

# Hluboké učení

Aleš Horák

E-mail: [hales@fi.muni.cz](mailto:hales@fi.muni.cz)  
<http://nlp.fi.muni.cz/uui/>

Obsah:

- ▶ Hluboké učení
- ▶ Aplikace hlubokého učení
- ▶ Techniky hlubokého učení

# Motivace

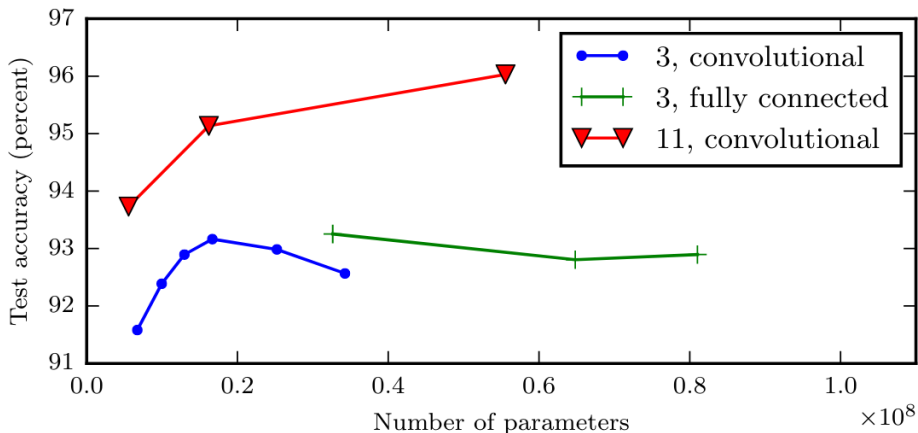
**vyjadřovací síla** klasických neuronových sítí:  
*s jednou skrytou vrstvou – všechny spojité funkce*  
*se dvěma skrytými vrstvami – všechny funkce*

Proč tedy pracovat s **hlubšími architekturami**?

# Motivace

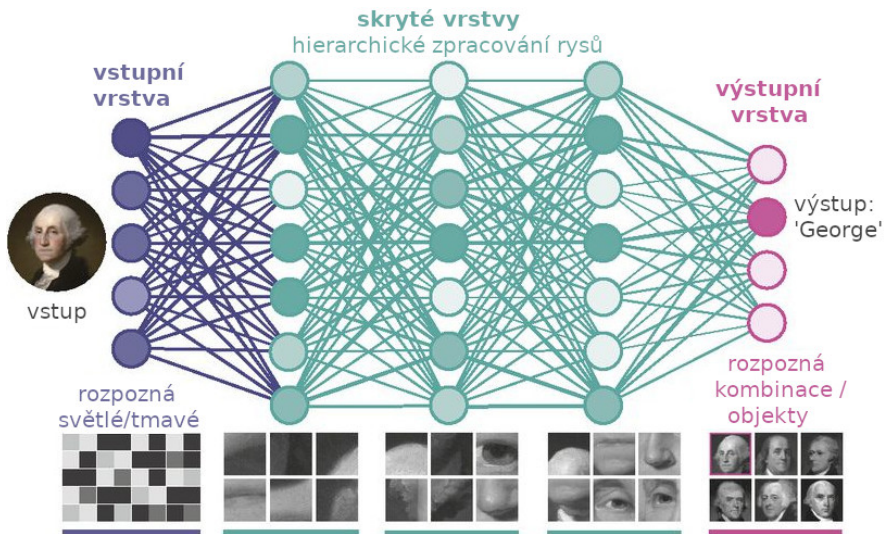
**Hluboké učení** (*Deep Learning, Deep Neural Networks*)

komplexní vstup se lépe modeluje pomocí **užší a hlubší sítě**



(příklad Goodfellow, 2017)

# Motivace



(příklad Waldrop, 2019)

# Aplikace hlubokého učení

hlavní **aplikační oblasti**

- ▶ počítačové vidění
- ▶ analýza a generování textu, *analýza signálu, analýza časové řady*
- ▶ zpětnovazební učení

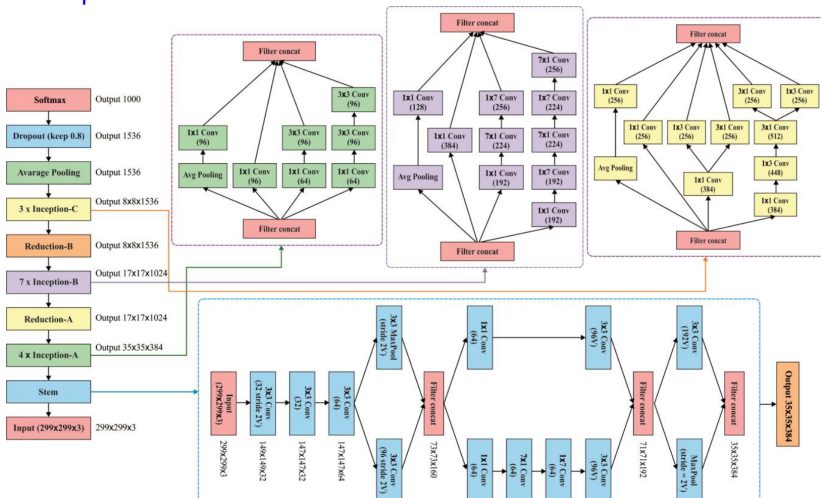
# Počítačové vidění

soutěž **ImageNet** – 1.2 mil. obrázků v 1000 kategoriích

2021 – chyba **1%**, člověk **5%**

např. **Inception V4**

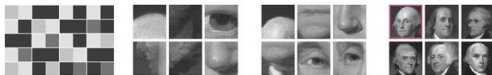
(Shankar et al, 2020)



# Konvoluční síť

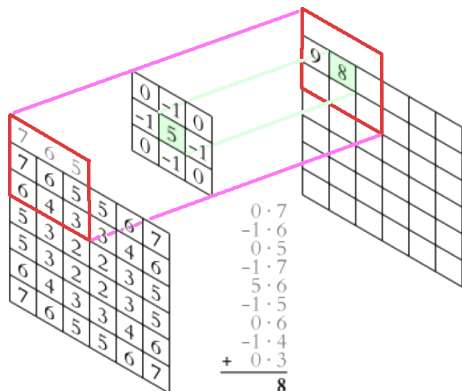
požadavky na **zpracování obrazu**:

- ▶ vztah **sousednosti** bodů
- ▶ barva – RGB kanály,  $\times 3$
- ▶ (pod)objekty a rysy – **kdekoliv** v obrázku – **prostorová invariance**



**Konvoluční síť (CNN):**

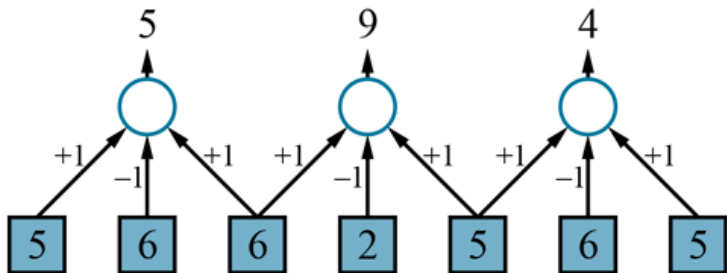
- ▶ učí se malou **kernelovou** maticí vah (*filtr*) aplikovat na body obrazu, aplikace = **konvoluce**
- ▶ kernelové matice **sdílí váhy**
- ▶ matice se posouvá o **krok** (*stride*)
- ▶ výstupem je **mapa rysů** (*feature map*)



(Ganesh, 2019)

## Parametry konvoluční vrstvy

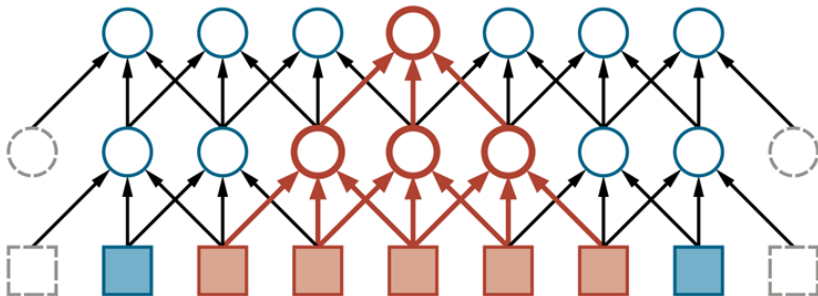
- ▶ rozměry kernelu – 1D ( $\ell$ ), 2D ( $\ell \times r$ ), 3D ...
- ▶ počet kernelů k učení  $\times d$
- ▶ krok  $s$ ,  $s \geq 2$  redukuje dimenzi





## Vícevrstvé konvoluční síť

- ▶ další konvoluční vrstvy zpracovávají **výstup předchozích** vrstev
- ▶ simulují **vyšší úroveň abstrakce**
- ▶ mají širší **recepční pole** (*receptive field*)



## Redukce dimenze – sdružování/pooling

**Pooling** vrstva (sdružování):

- ▶ používá se pro **zhuštění** informace – **redukci dimenze**
- ▶ **snižuje nároky** v dalších vrstvách
- ▶ podporuje **generalizaci**
- ▶ varianty:
  - **max-pooling** – vybírá maximum vstupu, značí **výskyt rysu** kdekoliv v recepčním poli
  - **average-pooling** – klasická redukce

Max Pooling

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

2 x 2  
pool size

100	184
12	45

Average Pooling

31	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

2 x 2  
pool size

36	80
12	15

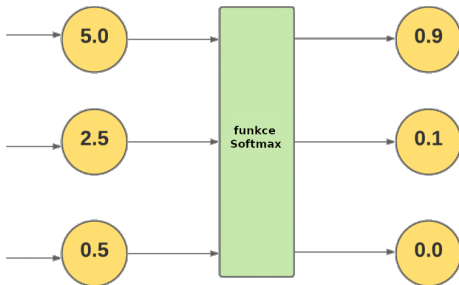
(Yani et al, 2019)

## Klasifikace výstupu – softmax

- ▶ klasifikace do  $c$  kategorií – **softmax** vrstva

$$\text{softmax}(\vec{in}) = \left\langle \frac{e^{in_k}}{\sum_{j=1}^c e^{in_j}} \right\rangle_k$$

- ▶ reprezentuje **pravděpodobnosti** (součet je **1**), akcentuje **rozdíly**
- ▶ někdy doplněná o 1–2 předcházející **plně propojené** vrstvy
- ▶ *logistická regrese (sigmoid)* – speciální případ pro 2 opačné kategorie (pozitivní a negativní)



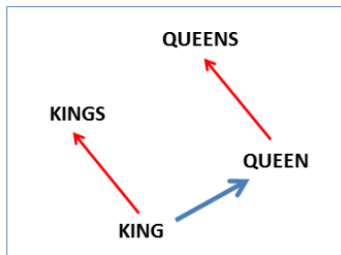
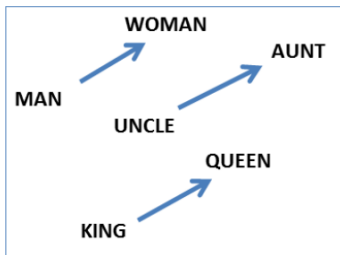
(Kumar, 2020)

## Analýza textu

neuronová síť pracuje **pouze s čísly** – jak zadat text?

slova jako **slovní vektory** (*word embeddings*):

- ▶ stanovení **pevné dimenze**
- ▶ **předpočítání/předtrénování** na velmi velkých neanotovaných textech  
⇒ **neurální jazykové modely**
- ▶ zachycení **sémantiky** – podobná slova síť zpracuje podobně
- ▶ univerzálnější – **vektory částí slov** (*subword/character embeddings*)
- ▶ jen výměnou modelu můžeme **zpřesnit výsledky**



(Mikolov, 2013)

## Predikce textu

vstup: začátek textu jako řetězec slov  $W = w_1 w_2 w_3 \dots w_{i-1}$

výstup: pravděpodobnostní distribuce dalšího slova  $P(w_i | w_1 w_2 \dots w_{i-1})$

základní **neurální jazykový model s pevným kontextem** (*fixed-window*)

výstupní distribuce

$$\vec{y} = \text{softmax}(U \cdot h + b_2) \in \mathbb{R}^{|V|}$$

skrytá vrstva

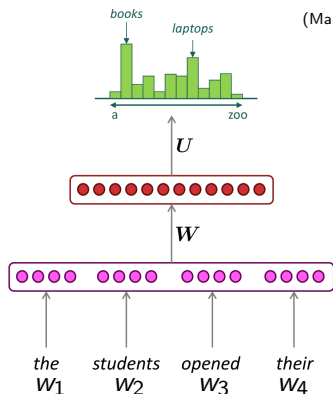
$$h = f(W \cdot e + b_1)$$

řetěžené vektory slov

$$e = [e_1; e_2; e_3; e_4]$$

slova na vstupu

$$w_1 w_2 w_3 w_4$$



(Manning, 2021)

**nevýhoda** – pevná velikost vstupu

# Rekurentní jazykový model

výstupní distribuce

$$\vec{y}^{(t)} = \text{softmax}(U \cdot h^{(t)} + b_2) \in \mathbb{R}^{|V|}$$

skryté stavy

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_e e_t + b_1)$$

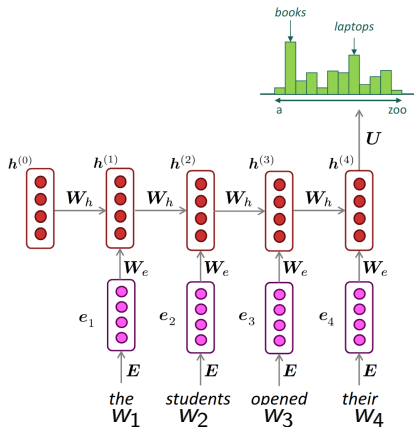
váhy  $W_h$  a  $W_e$  se aplikují opakovaně

jednotlivé vektory slov

$$e_1, e_2, e_3, e_4, \dots$$

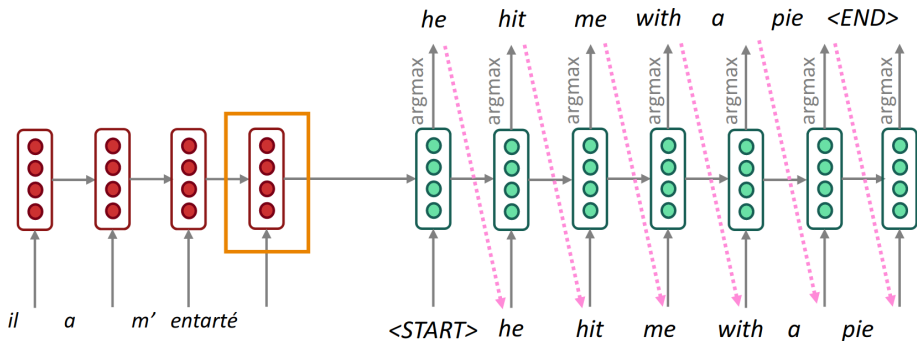
slova na vstupu

$$w_1 w_2 w_3 w_4$$



# Využití rekurentních sítí – seq2seq

častá varianta – model **sequence-to-sequence** (seq2seq)  
 dvě navazující rekurentní sítě – **enkodér** a **dekodér**

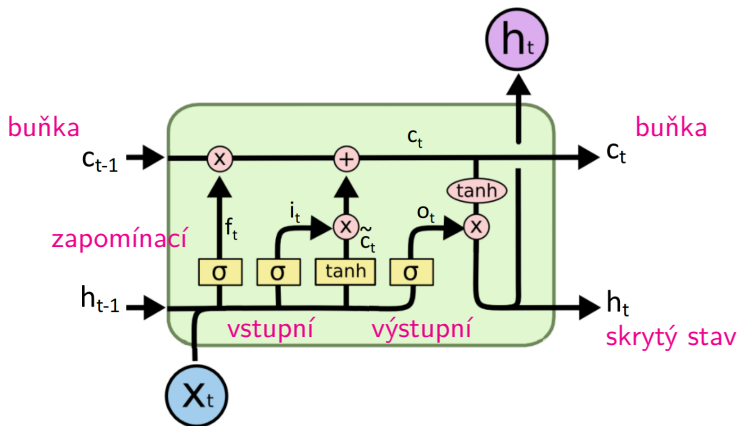


## Analýza textu

problém trénování velkých RNN – **mizející gradient** (násobení malých čísel  $\rightarrow 0$ )

řešení – **architektura Long Short-Term Memory, LSTM**:

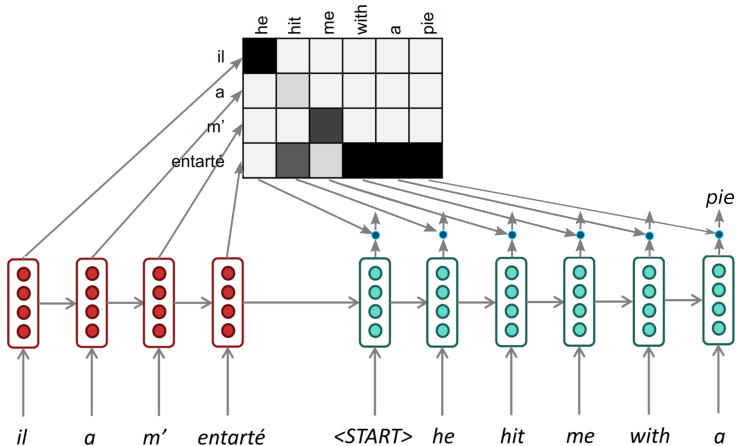
- ▶ **buňka** (*cell*)  $c_t$  – pomocná paměť
- ▶ 3 brány: **vstupní**, **výstupní** a **zapomínací** (*forget*) – regulace info do a z buňky





# Mechanismus attention

u rekurentních sítí – celá věta reprezentována jako jeden vektor  
 mechanismus **attention** (“pozornost”) – detailní provázání informací



# Architektura Transformer

“Attention is All You Need” (Vaswani et al, 2017)

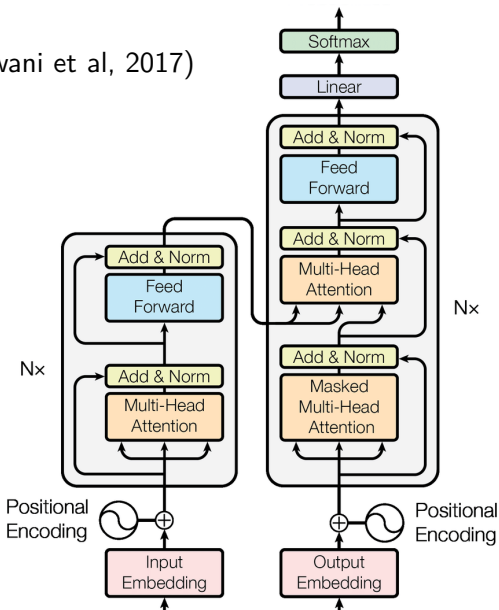
architektura **transformer**:

- ▶ vektory **pozice**
- ▶ **self-attention**
- ▶ více **hlav** (*multi-head attention*)
- ▶ **reziduální** spojení, **normalizace** a **škálování**

aktuálně **nejlepší** výsledky  
v mnoha úlohách zpracování **textu**

[beta.openai.com/examples](https://beta.openai.com/examples)

více v **další** přednášce



# Zpětnovazební učení

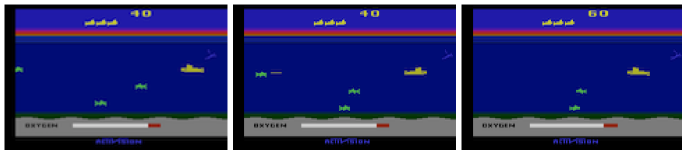
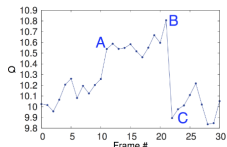
## Hluboké zpětnovazební učení (*deep reinforcement learning*, DRL)

- ▶ rozhodování agenta – ze sekvence **odměn**
- ▶ **cíl** – zvyšovat budoucí odměny
- ▶ možná **řešení** – učení funkce pro výběr akce
  - **ohodnocovací** funkce – *utility function*  $U(s)$  nebo *value function*  $V(s)$  pro stav  $s$  (AlphaGo)
  - **funkce** akce  $Q$ , **Q-learning** – *action-utility/quality function*  $Q(s, a)$  pro stav  $s$  a akci  $a$  (Atari)  
můžeme vypočítat  $U(s) = \arg \max_a Q(s, a)$
  - **politika** – *policy*  $\pi(s)$  pro stav  $s$ , která vrací přímo akci  $a$  (reflexní agent) (PPO)
- ▶ DRL zatím aplikačně obtížnější než ostatní techniky hlubokého učení

# Deep Q-Networks

Deep Q-Networks – Atari hry (2013):

- ▶ učení funkce  $Q$  přímo z obrázků, využívá konvoluční vrstvy
- ▶ odměna – herní skóre
- ▶ pozdější varianta lepší než člověk v 57 Atari hrách

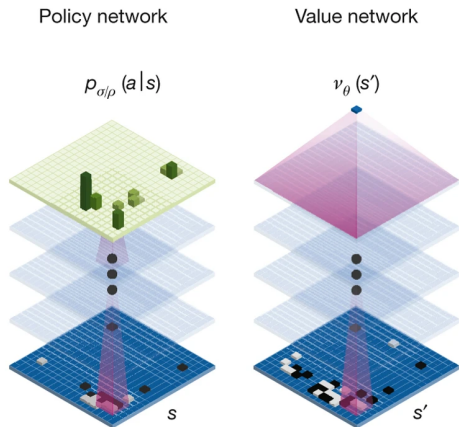


(Mnih et al, 2013)

# AlphaGo

AlphaGo (2016):

- ▶ učení funkce **politiky**  $p$  a **ohodnocovací** funkce  $v$
- ▶ Monte Carlo Tree Search
- ▶ politika  $p$  dává distribuci **pravděpodobností** možných **tahů**
- ▶ ohodnocovací funkce  $v$  predikuje **zisk** navrhovaných nových **konfigurací**  $s'$
- ▶ každá síť má 13 konvolučních vrstev



(Silver et al, 2016)

# Proximal policy optimization

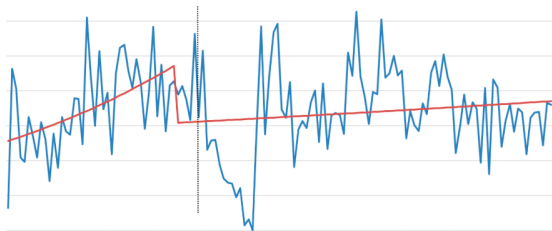
- ▶ novější učení **politiky  $\pi$**
- ▶ **jednodušší** a **stabilnější** než ostatní metody
- ▶ využívá standardních principů **gradientního sestupu**
- ▶ nová ztrátová funkce  $L^{CLIP}$  – omezuje velikost změn v zadaném pásu  $\pm\epsilon$
- ▶ využívaná pro učení **dlouhodobého** chování – pohyb robota, hraní videoher
- ▶ hlavní reinforcement technika v **OpenAI**



# Hledání architektury

## Hledání správné architektury

- ▶ převážně **experimentálně**
- ▶ **Auto ML** (*automated machine learning*)
- ▶ základní prvky:
  - **konvoluční** vrstvy – pro hledání **vzorů** (*patterns*) kdekoliv ve vstupu (1D – text, 2D – obraz, ...)  
*a **b c d** a a d c **b c d** c c a b d*
  - **rekurentní** vrstvy – hledání **závislostí** mezi prvky vstupu (text, signál, časová řada, ...)



(Adulapuram, 2020)

## Schopnosti hlubokých sítí

### co hluboké sítě umí výborně

- ▶ hledat **vzory** a **vztahy** v komplexních datech
- ▶ **generovat** nová komplexní data podle podmínek
- ▶ **detekovat**, že komplexní data byla vygenerovaná

### co hluboké sítě moc neumí

- ▶ kvantifikovat
- ▶ pracovat s **hierarchickými** strukturami (taxonomie)
- ▶ pracovat s **pravidly**

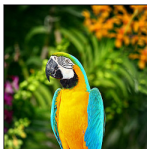
### kdy je vhodné hluboké sítě použít

máme

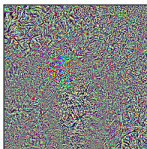
- ▶ velká data
- ▶ složité vzory

nevadí

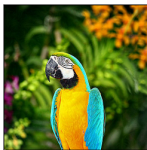
- ▶ chybí zdůvodnění
- ▶ možná zaujatost
- ▶ “nelogické” chyby



97 % papoušek



speciální šum



99 % knihovna