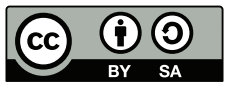
brmiversity: Umělá inteligence a teoretická informatika

Přednáška č. 9

Petr Baudiš ⟨pasky@ucw.cz⟩

brmlab 2011



brmlab hackerspace prague

Outline

brmlab hackerspace prague

Strojové učení

Učící se agent: Datový vstup a výstup, rozhodovací problém, užitková funkce

Máme trénovací množinu

- Učení s učitelem vs. bez učitele
- Rozpoznávání vs. samoorganizace

Nemáme trénovací množinu

- Exploration—exploitation dilemma
- Zpětnovazebné učení

brmlab hackerspace prague

Strojové učení nad daty

Dnes: Máme trénovací a testovací data, hledáme *hypotézu*.
Chceme řešit buď *klasifikační* nebo *regresní* úlohy.
Parametrické vs. reprezentační metody.

Strojové učení nad daty

Učení s učitelem

- Prohledávání prostoru verzí
- **Rozhodovací stromy**
- **Bayesovské učení**
- Částicové filtry
- Neuronové sítě
- **Support Vector Machines**

Učení bez učitele

- Karuhnenova-Loèveho transformace (PCA)
- **Clusterování**
- Kohonenovy mapy
- Lineární regrese

Rozhodovací stromy

- Mějme databázi $\vec{D}$ jedinců; jedinec je popsán příznakovým vektorem atributů $\vec{t}$; jedinec patří do třídy c
- Rozhodovací strom definuje posloupnost dotazů na atributy pro určení třídy
- Bankovní klienti, expertní systémy
- Konstrukce: Algoritmus ID3 maximalizující informační zisk
- *Informační entropie* $H = -\sum_a (p_a \log_2 p_a)$ (a je konkr. hodnota konkr. atributu)
- Entropie je míra překvapení v množině, chceme co největší redukci překvapení
- Výběr atributů nebo malý dataset: Pozor na přeučení

brmlab hackerspace prague

Bayesovské učení

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Nasbírané pravděpodobnosti, věrohodnost $P(B|A)$
- Predikce pomocí Bayesova pravidla
- Preference jednoduchých hypotéz
- MAP, MDL, ML (maximální věrohodnost)
- Naivní Bayesovský klasifikátor:
  Předpokládá podmíněnou nezávislost atributů
  $P(C|x_1, \ldots, x_i) = \alpha P(C)\Pi_i P(x_i|C)$

EM algoritmus

- Estimation-maximization, střídáme kroky

Support Vector Machines

- Kernelová metoda používající reprezentanty
- Lineární separace podobně jako u perceptronu
- Maximalizujeme okraj, separátor popisujeme reprezentanty
- Remapování neseparabilních množin kernelovou funkcí (mapování přes polynom či jinak)

Clusterování

- Chceme identifikovat shluky dat (bodů v n-rozměrném vektorovém prostoru)
- Hierarchické shlukování: Postupně spolu mergujeme oblaky bodů se vzrůstající maximální vzdáleností
- k-means: Předem daný počet k clusterů s danými středy; body volí nejbližší střed, středy iterativně přepočítáváme

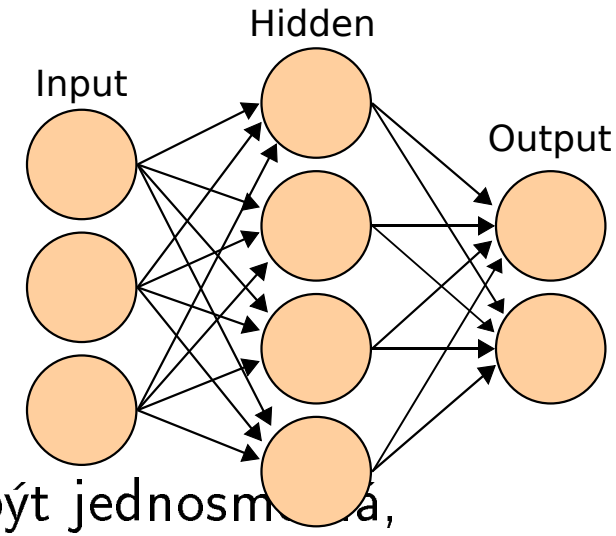
Otázky?

Příště: Strojové učení zkušeností.

Outline

Asociativní paměti

- Umělé neurony (“výpočetní krabičky”) dostávají vstupy (čísla) a na jejich základě generují výstup (číslo)
- Dnes: Chceme vstup transformovat na některý naučený vzor
- Heteroasociativní, autoasociativní a rozpoznávací sítě



- Zpětná vazba (rekurence) — síť nemusí být jednosm[illegible]á, čekáme na ustálení potenciálů ⇒ dynamický systém! Zajímavé jsou pevné body
- Skoková přenosová funkce, bipolární kódování

Hebb a BAM

- Hebbovské učení: *"What fires together, wires together."*
  $\Delta w_{ij} = \gamma x_i y_j \qquad W' = W + \vec{x}^T \vec{y}$
- TODO: obrazek
- **Bidirectional Associative Memory (BAM):** Dvouvrstvá (a)synchronní rekurentní síť s obousměrnými synapsemi, neurony každý s každým
- Váhová matice W, jedna iterace je $\vec{x} \cdot W = \vec{y}$
- Atraktory jsou vlastní vektory, nechť odpovídají vzorům
- Kapacita matice W je omezená (crosstalk); pro ortogonální vektory $m \simeq 0.18n$
- Korelační matice; neortogonální vektory: pseudoinverzní matice

brmlab hackerspace prague

Konvergence BAM

- Zkonverguje vyhodnocování BAM?

Konvergence BAM

- Zkonverguje vyhodnocování BAM? Ano! Ale proč?

Konvergence BAM

- Zkonverguje vyhodnocování BAM? Ano! Ale proč?
- Vstup do sítě by se měl blížit předchozímu vstupu
- Energetická funkce $E(\vec{x}, \vec{y}) = -\frac{1}{2}\vec{x}W\vec{y}^T$
- Lze ukázat, že v každé iteraci energetická funkce klesne!

Konvergence BAM

- Zkonverguje vyhodnocování BAM? Ano! Ale proč?
- Vstup do sítě by se měl blížit předchozímu vstupu
- Energetická funkce $E(\vec{x}, \vec{y}) = -\frac{1}{2}\vec{x}W\vec{y}^T$
- Lze ukázat, že v každé iteraci energetická funkce klesne!
- Počet hodnot energetické funkce je konečný
- QED

Hopfieldova síť

- TODO: obrazek
- Jednovrstvá rekurentní síť, neurony každý s každým
- Rozpoznávání vzorů (2D matice), optimalizační problémy
- Učení: $w_{ij} = \sum_{s=1}^{m} x_i^s y_i^s \quad i \neq j$
- Rozpoznávání: Iterace do ustálení
- Pro nulovou diagonálu konverguje (důkaz energetickou funkcí)

Optimalizační úlohy

- Hopfieldův model minimalizuje chybovou funkci
- $\Rightarrow$ je to řešítko optimalizačního problému!
- Vyjádříme-li jiný (lineární) optimalizační problém podle funkce a pak zpětně odvodíme váhy neuronů, máme optimalizovátko

Multiflop

- Na konci je aktivní právě jeden neuron
- Chybová funkce $E = \sum_i (x_i - 1)^2$
- Problém n věží, TSP

Stochastické učení

- TODO: obrazek
- Hopfieldův model na funkci E provádí **hillclimbing**
- Vždy se hýbeme po svahu co nejstrměji
- Funkce E má lokální minima, z nich se nedostaneme
- Řešení: Stochastické uhýbání ze svahu
- Simulované žíhání (annealing): $p_{\Delta E} = \frac{1}{1+e^{\Delta E/T}}$
- Aplikace na Hopfieldův model: Boltzmannův stroj

Otázky?

Příště: Samoorganizace v rámci ANN.

Outline

Rekapitulace — genetický algoritmus

- Populace řešení (s určitou strukturou), ohodnocovací funkce
- Máme populaci předchozí generace, vyrábíme novou generaci (dokud nenajdeme optimum).
- Nová generace (dokud není plná) — aplikuj genetické operátory:
  - Vyber dva jedince z minulé populace (selekce)
  - Vyrob dva nové jedince (křížení)
  - Mírně jedince uprav (mutace)
  - Dvojici vlož do nové generace

Reprezentační schémata

- Popisujme řešení jako binární řetězec
- Schéma S je slovo v abecedě 0, 1 a * (don't care)
- Schéma S reprezentuje množinu řešení
- Řád schématu $o(S)$, definující délka $d(S)$, fitness $F(S)$

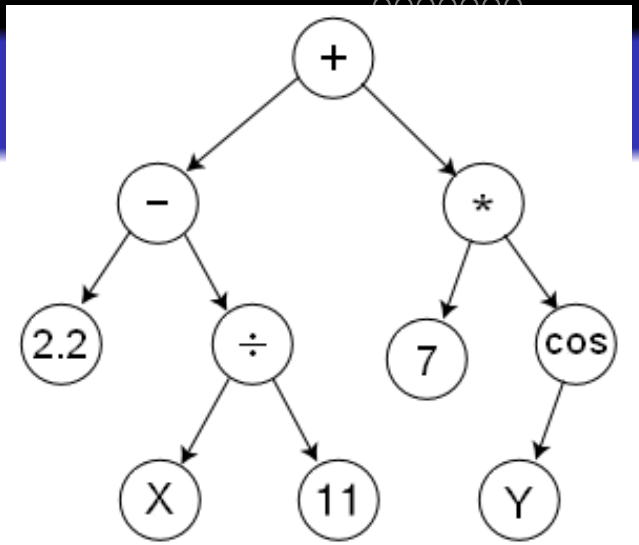
Reprezentační schémata

- Popisujme řešení jako binární řetězec
- Schéma S je slovo v abecedě 0, 1 a * (don't care)
- Schéma S reprezentuje množinu řešení
- Řád schématu $o(S)$, definující délka $d(S)$, fitness $F(S)$

- **Věta o schématech:** Schémata, která jsou krátká, nadprůměrná a malého řádu se v populaci exponenciálně množí
- Důkaz: Vliv operátorů na atributy schémat
- Tedy na kódování záleží!
- Vzpomeňte si příště: Exploration vs. exploitation problem!

brmlab hackerspace prague

Evoluční a genetické programování



$$\left(2.2 - \left(\frac{X}{11}\right)\right) + \left(7 * \cos(Y)\right)$$

- Model programu: Lispový funkcionální model nebo konečný automat
- Fitness: Přesnost výpočtu (ano/ne, odchylka, . . .)
- Mutace: Přidání/odebrání stavu nebo volání (prodlouzeni vetve programu)
- Crossover je problematický — headless chicken!
- Lineární genetické programování: Imperativní programování, trackování registrů, reprezentace pomocí bytecode
- Meta-Optimizing Semantic Evolutionary Search (MOSES)

brmlab hackerspace prague

Evoluční strategie

- Evolvujeme nikoliv pouze jedince, ale i parametry evoluce
- Jedinec je (G, S), $S = (M, L, R)$
- M počet jedinců v populaci
  L počet potomků
  R počet rodičů (gang bang)
- Křížení a mutace: Normální rozdělení, odchylky, rotace a průměr

Otázky?

Příště: Koevoluce a otevřená evoluce (brmlife).
Pravděpodobnostní model GA.

Outline

Binární vyhledávací strom

- Binární strom — hrany pouze "směrem dolů", jeden kořen, každý uzel má nula (list) až dva syny
- Vyhledávací strom — vše v levém podstromu $\leq n \leq$ vše v pravém podstromu
- Vyhledávání — cesta z kořenu k uzlu
- Vnitřní uzly — data nebo navigace?
- Chceme operace INSERT, DELETE
- Naivně: INSERT do listu, rotace u DELETE

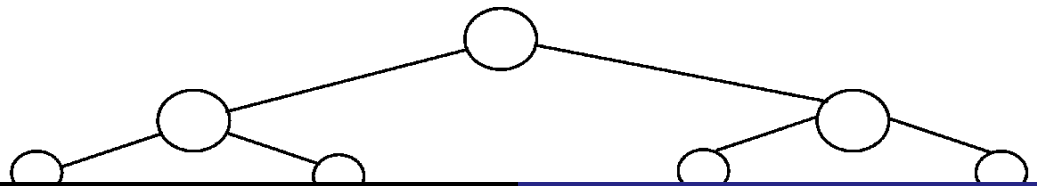
Binární vyhledávací strom

- Binární strom — hrany pouze "směrem dolů", jeden kořen, každý uzel má nula (list) až dva syny
- Vyhledávací strom — vše v levém podstromu $\leq n \leq$ vše v pravém podstromu
- Vyhledávání — cesta z kořenu k uzlu
- Vnitřní uzly — data nebo navigace?
- Chceme operace INSERT, DELETE
- Naivně: INSERT do listu, rotace u DELETE
- Jak vyrobit strom nad datasetem?

Binární vyhledávací strom

- Binární strom — hrany pouze "směrem dolů", jeden kořen, každý uzel má nula (list) až dva syny
- Vyhledávací strom — vše v levém podstromu $\leq n \leq$ vše v pravém podstromu
- Vyhledávání — cesta z kořenu k uzlu
- Vnitřní uzly — data nebo navigace?
- Chceme operace INSERT, DELETE
- Naivně: INSERT do listu, rotace u DELETE
- Jak vyrobit strom nad datasetem?

- Vyvážený strom — hloubky podstromů se liší jenom málo
- Ve vyváženém stromě trvá hledání $O(\log n)$
- Jak vyrobit a udržovat vyvážený strom?



brmlab hackerspace prague

Stromy rovnoměrně rozložených hodnot

- Hodnoty jsou rovnoměrně rozložené (třeba hashe)
- Je potřeba vůbec vyvažovat?

Splay stromy

- Zavedeme operaci *splay* — rotaci uzlu zevnitř stromu do kořene
- FIND — splay do kořene
- INSERT — do listu a splay
- DELETE — vymyslete!
- Asymptoticky vyvážený strom

Červeno-černé stromy

- Idea: některé vnitřní uzly červeně *obarvíme*; barvy opatrně střídáme
- Kořen a virtuální listy jsou vždy černé
  Oba synové červeného uzlu jsou černí
  Každá cesta z kořene do listu má stejný počet černých
- Je strom vyvážený?

Červeno-černé stromy

- Idea: některé vnitřní uzly červeně *obarvíme*; barvy opatrně střídáme
- Kořen a virtuální listy jsou vždy černé
  Oba synové červeného uzlu jsou černí
  Každá cesta z kořene do listu má stejný počet černých
- Je strom vyvážený? Nejdelší cesta je max. dvojnásobek nejkratší
- INSERT — vložíme červený skorolist a opravíme obarvení
- DELETE — vyměníme za následující hodnotu, její původní uzel utrhneme a opravíme obarvení

AVL stromy

- Idea: Každý uzel má balanční faktor; menší než -1 nebo větší než 1 je nutno opravit
- INSERT — vložíme, po cestě do kořene updatujeme a opravujeme balanční faktor
- DELETE — analogicky

Otázky?

Příště: B-stromy, haldy.

Děkuji vám

pasky@ucw.cz

Příště: Strojové učení podle zkušeností — v umělé inteligenci obecně a v rámci adaptivních agentů.
Neuronové sítě. Datové struktury.

brmlab hackerspace prague