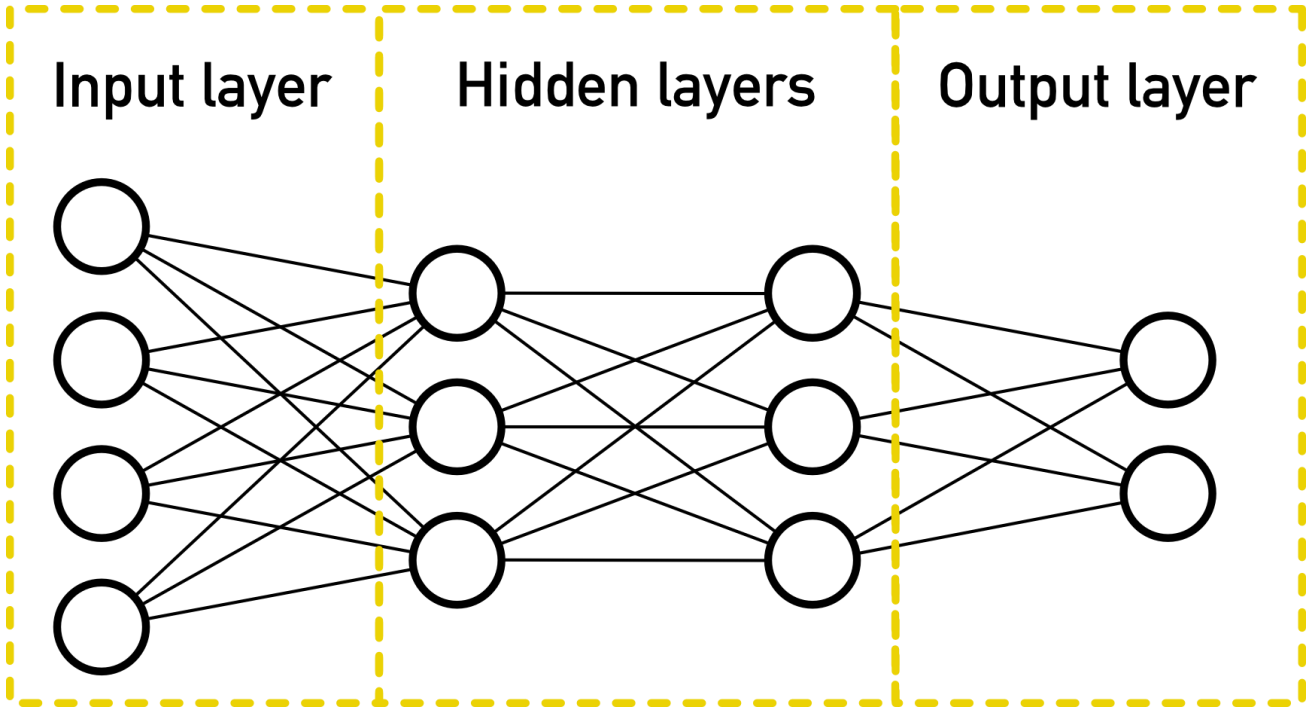
AI pozadi

By Filip ☺

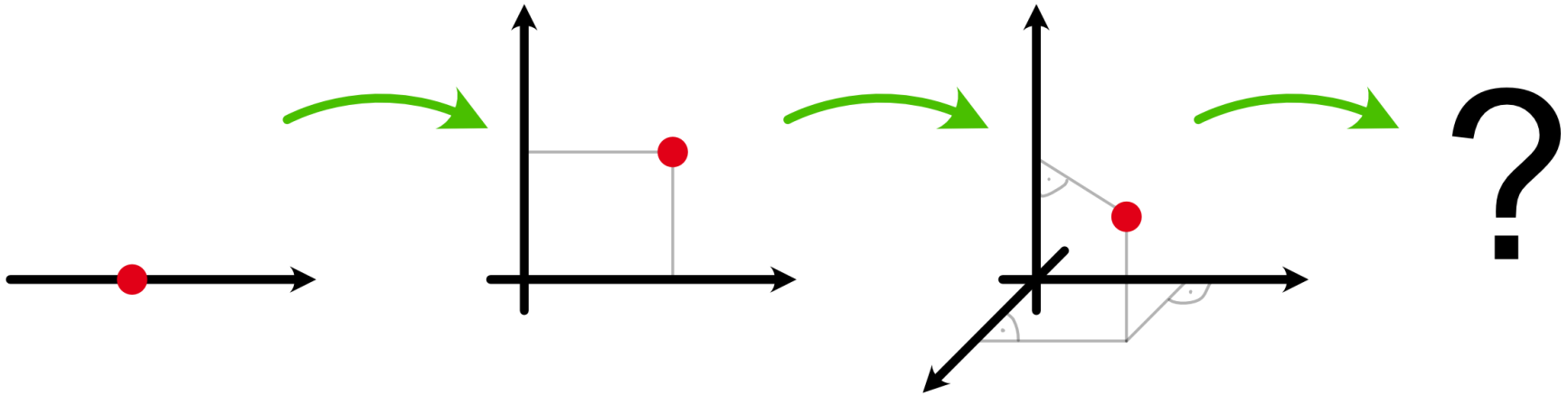
Neuronová síť (NN)

- Inspirována lidským mozkem
- Skládají se z neuronů
- více typů, které se liší strukturou (RNN, CNN)
- Pro jednoduchost se budeme věnovat pouze NN s tzv. MLP strukturou



Data

- Jsou důležitou součástí AI úloh nejrůznějšího typu
- Je dobré porozumět jak data v digitálním světě vypadají
- Komplexnější data tvoří v podstatě tzv n-dimenzionální prostředí



- Přejděte na **Počet_kategorii.ipynb** (Dodat odkaz)

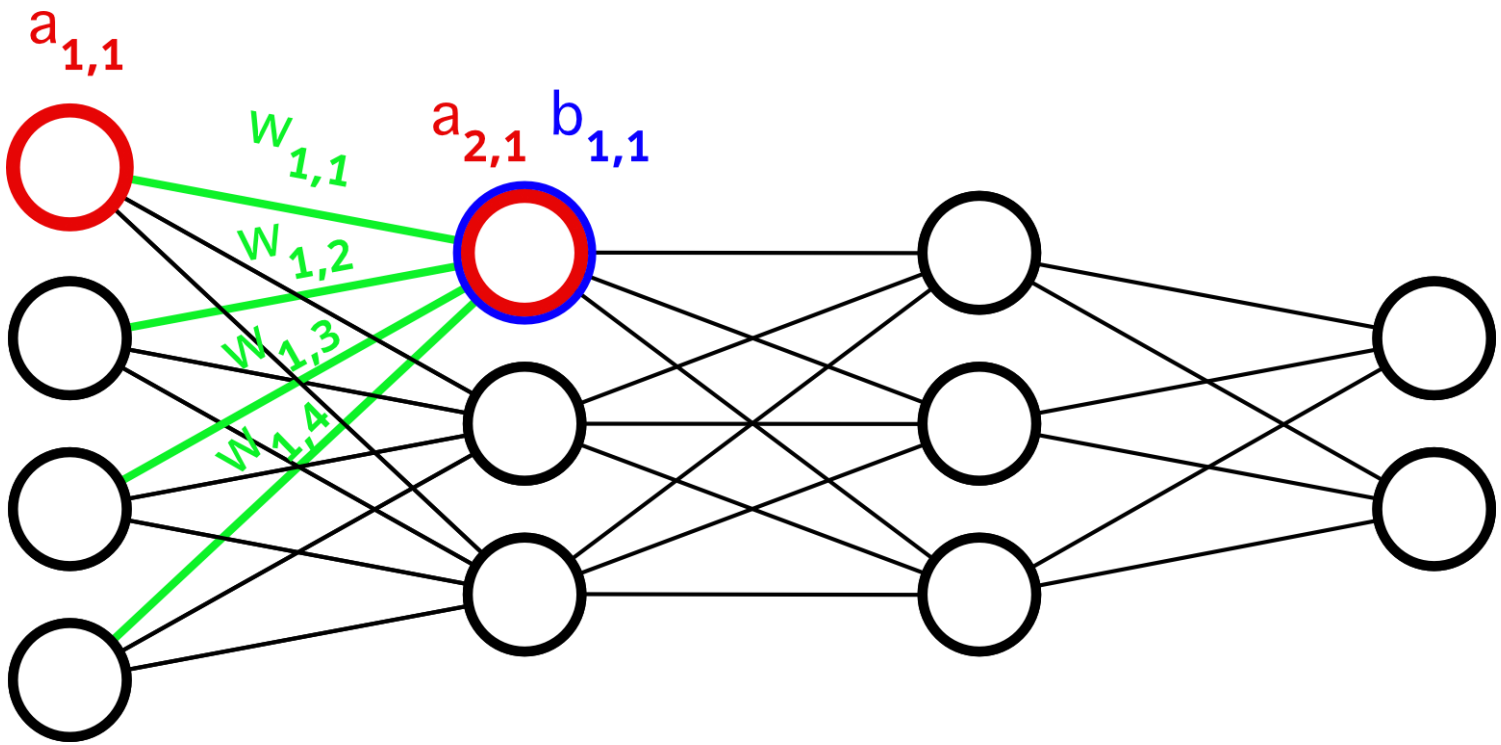
Konstrukce NN

- Je nejednoznačná
- Experimentování s konstrukcí je možné na playground.tensorflow.org/
- Funkce určující hodnotu následujícího neuronu je…

$$a_2 = F(w_1 * a_1 + b_2)$$

Jak funguje NN

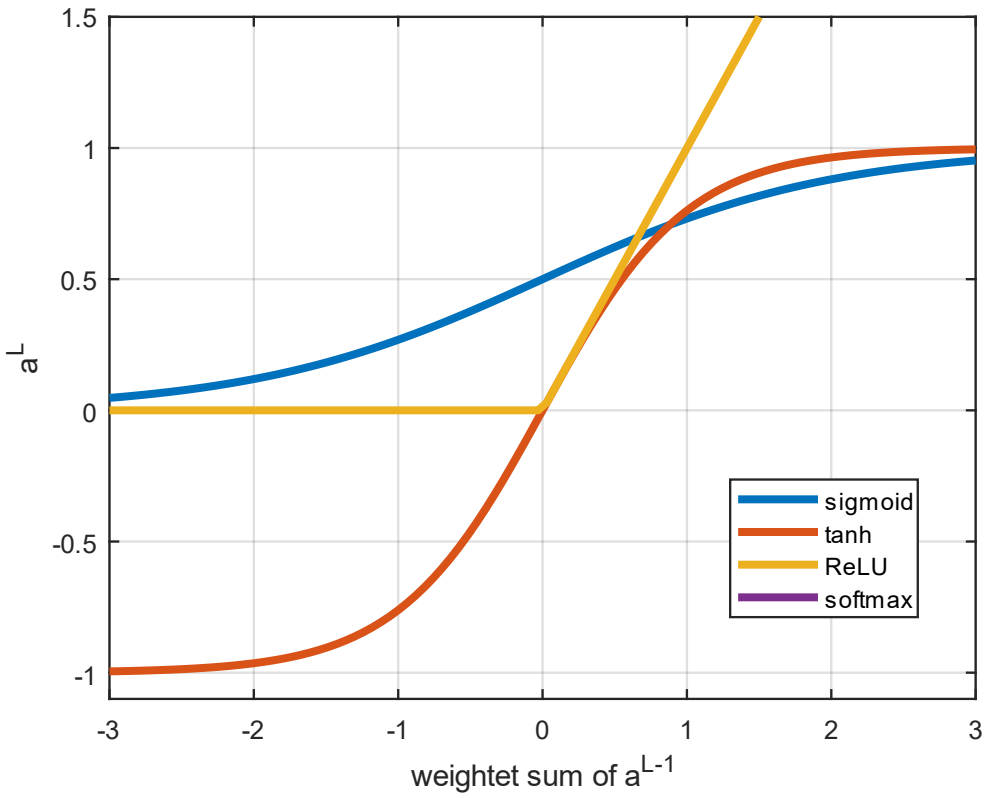
$$a_{2,1} = F(w_{1,1} * a_{1,1} + \ldots + w_{1,4} * a_{1,4} + b_{1,1})$$



Co to je „activation function“

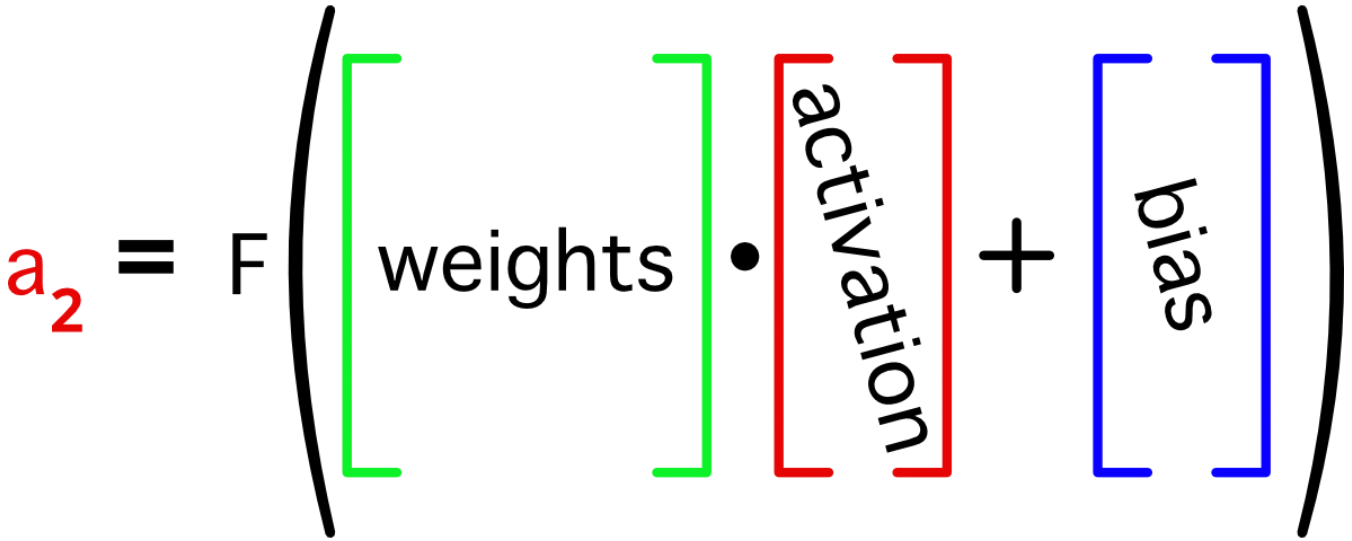
$$a_2 = F(w_1 * a_1 + b_2)$$

- Sigmoid – zastaralá, pomalá
  - $F(x) = \frac{1}{1+e^{-x}}$
- ReLU
  - $F(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$
- tanh
  - $F(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
- Softmax
  - $F(x_j) = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}, j = 1,2,\dots,K$



Vstup do NN

- Vstupem do NN jsou vektory
- Zjednodušení výpočtů maticovým počtem

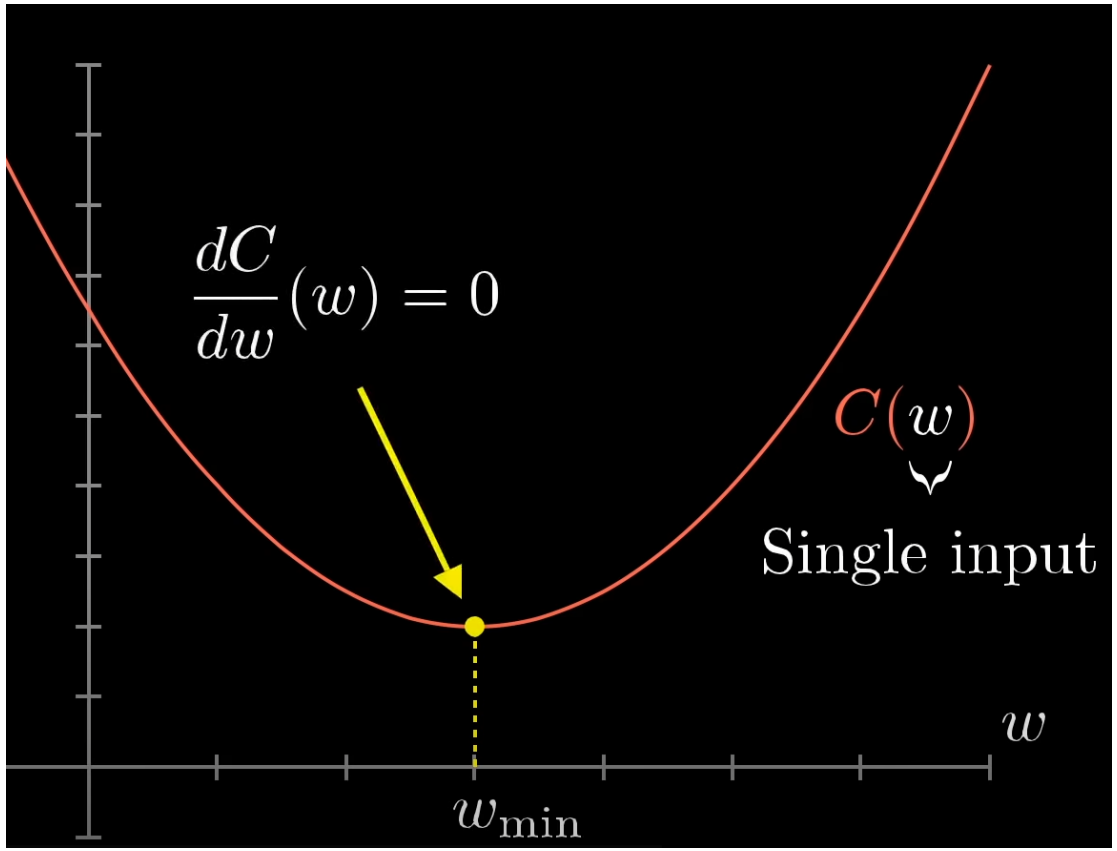


Trénování NN

- Pro trénování potřebujeme tzv datasety
- Před trénováním náhodně nastavíme všechny weights a biases
- Na vstup začneme posílat trénovací data
- Měříme přesnost neuronové sítě tzv. cost funkcí
- Cost funkce
  - Rozdíl mezi výstupem a správným výsledkem např. $(\text{output} - \text{result})^2$
  - Průměruje se přes všechna trénovací data
  - Parametry jsou weight a bias
  - Cílem je tuto funkci minimalizovat na co nejnižší hodnotu (lokální minimum)

Trénování NN - minimum

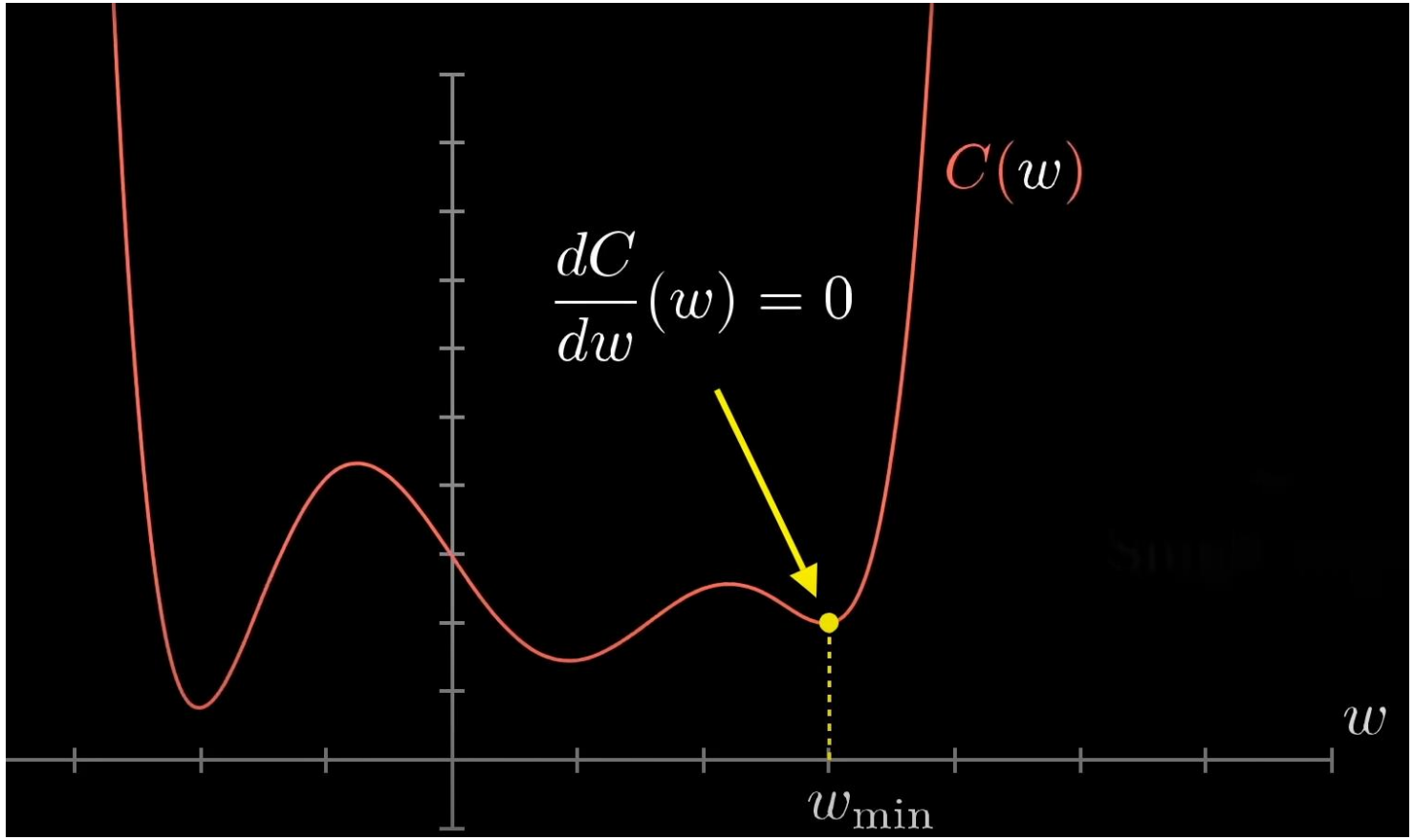
- Proč nehledáme rovnou globální minimum?



Zdroj obrázku: https://youtu.be/IHZwWFHWa-w

Trénování NN - minimum

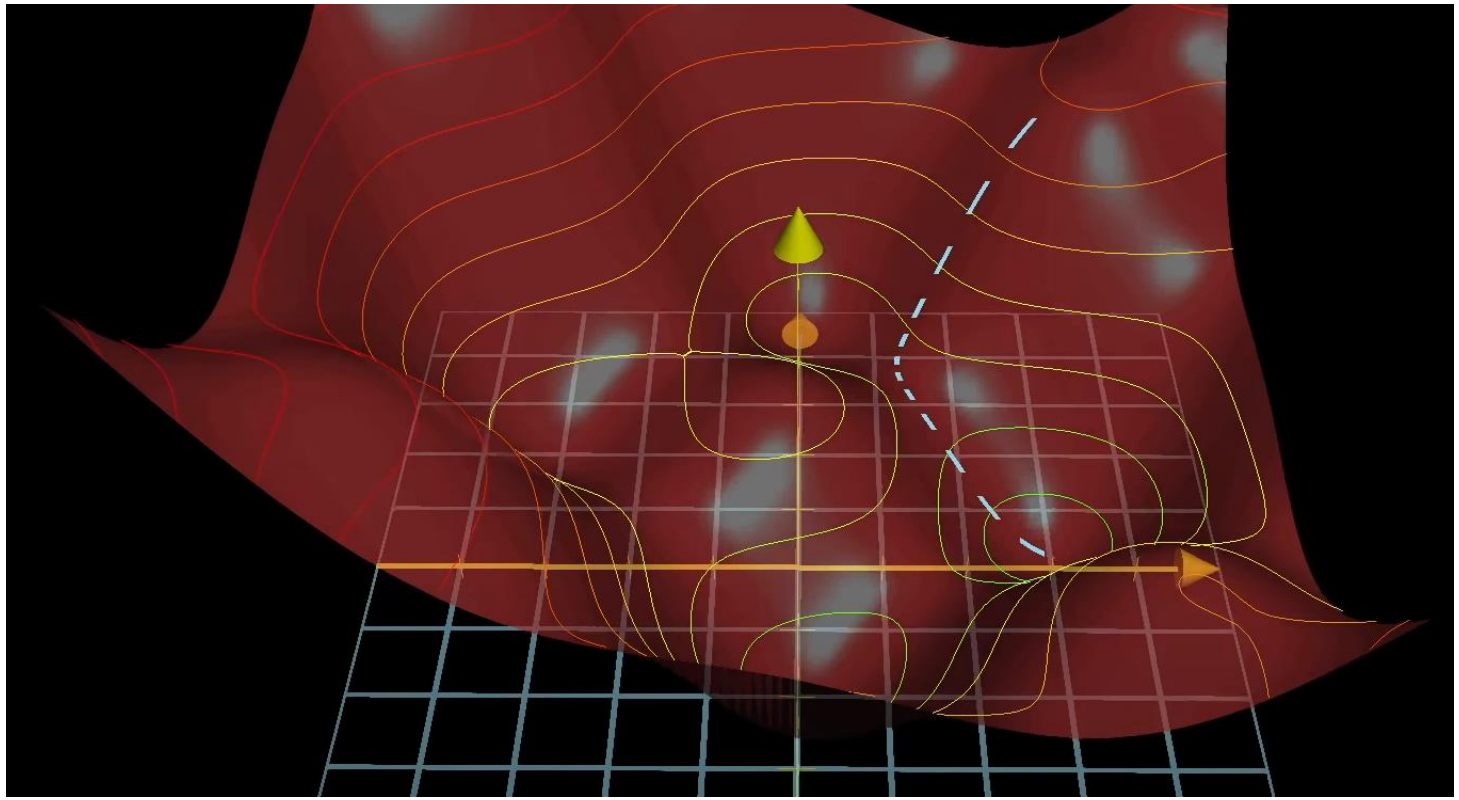
- Proč nehledáme rovnou globální minimum?



Zdroj obrázku: https://youtu.be/IHZwWFHWa-w

Trénování NN - minimum

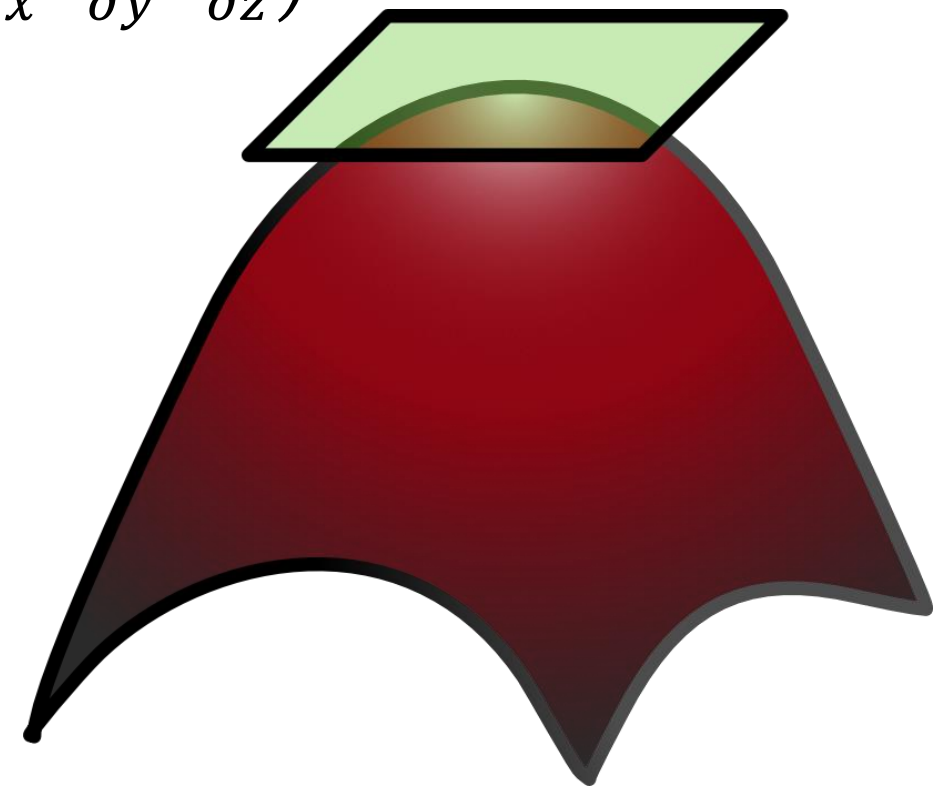
- Proč nehledáme rovnou globální minimum?



Zdroj obrázku: https://youtu.be/IHZwWFHWa-w

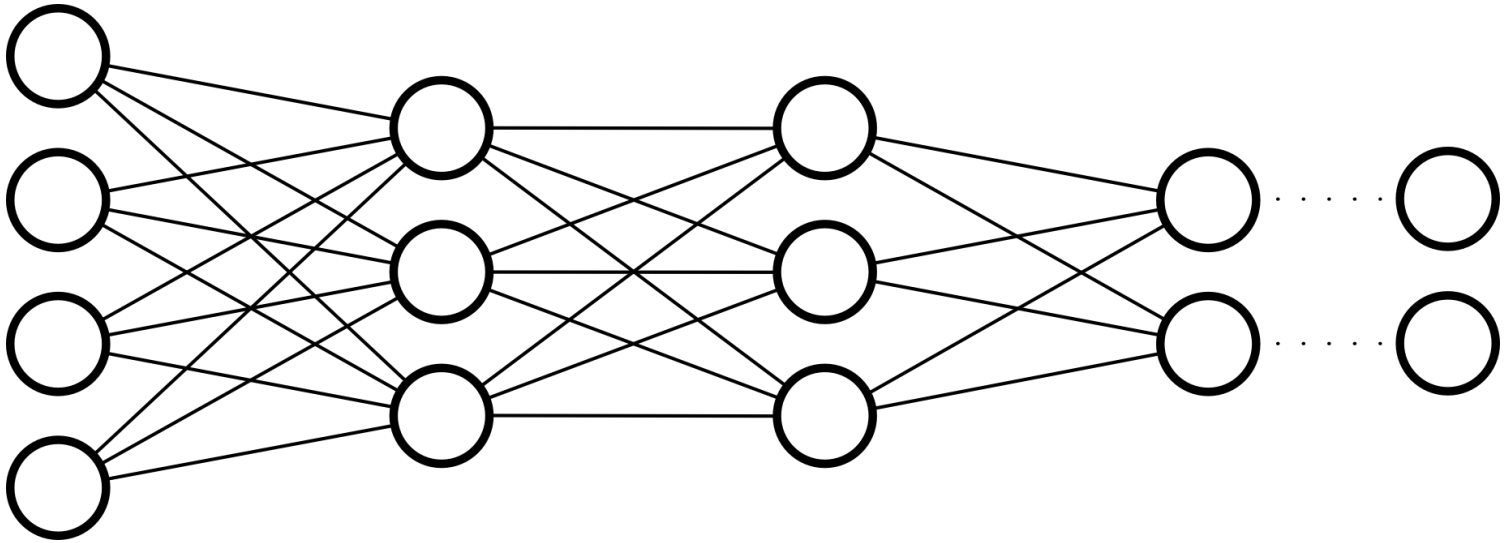
Trénování NN - gradient

- Parciální derivace cost funkce
- $\nabla\phi = \operatorname{grad}\phi = \left(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}, \frac{\partial\phi}{\partial z}\right)$



Trénování NN - backpropagation

- Nastavení parametrů vzhledem k výsledku
- Vychází se z gradientu cost funkce
- Alternativně lze chápat jako zpětnou propagaci korekce NN vůči anotaci

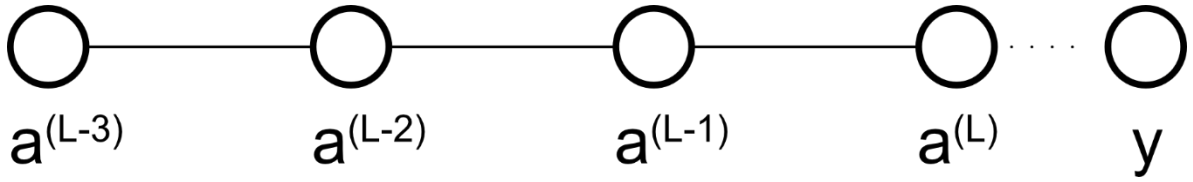


Trénování NN - batches

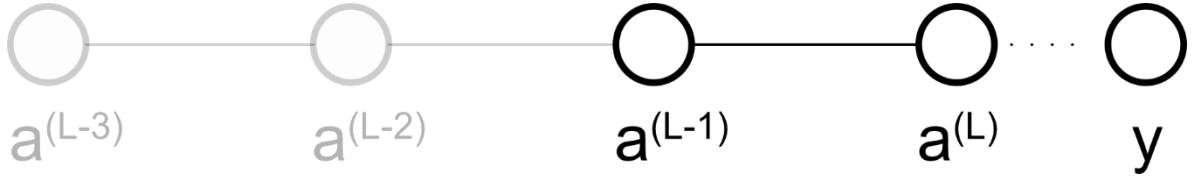
- Omezený výpočetní výkon → nelze zjistit dokonalý gradient
- Počítáme tedy tzv stochastický gradient
- Trénování proběhne na menším vzorku dat (batch)
- Po každé batch se vypočte průměrná změna jednotlivých parametrů (gradient) a podle toho se upraví parametry a úkon se opakuje pro všechna data

Trénování NN – detailed backpropagation

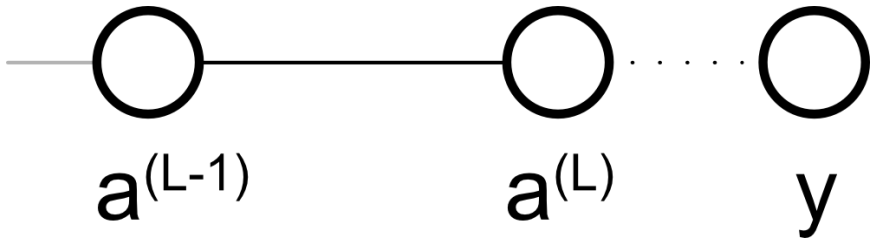
- Pro konkrétní příklad si zjednodušíme NN na tvar se 4mi neurony



- Máme tedy celkem 6 volných parametrů
- Cost funkce by tedy vypadala C(w_{L-2}, w_{L-1}, w_L, b_{L-2}, b_{L-1}, b_L)
- Budeme se soutředit na poslední dva a požadovaný výstup



Trénování NN – detailed backpropagation



- Představme si, že jsme na vstup dali první trénovací vzorek
- NN měla náhodně nastavené parametry a proto nejspíše výsledek nebude správný
- Jeden dílek celkové cost funkce pak bude vypadat takto…

$$C_0 = (a^{(L)} - y)^2$$

$$a^{(L)} = F(w^{(L)} * a^{(L-1)} + b^{(L)})$$

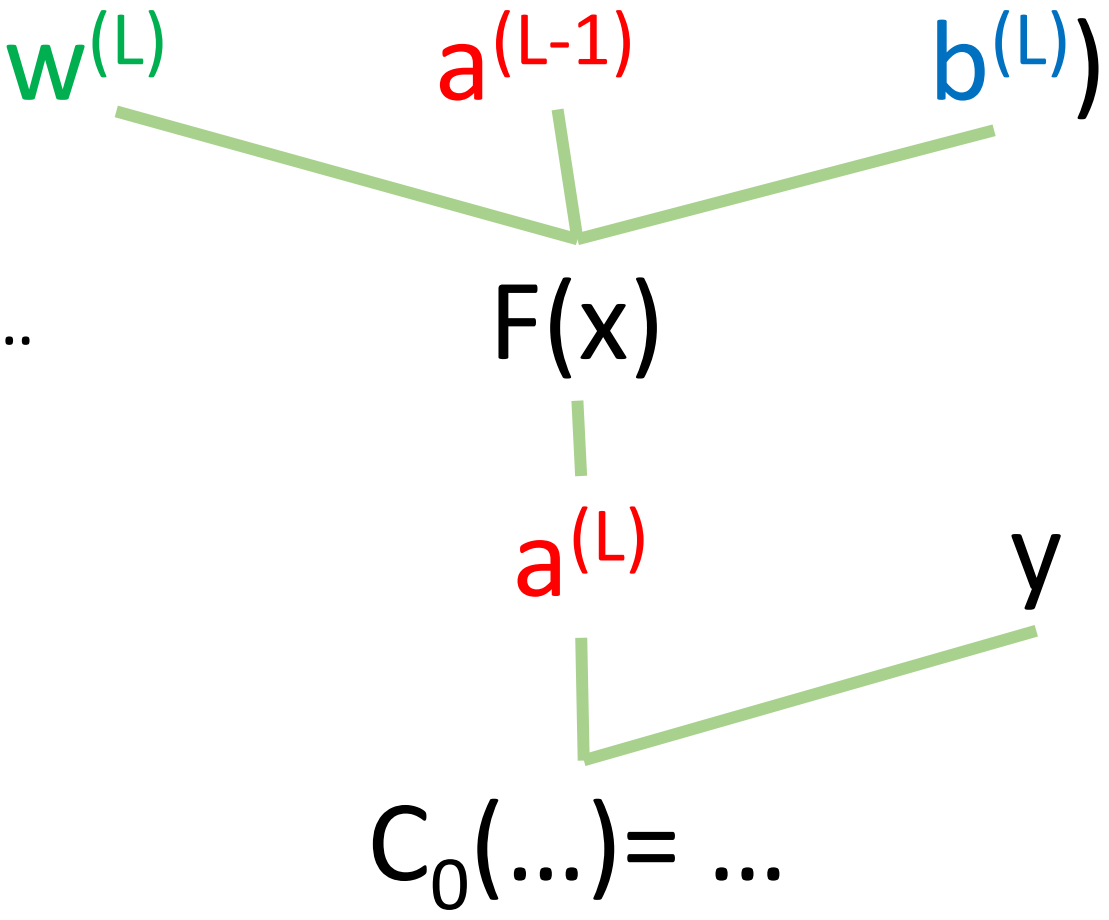
Trénování NN – detailed backpropagation

Parametry…

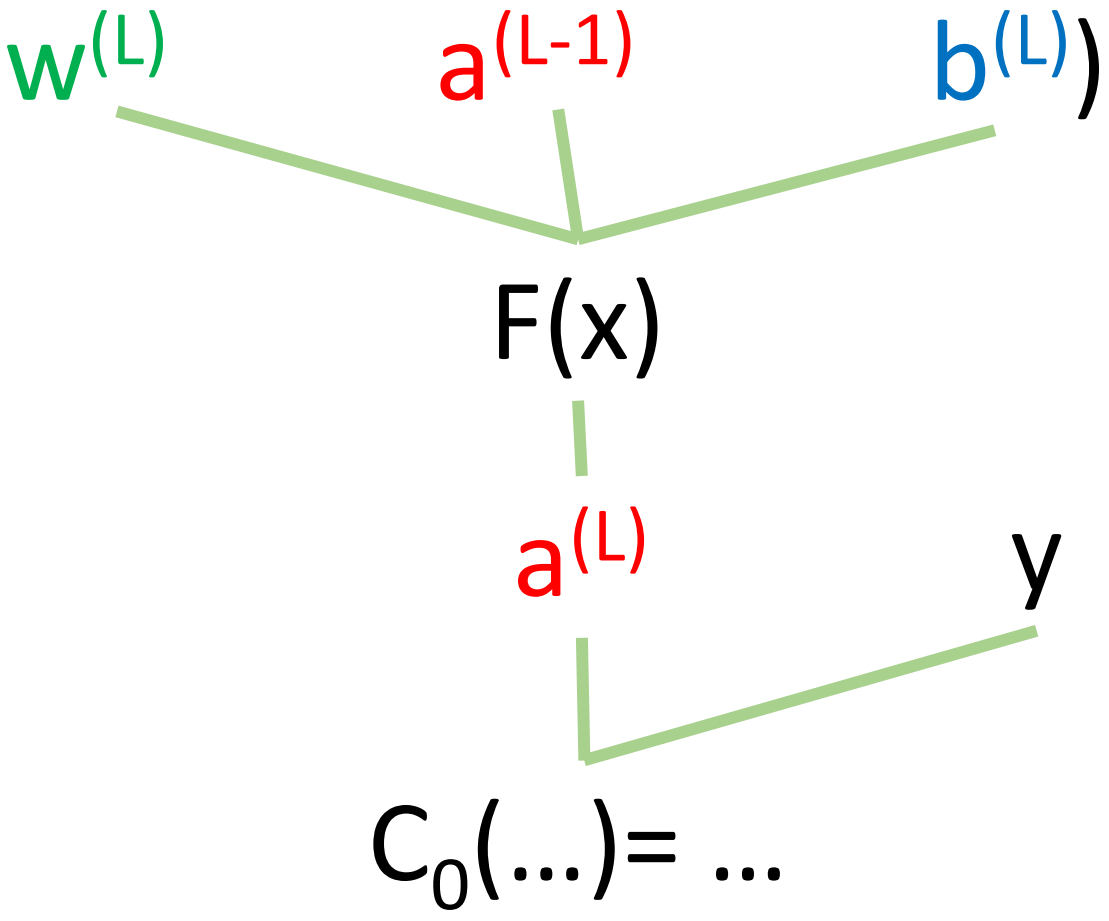
Activation funkce…

Výstup a řešení…

Cost funkce…



Trénování NN – detailed backpropagation



- Nyní zjišťujeme, jak velký má vliv změny weight na C_0 pomocí parciálních derivací.
- V podstatě hledáme poměr mezi diferenciálem w a C_0
- $\frac{\partial C_0}{\partial w^{(L)}} = \frac{\partial z^{(L)}}{\partial w^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial C_0}{\partial a^{(L)}}$
- z je vážená suma
- Obdobně se postupuje i pro bias
- Při více neuronech se používají sumy a řetězení nákresu

Tensorflow

- https://www.tensorflow.org/
- Obsahuje základní i pokročilé návody do NN
- Nabízí celý ekosystém sladěných knihoven
- Jak pro seznámení tak pro business
- Obsahuje řešení i pro mobilní a IoT zařízení v podobě Tensorflow lite
- Open-source

Proč grafické karty?

- Protože mají spoustu malých jader pro jednoduché matematické operace
- Vypočítávání grafiky ve hře počítá s maticemi -> výhoda v ML
- CPU(málo chytrých jader) vs GPU(hodně ne tak chytrých jader)
- Porovnání z pořadu Mythbusters

Děkujeme za pozornost

- V případě zájmu o další workshopy nás kontaktujte na adrese
- Nebo navštivte naše stránky https://www.nvias.org/
- Naše další aktivity můžete sledovat zde: