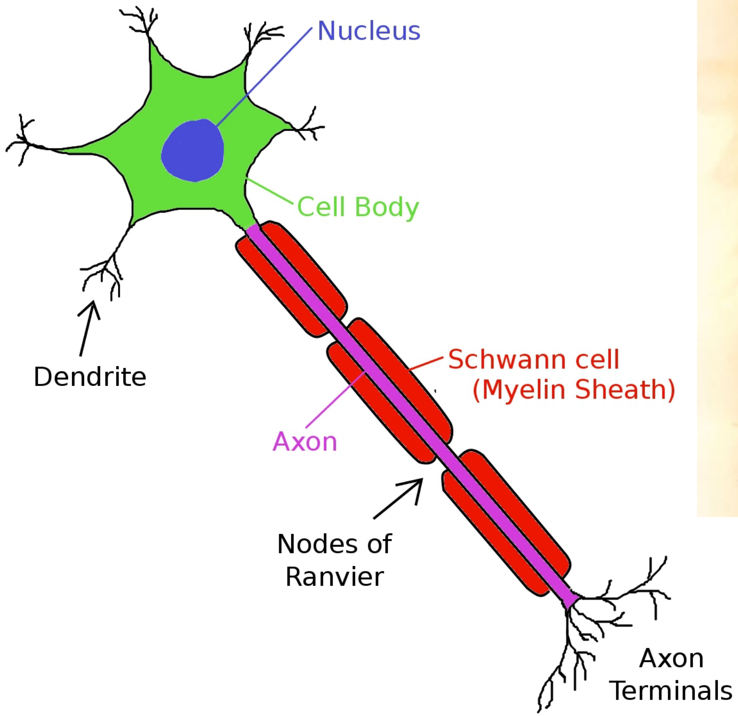
Umělé neuronové sítě
Historický vývoj

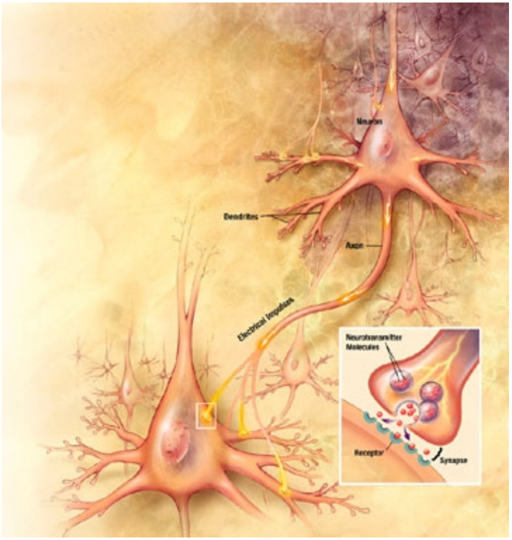
- v polovině 80. let 20. stol. dochází k masivní renesanci zájmu o ANN (mj. díky DARPA)
- znovuobjevení algoritmu **back-propagation** (1986) urychlilo rozvoj reálně nasaditelných systémů na bázi ANN
- Kunihiko Fukušima (1980): Neocognitron – „standardní architektura pro počítačové vidění”, práce inspirovaná analýzou funkce buněk primárního vizuálního kortexu
- Cybenko (1989), Hornik (1991) a jiní dokazují (tentokrát již korektně) matematické vlastnosti neuronových sítí
- J. Schmidhuber (2009 – 2012): nové architektury neuronových sítí Recurrent NN a Deep Feedforward NN vyhrávají mj. soutěže v rozpoznávání spojitého rukopisu, analýze textu, ap.

Umělé neuronové sítě
Oblasti úspěšných aplikací

- výzkum v oblasti modelování informačních systémů živých organismů
- výzkum procesů učení, testování, adaptace a generalizace
- predikce časových řad (ekonomie, ekologie, energetika, doprava, meteorologie, logistika)
- analýza signálů (EKG, EEG, akustika, ...)
- de-/komprese a kódování signálů (telekomunikace)
- adaptivní řízení technologických celků
- automatické řízení autonomních systémů
- systémy pro podporu rozhodování a expertní systémy
- modelování komplexních jevů (katastrofy, ekonometrie)
- kontrola kvality a automatické testování
- ...

Biologický neuron
Předobraz matematického modelu

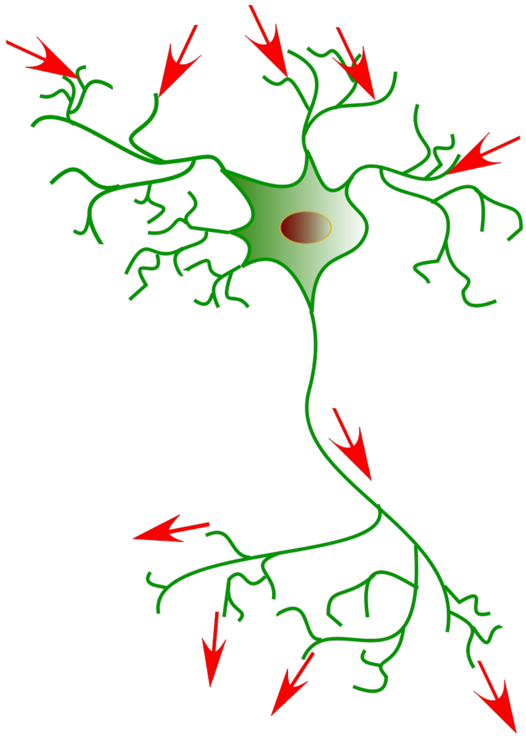


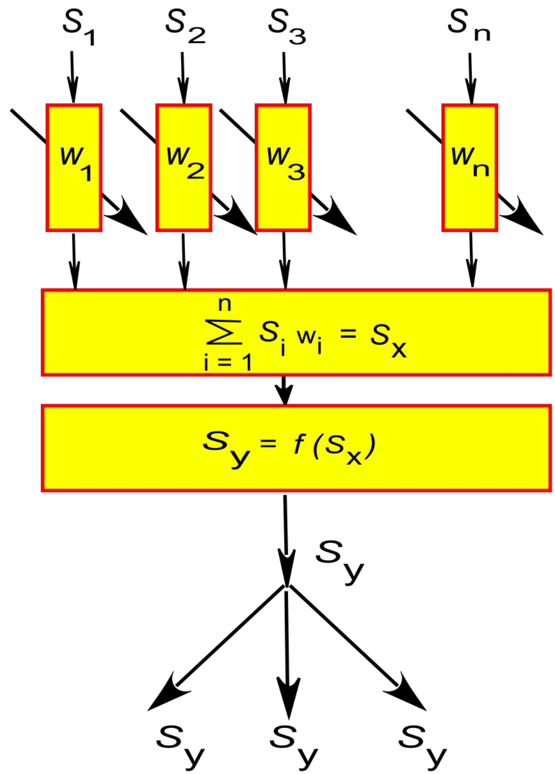


(Obrázek © Wikimedia Commons, Nick Gorton, 2005)

(Obrázek © US National Institutes of Health, National Institute on Aging)

Biologický neuron
Matematický model

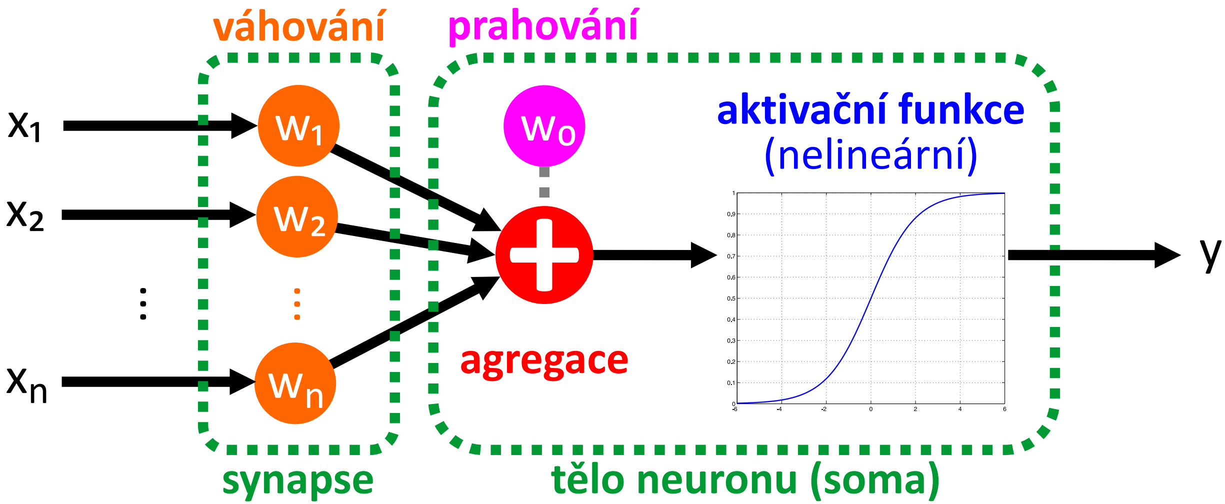




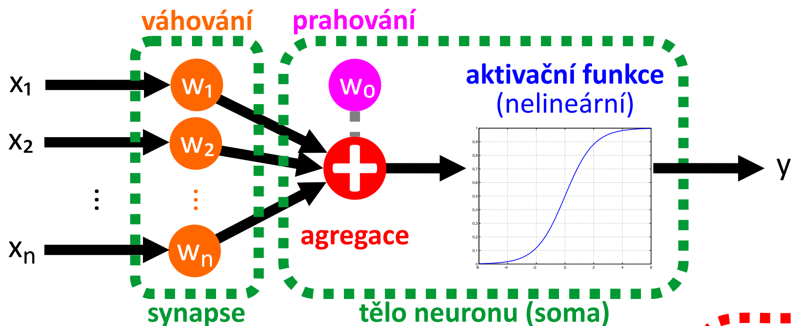
(Obrázek © Wikimedia Commons, Joanna Kośmider, 2011)

Teorie kognitivních systémů

McCulloch-Pittsův neuron
Matematický model umělého neuronu



McCulloch-Pittsův neuron
Matematický model umělého neuronu



konfluence

Synaptické operace: $$\mathbf{z}(t) = \mathbf{x}(t) \oplus \mathbf{w}(t) = \left(x_i(t) \cdot w_i(t)\right)_{i=1}^{n}$$

Somatické operace:

g(**z**)

$$\left\{ \begin{aligned} u(t) &= \mathop{\mathrm{G}}_{i=1}^{n} \mathbf{z}(t) = \sum_{i=1}^{n} z_i(t) \\ v(t) &= u(t) - w_0 \\ y(t) &= \frac{1}{1 + e^{-v(t)}} \end{aligned} \right.$$

RBF neuron
Aktivační funkce s radiální bází

LBF (*Linear Basis Function*)
– dělí vstupní prostor lineárně prostřednictvím n-rozměrných nadrovin

$$u(t) = \sum_{i=1}^{n} x_i(t) w_i(t)$$

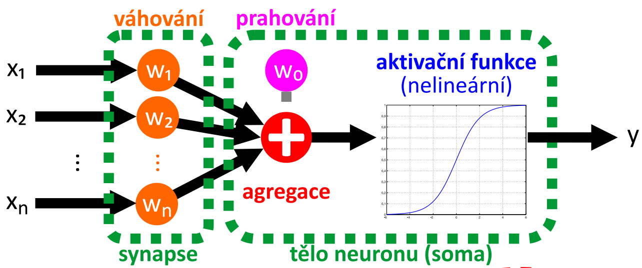
➔ standardní McCulloch-Pittsův neuron

Modifikace:

RBF (*Radial Basis Function*)
– dělí vstupní prostor prostřednictvím hyperkoulí

$$u(t) = \sqrt{\sum_{i=1}^{n} \left(x_i(t) - w_i(t)\right)^2}$$

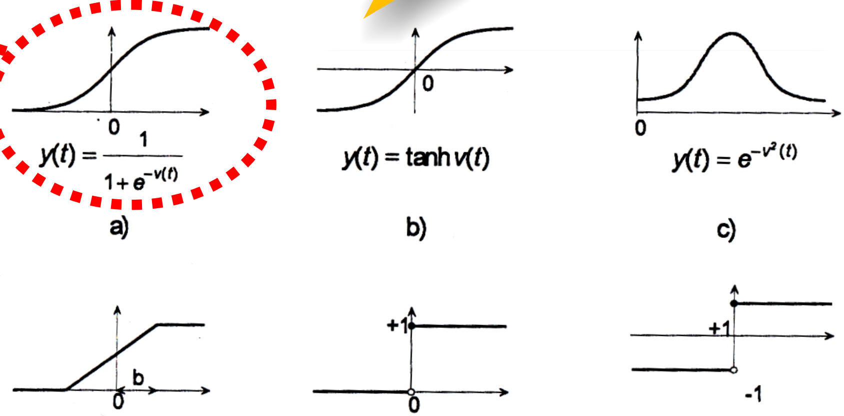
Umělý neuron
Používané typy aktivačních funkcí



Výběr vhodné aktivační funkce závisí na charakteru řešené úlohy...

sigmoida je vhodná volba, protože je spojitá, a tedy diferencovatelná v celém **R**

některé jsou sice výpočetně jednodušší, ale mají jiné nevýhody



a) $y(t) = \dfrac{1}{1+e^{-v(t)}}$

b) $y(t) = \tanh v(t)$

c) $y(t) = e^{-v^2(t)}$

d) $y(t) = 0 \quad v(t) \le -b$; $y(t) = c \cdot v(t) \quad -b < v(t) < b$; $y(t) = 1 \quad v(t) \ge b$

e) $y(t) = 0 \quad v(t) < 0$; $y(t) = 1 \quad v(t) \ge 1$

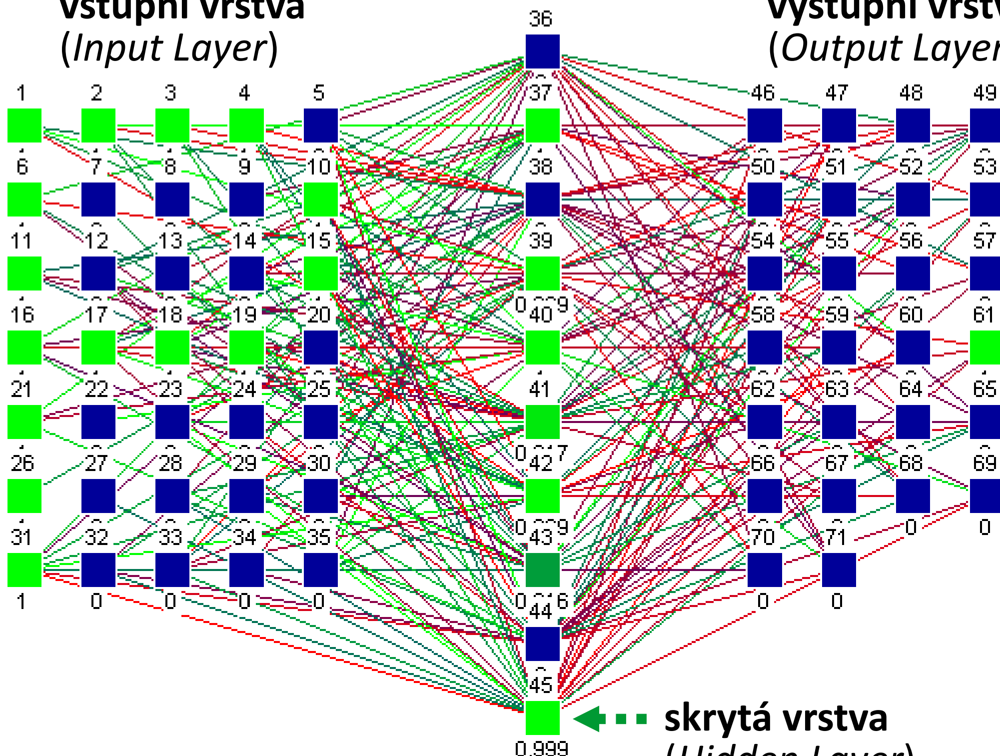
f) $y(t) = -1 \quad v(t) < 0$; $y(t) = 1 \quad v(t) \ge 1$

Neuronová síť
(Vrstevnatá) struktura z umělých neuronů

Vícevrstvý perceptron (*Multi-Layer Perceptron, MLP*)

vstupní vrstva
(*Input Layer*)

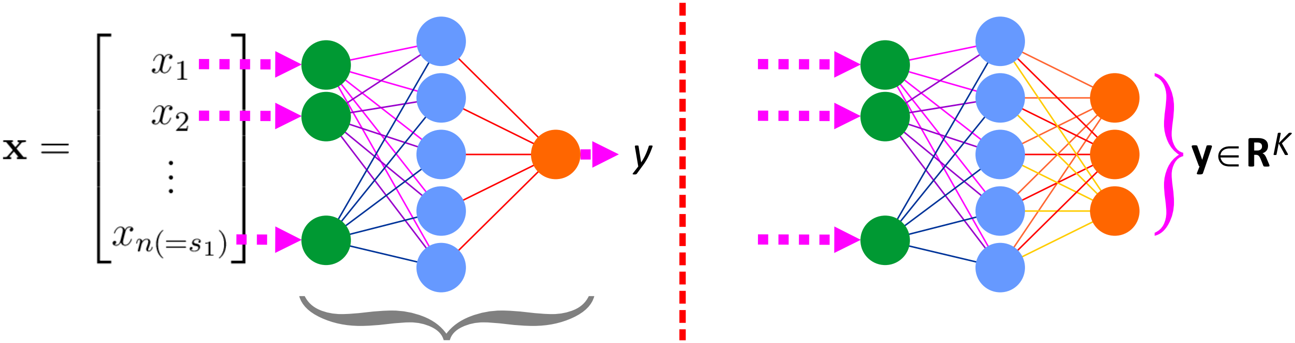
výstupní vrstva
(*Output Layer*)



skrytá vrstva
(*Hidden Layer*)

Klasifikace neuronovou sítí
Binární vs multi-class klasifikace

Binární klasifikace | **Multi-class klasifikace**



L – celkový počet vrstev sítě
(více než 3 nemá smysl ← Cybenko)

1 neuron ve výstupní vrstvě, tj. $y = \{0, 1\}$; $h_\Theta(\mathbf{x}) \in \mathbf{R}$; počet neuronů v *L*-té vrstvě $s_L = 1$, počet tříd $K = 1$ (aktivace výstupního neuronu indikuje, zda vzorek do třídy ne-/patří)

K neuronů ve výstupní vrstvě, tj. $\mathbf{y} = \mathbf{R}^K$; $h_\Theta(\mathbf{x}) \in \mathbf{R}^K$; $s_L = K$;

$$\mathbf{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

indikovaná třída
(max)

Cenová funkce pro neuronovou síť
Generalizace cenové funkce logistické regrese

Cenová funkce logistické regrese:

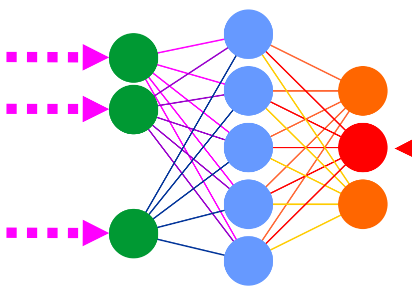
$$J(\boldsymbol{\Theta}) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_{\Theta}\left(\mathbf{x}^{(i)}\right) + \left(1-y^{(i)}\right) \log\left(1-h_{\Theta}\left(\mathbf{x}^{(i)}\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\Theta_j^2$$

Cenová funkce neuronové sítě:

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_{\Theta}(\mathbf{x}^{(i)})\right)_k + \left(1-y_k^{(i)}\right)\log\left(1-\left(h_{\Theta}(\mathbf{x}^{(i)})\right)_k\right)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{ji}^{(l)}\right)^2$$

požadovaná aktivace



$$\begin{bmatrix}0\\1\\0\end{bmatrix} = (y_k)$$

$(h_{\Theta}(x))_i$ – i-tý výstup

$h_{\Theta}(x) \in \mathbb{R}^K$

Výpočet gradientu
Minimalizace cenové funkce

Cenová funkce neuronové sítě:

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_\Theta(\mathbf{x}^{(i)})\right)_k + \left(1 - y_k^{(i)}\right)\log\left(1 - \left(h_\Theta(\mathbf{x}^{(i)})\right)_k\right)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{ji}^{(l)}\right)^2$$

Cílem optimalizace je: $\min_{\Theta} J(\boldsymbol{\Theta})$ – minimalizace cenové funkce...

Potřebujeme umět spočítat:

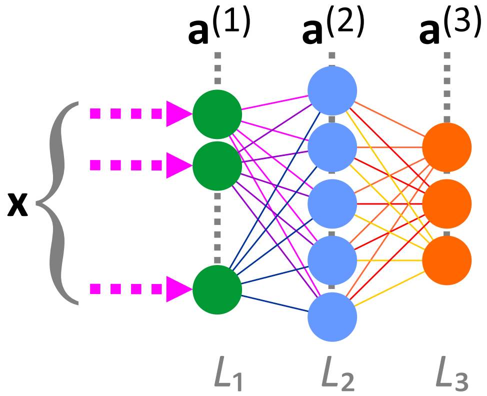
- $J(\boldsymbol{\Theta})$
- $\boxed{\dfrac{\partial J(\boldsymbol{\Theta})}{\partial \Theta_{ij}^{(l)}}}$

pro potřeby formalizace a uvedení do souvislosti s předchozími optimalizacemi jsou váhy w_{ij} přeznačeny na $\Theta_{ij}^{(l)}$

„přímý” analytický výpočet – problém

Výpočet parciální derivace $J(\Theta)$
Dopředné šíření (*Forward Propagation*)

Je-li dán jeden trénovací vzorek ($\mathbf{x}$, y), pak vektorizovaná verze výpočtu dopředného šíření aktivace sítí vypadá:



$$\mathbf{a}^{(1)} = \mathbf{x}$$

$$\mathbf{z}^{(2)} = \mathbf{\Theta}^{(1)}\mathbf{a}^{(1)}$$

$$\mathbf{a}^{(2)} = g\left(\mathbf{z}^{(2)}\right)$$ odečteme práh $a_0^{(2)}$

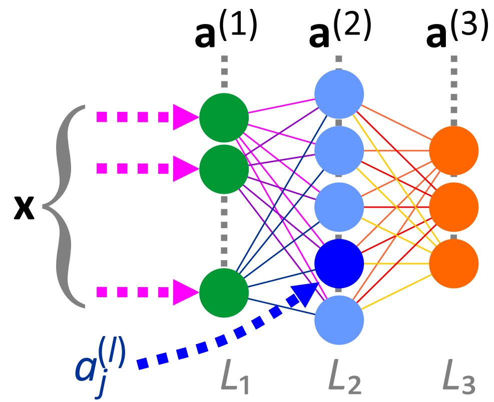
$$\mathbf{z}^{(3)} = \mathbf{\Theta}^{(2)}\mathbf{a}^{(2)}$$

$$\mathbf{a}^{(3)} = g\left(\mathbf{z}^{(3)}\right) = h_\Theta(\mathbf{x})$$

→ vypočítá aktivaci všech neuronů v celé síti

Výpočet parciální derivace $J(\boldsymbol{\Theta})$
Zpětné šíření (*Back-Propagation*)

Učící algoritmus **Backpropagation** vlastně provádí numerický výpočet derivace $J(\boldsymbol{\Theta})$:



$\delta_j^{(l)}$ = „chyba" j-tého neuronu v l-té vrstvě

$$\left.\begin{aligned} \delta_j^{(3)} &= a_j^{(3)} - y_j \\ a_j^{(3)} &= \left(h_\Theta(\mathbf{x})\right)_j \end{aligned}\right\}$$

pro vstupní vrstvu

násobení vektoru po složkách

$$\left.\delta^{(2)} = \left(\boldsymbol{\Theta}^{(2)}\right)^{\mathsf{T}} \delta^{(3)} .* \frac{\mathrm{d}g\left(\mathbf{z}^{(2)}\right)}{\mathrm{d}\mathbf{z}^{(2)}}\right\}$$

pro skrytou vrstvu

Vektor chyb $\boldsymbol{\delta}^{(1)}$ neexistuje – jsou to přímo vstupy do sítě, tj. nelze mluvit o chybě...

Výpočet parciální derivace *J*(**Θ**)
Souvislost zpětného šíření s par. deriv. *J*(**Θ**)

Derivace aktivační funkce:

$$\frac{\mathrm{d}g\left(\mathbf{z}^{(l)}\right)}{\mathrm{d}\mathbf{z}^{(l)}} = \mathbf{a}^{(l)} .* \left(1 - \mathbf{a}^{(l)}\right) \equiv \frac{\mathrm{d}\sigma(x)}{\mathrm{d}x} = \sigma(x)\left(1 - \sigma(x)\right)$$

protože

$$s(x) = \frac{1}{\sigma(x)} = 1 + e^{-x} \qquad s'(x) = \frac{-\sigma'(x)}{\sigma(x)^2}$$

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

Použitím poměrně komplikované matematiky lze ukázat, že

$$\boxed{\frac{\partial J(\mathbf{\Theta})}{\partial \Theta_{ij}^{(l)}} = a_j^{(l)} \delta_i^{l+1}}$$

za předpokladu ignorování regularizačního parametru, tj. λ = 0.

Algoritmus Backpropagation
Optimalizace *L*-vrstvé neuronové sítě

Trénovací množina $\left\{\left(\mathbf{x}^{(1)},\mathbf{y}^{(1)}\right),\left(\mathbf{x}^{(2)},\mathbf{y}^{(2)}\right),\ldots,\left(\mathbf{x}^{(m)},\mathbf{y}^{(m)}\right)\right\}$

Nastavíme $\Delta_{ij}^{(l)} \leftarrow 0, \forall i,j,l$

for $i \leftarrow 1 \ldots m$ {

$\mathbf{a}^{(1)} \leftarrow \mathbf{x}^{(i)}$

výpočet dopředného šíření, abychom získali $\mathbf{a}^{(l)}$ pro $l = 2, \ldots, L$

výpočet chyby ve výstupní vrstvě $\delta^{(L)} \leftarrow \mathbf{a}^{(L)} - \mathbf{y}^{(i)}$

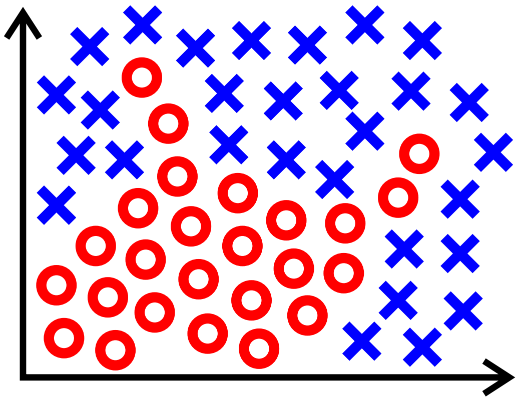
výpočet zpětného šíření chyby $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$

$$\Delta_{ij}^{(l)} \leftarrow \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$$

}

$$\left.\begin{array}{ll} D_{ij}^{(l)} \leftarrow \frac{1}{m}\Delta_{ij}^{(l)} + \lambda\Theta_{ij}^{(l)} & \text{jestliže } j \neq 0 \\ D_{ij}^{(l)} \leftarrow \frac{1}{m}\Delta_{ij}^{(l)} & \text{jestliže } j = 0 \end{array}\right\} \qquad \frac{\partial J(\mathbf{\Theta})}{\partial \Theta_{ij}^{(l)}} = D_{ij}^{(l)}$$

Použití neuronových sítí
Motivace – nelineární klasifikace



Další typy neuronových sítí
Náměty k samostudiu

- **SOM** (*Self-Organizing Map*), též **Kohonenova síť**
- **LAM** (*Linear Associative Memory*)
- **ART** (*Adaptive Resonance Theory*)
  ART 1, 2, 2-A, 3, Fuzzy ART, ARTMAP
- **Cognitron** a **Neocognitron**
- **Hopfieldova síť**
- ADALINE/MADALINE