Aplikace teorie neuronových sítí

Doc. RNDr. Iveta Mrázová, CSc.

Katedra teoretické informatiky
Matematicko-fyzikální fakulta
Univerzity Karlovy v Praze

Aplikace teorie neuronových sítí

- modely založené na samoorganizaci -

Doc. RNDr. Iveta Mrázová, CSc.

Katedra teoretické informatiky
Matematicko-fyzikální fakulta
Univerzity Karlovy v Praze

Učení bez učitele

- **Učení bez učitele:**
  - Samoorganizace a shlukování
- **Motivace:**
  - Síť sama rozhodne, která odezva je pro daný vzor nejlepší a podle toho nastaví své váhy
- **Problém:**
  - Určit počet a rozložení shluků v příznakovém prostoru

Učení bez učitele (2)

Kompetiční učení:

- Boj o „právo reprezentovat předložený vzor“
- „Potlačování soupeřů“ → **INHIBICE**
- Pravidlo „**vítěz bere vše**“
  (WTA – Winner_takes_all)

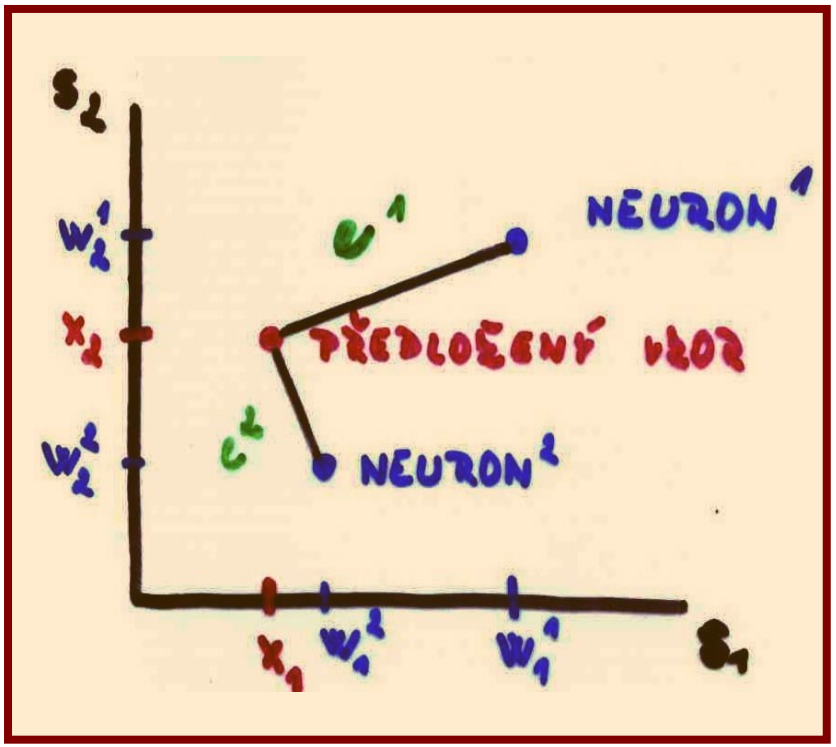
Posilované učení (reinforcement):

- Důraz na co nejlepší reprodukci vstupů

Kompetiční učení bez učitele

- ***n*** – rozměrný vstupní vzor je zpracováván pomocí takového počtu neuronů, který odpovídá (předpokládanému) počtu shluků
- Neurony v tomto případě počítají (Euklidovskou) **vzdálenost mezi předloženým vzorem a** svým **váhovým vektorem**

Kompetiční učení bez učitele (2)



- V kompetici „vítězí“ neuron, který ke nejblíže předloženému vzoru
- Vítězný neuron bude nejaktivnější a bude potlačovat – **inhibovat** – aktivitu ostatních neuronů

Kompetiční učení bez učitele (2a)

- Inhibice pomocí „laterálních spojů“

  ==> **laterální inhibice**

- Pro rozhodnutí, zda bude neuron aktivní nebo ne, je nutná globální informace o stavu všech neuronů v síti
- Aktivita neuronu signalizuje příslušnost předloženého vstupu ke shluku vektorů reprezentovaných tímto neuronem

Kompetiční učení bez učitele (3)

- Vítězný neuron zadaptuje své váhy směrem k předloženému vzoru:

$$\Delta \vec{w} = \alpha \cdot \left(\vec{x} - \vec{w} \right)$$

plasticita sítě (během učení pomalu klesá)

Cíl:

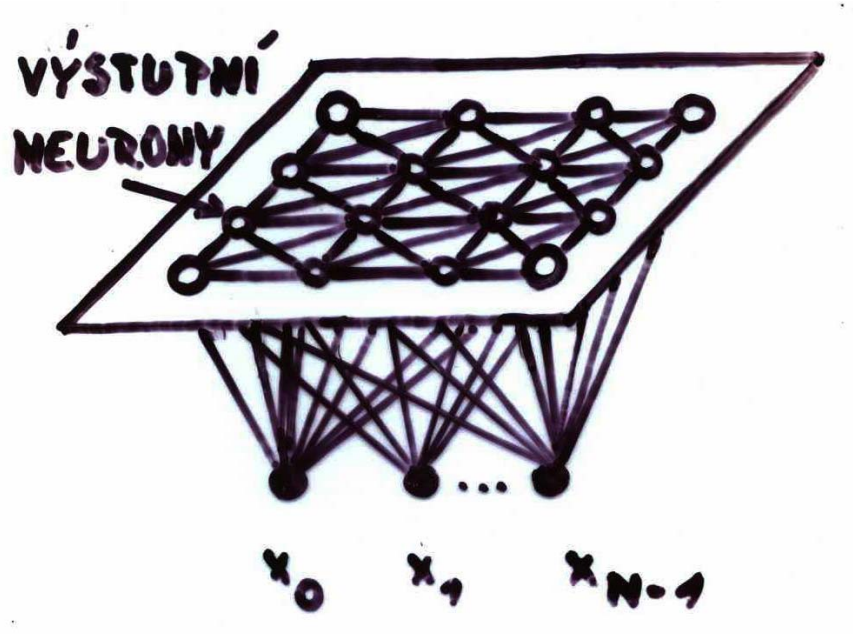
- Umístit neurony do středu shluků vzorů
- Zachovat již vytvořenou strukturu sítě

Kompetiční učení bez učitele (4)

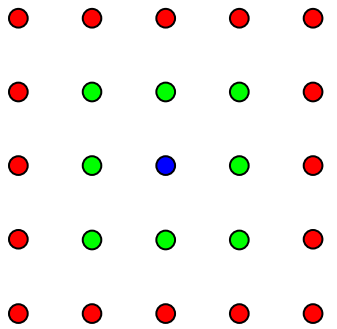
- **Urychlení procesu učení:**
  - Vhodná inicializace vah
  - Např. podle náhodně vybraných vzorů
- **Problémy:**
  - Mrtvé (nevyužité) neurony
    - Mřížka v Kohonenově vrstvě
    - Topologické okolí neuronu
    - Řízená kompetice a mechanismus svědomí

Kohonenovy mapy

- Teuvo Kohonen – fonetický psací stroj



Topologické okolí



$NE_j(O)$

$NE_j(t_1)$

Kohonenův model – algoritmus učení

Motivace:

- Mřížka, na níž jsou uspořádané neurony, umožňuje identifikaci nejbližších sousedů daného neuronu

  → v průběhu učení se aktualizují váhy příslušných neuronů i jejich sousedů

  Cíl: sousední neurony by měly také reagovat na velmi podobné signály

Kohonenův model – algoritmus učení (5)

Definice okolí:

- V jednorozměrné Kohonenově mapě patří do okolí neuronu k s poloměrem 1 neurony $k-1$ a $k+1$
- Neurony na obou koncích jednorozměrné Kohonenovy mapy mají asymetrické okolí
- V 1 – rozměrné Kohonenově mapě patří do okolí neuronu k o poloměru r všechny neurony, které jsou od k vzdáleny až o r pozic doleva či doprava
- Obdobně pro vícerozměrné Kohonenovy mapy a zvolenou metriku na mřížce (čtvercová, hexagonální, …)

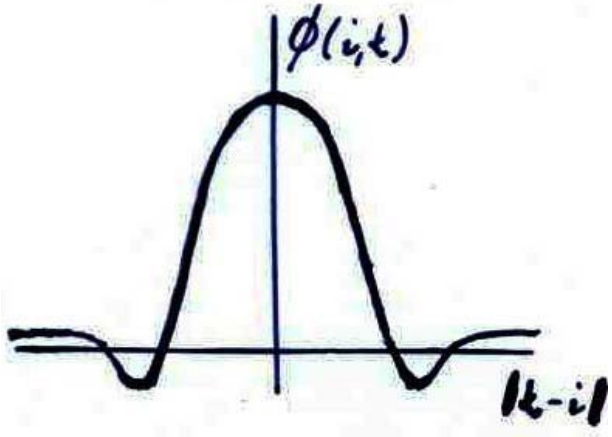
Kohonenův model – algoritmus učení (6)

Funkce laterální interakce *Φ(i,k)*:

~ „síla laterální vazby“ mezi neurony ***i*** a ***k*** během učení

Příklad:

- ***Φ(i,k)=1*** $\forall$ ***i*** z okolí ***k*** s poloměrem ***r*** a ***Φ(i,k)=0*** $\forall$ ostatní ***i***
- Funkce „mexického klobouku“
- … a další …



Kohonenovy samoorganizující se příznakové mapy: algoritmus

Krok 1: Zvol hodnoty vah mezi N vstupmíni a M výstupními neurony jako malé náhodné hodnoty. Zvol počáteční poloměr okolí a funkci laterální interakce Φ.

Krok 2: Předlož nový trénovací vzor.

Krok 3: Spočítej vzdálenosti d_j mezi vstupním a váhovým vektorem pro každý výstupní neuron j pomocí:

$$d_j = \sum_{i=0}^{N-1} \left(x_i(t) - w_{ij}(t) \right)^2$$

Kde $x_i(t)$ je vstupem neuronu i v čase t a $w_{ij}(t)$ je váhou synapse ze vstupního neuronu i k výstupnímu neuronu j v čase t. Tuto vzdálenost lze upravit váhovým koeficientem a předat kompetiční vrstvě.

Kohonenovy samoorganizující se příznakové mapy: algoritmus (2)

Krok 4: Vyber (např. pomocí laterální interakce) takový výstupní neuron c, který má minimální d_j a označ ho jako „vítěze“.

Krok 5: Váhy se aktualizují pro neuron c a všechny neurony v okolí definovaném pomocí N_c. Nové váhy jsou:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t)\, \Phi(c,j)\, (x_i(t) - w_{ij}(t))$$

Pro $j \in N_c$; $0 \le i \le N-1$

$\alpha(t)$ je vigilanční koeficient ($0 < \alpha(t) < 1$) , který klesá v čase (vigilance ~ bdělost) .

Kohonenovy samoorganizující se příznakové mapy: algoritmus (3)

Pro volbu $\alpha(t)$ by mělo platit:

$$\sum_{t=1}^{\infty} \alpha(t) = \infty \quad \wedge \quad \sum_{t=1}^{\infty} \alpha^2(t) < \infty$$

Při procesu učení tak vítězný neuron upraví svůj váhový vektor směrem k aktuálnímu vstupnímu vektoru.
Totéž platí pro neurony v okolí „vítěze“.
Hodnota funkce $\Phi(c,j)$ klesá s rostoucí vzdáleností neuronů od středu okolí N_c.

Krok 6: Přejdi ke Kroku 2.

Kompetiční AVQ-algoritmy
(Adaptive Vector Quantization)

Krok 1: inicializujte vektory synaptických vah:

$$\vec{m}_i(0) = \vec{x}_i \; ; \quad i = 1, \ldots, p$$

- inicializace pomocí trénovacích vzorů omezuje výskyt patologických jevů při učení založeném na algoritmu NNR

Krok 2: pro náhodný trénovací vzor $\vec{x}(t)$ najdi nejbližší „vítězný“ vektor synaptických vah $\vec{m}_j(t)$

$$\left\| \vec{m}_j(t) - \vec{x}(t) \right\| = \min_i \left\| \vec{m}_i(t) - \vec{x}(t) \right\|$$

kde $\left\| \vec{x} \right\|^2 = x_1^2 + \ldots + x_n^2$

Krok 3: aktualizujte vektor „vítězných“ synaptických vah $\vec{m}_j(t)$ pomocí algoritmů UCL, SCL1 anebo SCL2

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (2)

Kompetiční učení bez učitele – UCL:

~ unsupervised competitive learning

$$\vec{m}_j(t+1) = \vec{m}_j(t) + c_t\left[\vec{x}(t) - \vec{m}_j(t)\right]$$

$$\vec{m}_i(t+1) = \vec{m}_i(t) \qquad \text{jestliže} \qquad i \neq j$$

Parametr učení:

$\{c_t\}$ … je pomalu klesající posloupnost parametrů učení, např. $c_t = 0.1\,(1 - t/2000)$, resp. $c_t = 1/t$

Konvergence k lokálnímu minimu:

$$\sum_{t=1}^{\infty} c_t = \infty \qquad \sum_{t=1}^{\infty} c_t^2 < \infty$$

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (3)

Kompetiční učení s učitelem 1 – SCL1:

~ supervised competitive learning 1

$$\vec{m}_j(t+1) = \begin{cases} \vec{m}_j(t) + c_t\left[\vec{x}(t) - \vec{m}_j(t)\right] & \text{jestliže} \quad \vec{x}(t) \in D_j \\ \vec{m}_j(t) - c_t\left[\vec{x}(t) - \vec{m}_j(t)\right] & \text{jestliže} \quad \vec{x}(t) \notin D_j \end{cases}$$

$$\vec{m}_i(t+1) = \vec{m}_i(t) \quad \text{jestliže} \quad i \neq j$$

Jiný zápis: $\vec{m}_j(t+1) = \vec{m}_j(t) + c_t\, r_j \left[\vec{x}(t) - \vec{m}_j(t)\right]$

Posilovací funkce pro učení s učitelem: $r_j = I_{D_j} - \sum_{i \neq j} I_{D_i}$

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (4)

Kompetiční učení s učitelem 2 – SCL2:

~ supervised competitive learning 2

$$\begin{cases} \vec{m}_j(t+1) = \vec{m}_j(t) - c_t\left[\vec{x}(t) - \vec{m}_j(t)\right] \\ \vec{m}_l(t+1) = \vec{m}_l(t) + c_t\left[\vec{x}(t) - \vec{m}_l(t)\right] \end{cases}$$

jestliže $\vec{x} \in D_l$ namísto $\vec{x} \in D_j$ a jestliže je $\vec{m}_j(t)$ nejbližší vektor synaptických vah a $\vec{m}_l(t)$ je další nejbližší vektor synaptických vah:

$$\left\| \vec{m}_j(t) - \vec{x}(t) \right\| < \left\| \vec{m}_l(t) - \vec{x}(t) \right\| = \min_{i \neq j} \left\| \vec{m}_i(t) - \vec{x}(t) \right\|$$

$$\vec{m}_i(t+1) = \vec{m}_i(t) \quad \text{pro všechny ostatní případy}$$

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (5)

Diferenciální kompetiční učení – DCL:

~ differential competitive learning

- **Kombinace diferenciálního Hebbovského a kompetičního učení:**

$$m'_{ij} = S'_j(y_j)\left[S_i(x_i) - m_{ij}\right]$$

$$\left(\vec{m}'_j = S'_j(y_j)\left[\vec{S}(\vec{x}) - \vec{m}_j\right]\right)$$

kde $\vec{S}(\vec{x}) = (S_1(x_1), \ldots, S_n(x_n))$ a $\vec{m}_j = (m_{1j}, \ldots, m_{nj})$

m_{ij} označuje váhu synapse mezi i – tým neuronem vstupní vrstvy F_X a j – tým neuronem kompetiční vrstvy F_Y

S_i, S_j jsou nezáporné přenosové funkce

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (6)

Diferenciální kompetiční učení – DCL (pokračování):

m'_{ij} a $S'_j(y_j)$ označují derivace m_{ij} a $S_j(y_j)$ v čase,

obvykle: $$S_j(y_j) = \frac{1}{1 + e^{-cy_j}}$$

pro nějakou konstantu $c > 0$

- j – tý neuron „vítězí“ v kompetici, jestliže $S_j = 1$, a prohrává, jestliže $S_j = 0$

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (7)

Diferenciální kompetiční učení – DCL (pokračování):

→ $\vec{m}_j$ **se adaptuje pouze v případě, že se hodnota** $S_j(y_j)$ **mění**

→ **rozdíl proti klasickému kompetičnímu učení:**

$$m'_{ij} = S_j(y_j)\,[\,S_i(x_i) - m_{ij}\,]$$

→ $\vec{m}_j$ **se adaptuje v případě, že** *j*-**tý neuron „vyhrál“ kompetici**

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (8)

Diferenciální kompetiční učení – DCL:

Krok 1: Inicializace: $\vec{m}_i(0) = \vec{x}(i)$

Krok 2: Najdi „vítězný“ $\vec{m}_j(t)$:

$$\left\| \vec{m}_j(t) - \vec{x}(t) \right\| = \min_i \left\| \vec{m}_i(t) - \vec{x}(t) \right\|$$

Krok 3: Aktualizace „vítězného“ $\vec{m}_j(t)$:

$$\vec{m}'_j(t+1) = \vec{m}'_j(t) + c_t \Delta S_j(y_j(t)) \left[S(\vec{x}(t)) - \vec{m}_j(t) \right]$$

jestliže ***j***-tý neuron „vítězí“

$\vec{m}_i(t+1) = \vec{m}_i(t)$ jestliže ***i***–tý neuron „prohrává“

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (9)

Diferenciální kompetiční učení – DCL (pokračování):

- $\Delta S_j(y_j(t))$ představuje změnu kompetičního signálu $S_j(y_j)$ j–tého neuronu z kompetiční vrstvy:

$$\Delta S_j(y_j(t)) = sgn\,[\,S_j(y_j(t+1)) - S_j(y_j(t))\,]$$

- Operátor signum definujeme následovně:

$$\operatorname{sgn}(x) = \begin{cases} 1 & \text{jestliže } x > 0 \\ 0 & \text{jestliže } x = 0 \\ -1 & \text{jestliže } x < 0 \end{cases}$$

- Aditivní aktualizace y_j ve vrstvě F_Y:

$$y_j(t+1) = y_j(t) + \sum_i^n S_i(x_i(t))\, m_{ij}(t) + \sum_k^p S_k(y_k(t))\, w_{kj}$$

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (10)

Diferenciální kompetiční učení – DCL (zjednodušení):

- $S_i(x_i(t)) = x_i \quad \rightarrow \quad \sum_i^n x_i m_{ij}(t)$
- Jeden „vítěz“ v každé iteraci $\rightarrow$ $y_k w_{kj}$, k značí „vítězný“ neuron
  - Matice W $(p \times p)$:

$$W = \begin{bmatrix} +2 & -1 & \dots & -1 \\ -1 & +2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ -1 & \dots & -1 & +2 \end{bmatrix}$$

- Každý neuron v F_Y kóduje jistou třídu vzorů

$\rightarrow \quad S(\vec{x}) \cdot \vec{m}_j = \|S(\vec{x})\| \, \|\vec{m}_j\| \cos\left(S(\vec{x}), \vec{m}_j\right)$

Pozitivní učení ($m'_{ij} > 0$) nastává v případě, že systém klasifikuje do nejbližší třídy D_j

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (11)

Diferenciální kompetiční učení – DCL (zjednodušení):

- Lineární data dávají často velké hodnoty pro $\left| \sum_{i}^{n} x_i \, m_{ij}(t) \right|$: to způsobuje saturaci signálů S_j

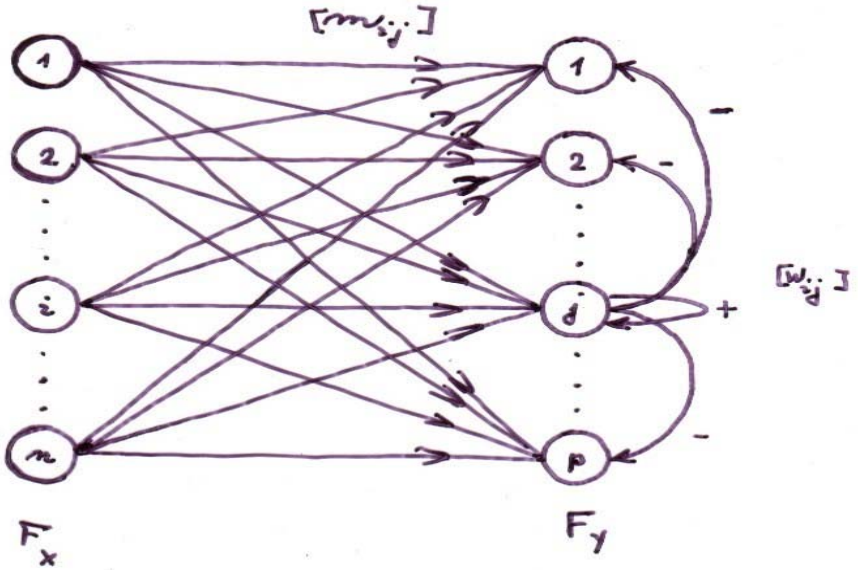
→ $\Delta S_j = 0$ ~ při kompetici nejsou patrné žádné změny

X Δy_j ~ zůstává citlivost vůči změnám při kompetici)

Kompetiční AVQ-algoritmy (Adaptive Vector Quantization) (12)

Diferenciální kompetiční učení – DCL (zjednodušení):

Topologie sítě DCL:



- Učení bez učitele
- Konvergence (rychlejší než SCL)
- Procházení centroidů

Segmentace hloubkových map

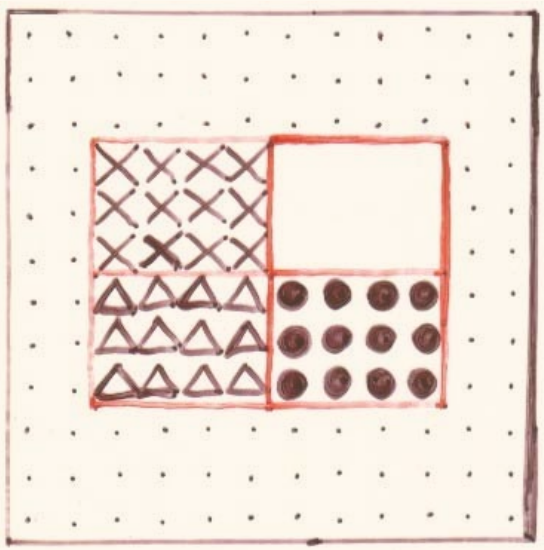
Povrchový model:

- reprezentuje povrch objektu pomocí grafu, který vyjadřuje sousednost ploch na povrchu objektu
- Povrch jako graf funkce:

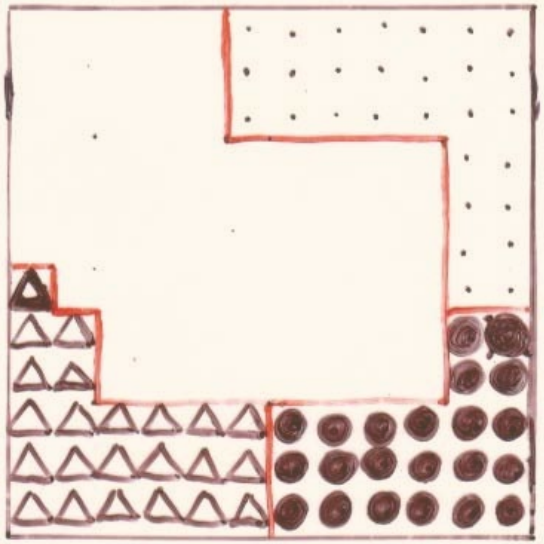
$$S = \{ (x, y, z) ; z = f(x, y) \in D \subset \mathcal{R} \}$$

- **Segmentace** ~ rozdělení pixlů hloubkové mapy do shluků, které reprezentují hladné oblasti povrchu ohraničené nespojitými konturami povrchu

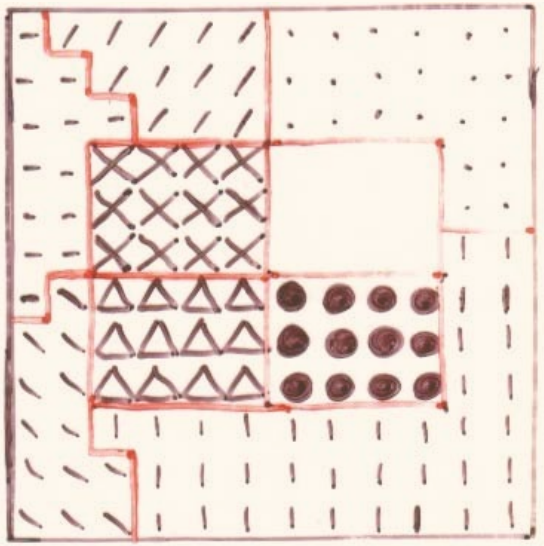
Segmentace hloubkových map (2)



Původní obrázek



„Undersegmentation“



„Oversegmentation“

Segmentace hloubkových map (3)

Hranový přístup:

- Detekce pixlů mapy představujících nespojitosti plochy a jejich klasifikace jako skokové, ostré nebo oblé hrany
- Detekované pixly hran jsou pak navzájem pospojovány do hran
- x Fragmentace hran, které je pak nutné spojovat pomocí různých heuristik
- x Vyhlazovací operátory pro redukci šumu rozmývají hrany do více pixlů → obtížnější lokalizace hran

Segmentace hloubkových map (4)

Plochový přístup:

- Odhad křivosti plochy v každém pixlu
- Klastrování pixlů mapy do hladkých oblastí s homogenní křivostí
- Techniky narůstání oblastí
- Techniky klastrování příznakových vektorů

Segmentace hloubkových map (5)

Techniky narůstání oblastí:

- Každý pixl je klasifikován do jednoho z předem definovaných typů podle své křivosti
- Počáteční klasifikace představuje základ následného narůstání oblastí
- Spojování a dělení sousedních oblastí je citlivé na volbu vhodného kritéria
  - Návrh dobrého kritéria založeného na odhadu senzorového šumu a zakřivení plochy je problematický vzhledem ke štěpení a spojování sousedních oblastí

Segmentace hloubkových map (6)

Techniky narůstání oblastí (pokračování):

- Možnost špatné segmentace v případě ostrých nebo oblých hran

  → „undersegmentation“, resp. „oversegmentation“ sousedních ploch:

  - **Undersegmentation** (~ podsegmentování)
    - splynutí vícero oblastí do jedné
  - **Oversegmentation** (~ přesegmentování)

Segmentace hloubkových map (7)

Techniky klastrování příznakových vektorů:

- Každému pixlu je přiřazen odpovídající příznakový vektor
- Vektorová kvantizace příznakových vektorů
  - Rozčlenění ***n*** –rozměrných příznakových vektorů do ***m*** oblastí za současné minimalizace chybové funkce
  - Všechny body každé plochy by přitom měly být aproximovány reprezentativním vektorem $\vec{x}_i$ přiřazeným příslušné ploše
- x Počet oblastí ***m*** by měl být znám a-priori
- x Není zaručena prostorová spojitost (ve smyslu obrazových souřadnic)

Segmentace hloubkových map (8)

Volba příznakového vektoru:

- Homogenita segmentovaných oblastí
- Vlastnosti příznaků:
  - Obecnost ~ použitelnost v širokém spektru aplikací
  - Úplná charakteristika povrchu
  - Poskytnutí úplného popisu pro zpracování na vyšších úrovních
  - Lokální podpora ~ výpočet příznaků se provádí v lokálním okénku uvažovaného pixlu
  - Stabilita a invariance – změna měřítka apod.
  - Robustnost, snadná detekce, spolehlivost

Segmentace hloubkových map (9)

Volba příznakového vektoru (pokračování):

Homogenní oblasti hloubkových map mohou být ohraničeny třemi typy hran:

- Skokové hrany:
  - Diskontinuita hloubkových hodnot
    ~ jeden objekt zakrývá druhý anebo sám sebe
    → informaci o hloubce zahrnout do příznakového vektoru
- Ostré hrany:
  - Diskontinuita normálových vektorů
    → normálový vektor také zahrnout do příznakového vektoru

Segmentace hloubkových map (10)

Volba příznakového vektoru (pokračování):

- Oblé hrany:
  - Diskontinuita křivosti plochy
    → křivosti plochy by měly být rovněž zahrnuty do příznakového vektoru
- Zároveň by měly být do příznakového vektoru zahrnuty i ***x***–ové a ***y***–ové souřadnice
  → zachování prostorové spojitosti oblastí v segmentovaném obraze

Segmentace hloubkových map (11)

Volba příznakového vektoru (pokračování):

→ příznakový vektor klastrovacího algoritmu je zvolen jako ***8*** – rozměrný vektor:

$$\vec{x} = (x, y, z; n_x, n_y, n_z; H; K)$$

(n_x, n_y, n_z) … jednotkový normálový vektor v bodě (x, y, z) na 3-rozměrné ploše

H a K … střední a Gaussovská křivost

Předpoklad: $x = u \quad y = v \quad z = f(x, y) = f(u, v)$

Segmentace hloubkových map (12)

Volba příznakového vektoru (pokračování):

$\Rightarrow$ pro $\vec{x}(u,v) = (u, v, f(u,v))$ platí :

$$\vec{x}_u(u,v) = (1,0, f_u(u,v))$$

$$\vec{x}_v(u,v) = (0,1, f_v(u,v))$$

$$\vec{x}_{uu}(u,v) = (0,0, f_{uu}(u,v))$$

$$\vec{x}_{vv}(u,v) = (0,0, f_{vv}(u,v))$$

$$\vec{x}_{uv}(u,v) = \vec{x}_{vu}(u,v) = (0,0, f_{uv}(u,v))$$

Segmentace hloubkových map (13)

Volba příznakového vektoru (pokračování):

$$\vec{n}(u,v) = \frac{\vec{x}_u(u,v) \times \vec{x}_v(u,v)}{\left\| \vec{x}_u(u,v) \times \vec{x}_v(u,v) \right\|} = \frac{(-f_u, -f_v, 1)}{\sqrt{1 + f_u^2 + f_v^2}}$$

kde : $\vec{x}_u(u,v) = \{ [\partial \vec{x}(u,v)] / \partial u \}$

$\vec{x}_v(u,v) = \{ [\partial \vec{x}(u,v)] / \partial v \}$

$\vec{x}_{uu}(u,v) = \{ [\partial^2 \vec{x}(u,v)] / \partial u^2 \}$

$\vec{x}_{uv}(u,v) = \{ [\partial^2 \vec{x}(u,v)] / \partial u \partial v \}$

$\vec{x}_{vv}(u,v) = \{ [\partial^2 \vec{x}(u,v)] / \partial v^2 \}$

Segmentace hloubkových map (14)

Volba příznakového vektoru (pokračování):

a $\vec{n}(u,v)$ je normálový vektor plochy v bodě ***(u , v)***

Dále je třeba určit ***G*** a ***B***:

$$G: \quad g_{11} = E = \vec{x}_u \cdot \vec{x}_u = 1 + f_u^{\,2}$$

$$g_{12} = g_{21} = F = \vec{x}_u \cdot \vec{x}_v = f_u \cdot f_v$$

$$g_{22} = G = \vec{x}_v \cdot \vec{x}_v = 1 + f_v^{\,2}$$

Segmentace hloubkových map (15)

Volba příznakového vektoru (pokračování):

$$B: \quad b_{11} = L = \vec{x}_{uu} \cdot \vec{n} = \frac{f_{uu}}{\sqrt{1 + f_u^2 + f_v^2}}$$

$$b_{12} = b_{21} = M = \vec{x}_{uv} \cdot \vec{n} = \frac{f_{uv}}{\sqrt{1 + f_u^2 + f_v^2}}$$

$$b_{22} = N = \vec{x}_{vv} \cdot \vec{n} = \frac{f_{vv}}{\sqrt{1 + f_u^2 + f_v^2}}$$

→ **střední křivost:** $H = \frac{1}{2}\left[trace \left(G^{-1} B \right) \right]$

a **Gaussovská křivost:** $K = \| B \| / \| G \|$

Segmentace hloubkových map (16)

Volba příznakového vektoru (pokračování):

- výpočet normálového vektoru a křivostí pomocí okénkových operátorů

Výpočet normálového vektoru:

- Jednotkový normálový vektor v bodě $\vec{p}(u,v) = (u, v, f(u,v))$

  ~ vektorový součin prvních derivací podle ***u*** a ***v***:

$$\vec{n}_{\vec{p}} = \frac{\frac{\partial \vec{p}}{\partial u} \times \frac{\partial \vec{p}}{\partial v}}{\left\| \frac{\partial \vec{p}}{\partial u} \times \frac{\partial \vec{p}}{\partial v} \right\|}$$

Segmentace hloubkových map (17)

Výpočet normálového vektoru (pokračování):

- Odhad prvních derivací pomocí operátorů $\boldsymbol{D_u}$ a $\boldsymbol{D_v}$:

$$D_u = \vec{d}_0 \cdot \vec{d}_1^{\,T} \qquad \text{a} \qquad D_v = \vec{d}_1 \cdot \vec{d}_0^{\,T}$$

$$\text{kde} \quad \vec{d}_0 = \frac{1}{5}\left[1, 1, 1, 1, 1\right]^T$$

$$\vec{d}_1 = \frac{1}{5}\left[-2, -1, 0, 1, 2\right]^T$$

Segmentace hloubkových map (18)

Výpočet křivostí plochy (pokračování):

- Odhad křivosti plochy v bodě $\vec{p}$ ve směru pixlu $\vec{q}$ podle:

$$k(\vec{p},\vec{q}) = \frac{\left\|\vec{n}_{\vec{p}} - \vec{n}_{\vec{q}}\right\|}{\left\|\vec{p} - \vec{q}\right\|} \times s(\vec{p},\vec{q})$$

kde $s(\vec{p},\vec{q}) = 1$, jestliže $\left\|\vec{p} - \vec{q}\right\| \leq \left\|\vec{n}_{\vec{p}} - \vec{n}_{\vec{q}}\right\|$

$s(\vec{p},\vec{q}) = -1$, jinak

$\vec{n}_{\vec{p}}$ a $\vec{n}_{\vec{q}}$ … jednotkové normálové vektory v bodě $\vec{p}$ resp. $\vec{q}$

$s(\vec{p},\vec{q})$ … určuje konvexnost / konkávnost

Segmentace hloubkových map (19)

Výpočet křivostí plochy (pokračování):

→ Výpočet střední a Gaussovské křivosti pomocí $k(\vec{p},\vec{q})$

→ nechť $\Omega(\vec{p})$ je množina pixlů v okolí pixlu $\vec{p}$:

→ střední křivost: $$H = \frac{\left(k^{\min}(\vec{p}) + k^{\max}(\vec{p})\right)}{2}$$

→ Gaussovská křivost: $$K = \left(k^{\min}(\vec{p}) \times k^{\max}(\vec{p})\right)$$

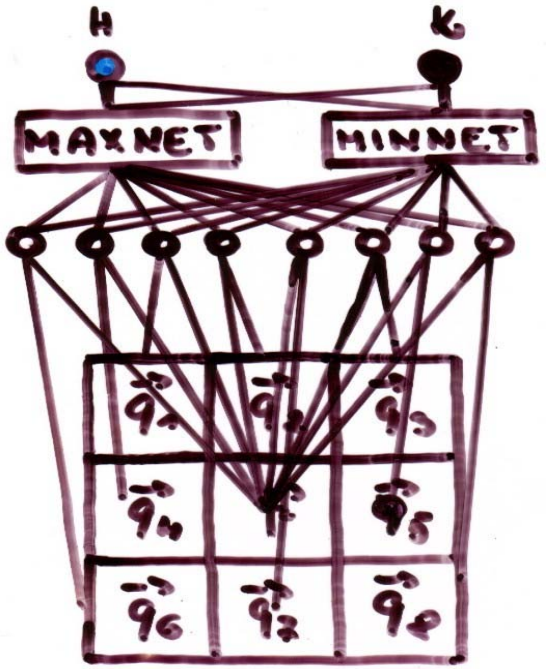
kde $$k^{\min}(\vec{p}) = k(\vec{p},\vec{q}_0) = \min_{q \in \Omega(\vec{p})} k(\vec{p},\vec{q})$$

$$k^{\max}(\vec{p}) = k(\vec{p},\vec{q}_1) = \max_{q \in \Omega(\vec{p})} k(\vec{p},\vec{q})$$

Segmentace hloubkových map (20)

Výpočet křivostí plochy (pokračování):

- Realizace výpočtu pomocí neuronové sítě

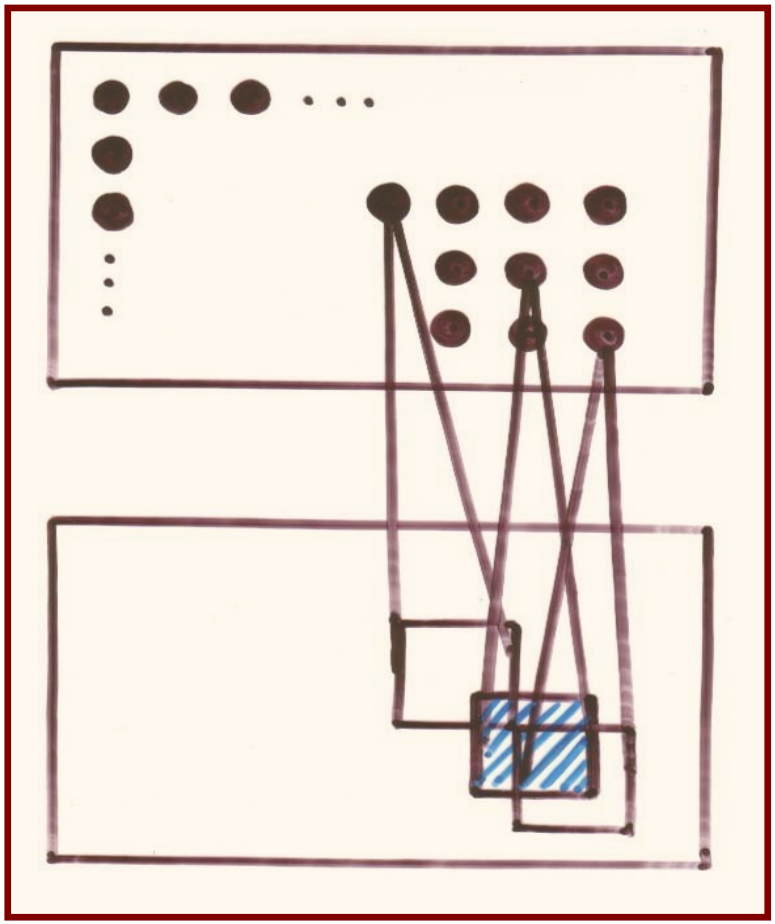


neuron pro násobení

Neuron pro průměrování

Neuron pro výpočet $k(\vec{p}, \vec{q}_i)$

Segmentace hloubkových map (21)



Využití neuronové sítě pomocí navzájem se překrývajících receptorových polí

Segmentace hloubkových map (22)

Využití Kohonenových map k vektorové kvantizaci příznakových vektorů

$$\vec{x} = \left(x , y , z ; n_x , n_y , n_z ; H ; K \right)$$

- Nechť $R_1, R_2, \ldots, R_M$ označuje M oblastí po segmentaci
- Kohonenova mapa „se musí naučit“ zobrazení $\vec{x}(x, y) \mapsto R_i$, které by minimalizovalo předem dané kritérium
- Váhové vektory: $\vec{w} = \left(w_x, w_y, w_z, w_{nx}, w_{ny}, w_{nz}, w_H, w_K \right)$
- **Problém:** počet neuronů

Segmentace hloubkových map (23)

Samoorganizující se příznakové mapy

- Kompetiční učení
- Vektorová kvantizace

Váhové vektory charakterizují jednotlivé shluky vstupních vektorů

→ středy shluků by měly odpovídat hustotě pravděpodobnosti vstupních dat

V ideálním případě by měl být po skončení učení každý neuron kompetiční vrstvy citlivý na jedinou oblast

Segmentace hloubkových map (24)

Samoorganizující se příznakové mapy (pokračování):

Váhový vektor $\vec{w}_j = (w_x, w_y, w_z, w_{nx}, w_{ny}, w_{nz}, w_H, w_K)$ reprezentuje příslušnou oblast

- (w_x, w_y, w_z) … souřadnice centra oblasti ve *3*–rozměrném prostoru
- (w_{xx}, w_{ny}, w_{nz}) … reprezentativní normálový vektor oblasti
- w_H, w_K … reprezentativní střední a Gaussovská křivost (~ nabývají zhruba průměrných hodnot křivostí jednotlivých pixlů v dané oblasti)

Segmentace hloubkových map (25)

Samoorganizující se příznakové mapy (pokračování):

- Počet neuronů by měl odpovídat počtu požadovaných oblastí v segmentovaném obrázku!
- x Velká závislost na aktuálním obsahu scény, který není apriori znám

Struktura (první) kompetiční vrstvy vícevrstvé Kohonenovy mapy

- $X_1 = \left\{ \vec{x}_i \; ; \; i = 1, 2, \ldots, N_f^2 \right\}$ … vstupní prostor pro 1. vrstvu
- $W_1 = \left\{ \vec{w}_j = \left(w_x, w_y, w_z, w_{nx}, w_{ny}, w_{nz}, w_H, w_K \right) ; j = 1, \ldots, N^2 \right\}$

  po skončení učení obsahuje reprezentativní vektory odpovídající první úrovni abstrakce vstupních vektorů

Segmentace hloubkových map (26)

Struktura (první) kompetiční vrstvy vícevrstvé Kohonenovy mapy

- $O_1 \subseteq W_1$ … množina váhových vektorů odpovídajících vítězným neuronům
  - Tvoří množinu vstupních vektorů pro druhou vrstvu ($X_2 = O_1$)
  - Ostatní neurony se adaptují tak, aby interpolovaly hodnoty vítězných neuronů
- Vážená euklidovská metrika:

$$d\left(\vec{x}_i, \vec{w}_j\right) = a \cdot \left\|\vec{x}_p - \vec{w}_p\right\| + b \cdot \left\|\vec{x}_n - \vec{w}_n\right\| + c \cdot \left|H - w_H\right| + d \cdot \left|K - w_K\right|$$

kde $0 \le a, b, c, d \le 1$ a $\vec{x}_i = \left(\vec{x}_p; \vec{x}_n; H; K\right)$; $\vec{x}_p = \left(x, y, z\right)$

$\vec{x}_n = \left(n_x, n_y, n_z\right)$ a analogicky pro $\vec{w}_j$

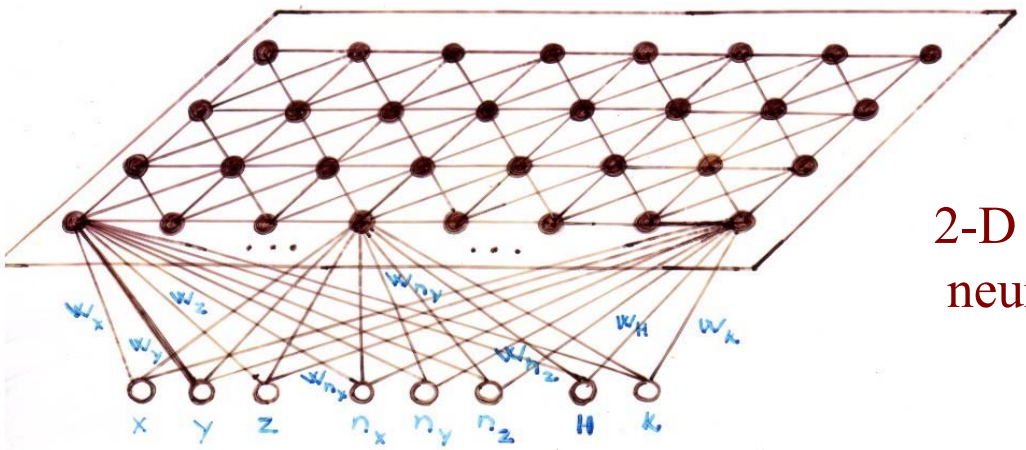
Segmentace hloubkových map (27)

Struktura (první) kompetiční vrstvy vícevrstvé Kohonenovy mapy

$$\vec{x}_2 = \vec{o}_1$$

dávková jednotka

$$\vec{w}_j = (w_x, w_y, w_z, w_{nx}, w_{ny}, w_{nz}, w_H, w_K) \in W_1$$

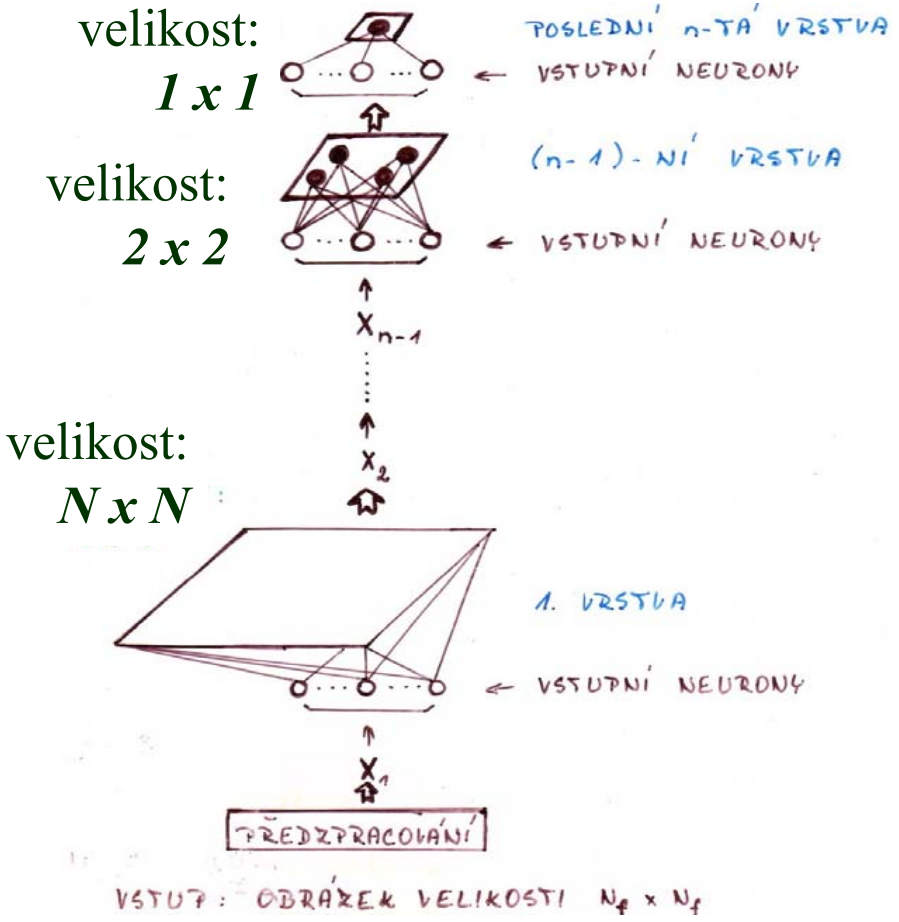


2-D pole neuronů

$$\vec{x}_i = (x, y, z; n_x, n_y, n_z; H; K) \in X_1$$

Segmentace hloubkových map (27)

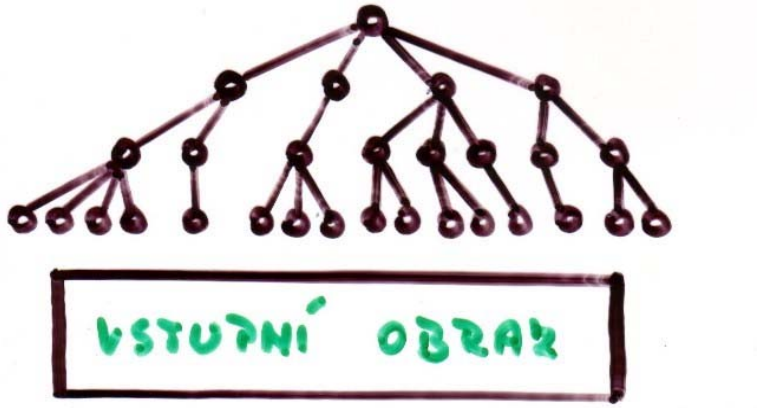
- Vícevrstvé samo-organizující se příznakové mapy:
  (~ multi-layer self-organizing feature maps)
- Kontrola počtu segmentovaných oblastí, resp. neuronů



Segmentace hloubkových map (28)

Kontrola počtu segmentovaných oblastí

1 oblast
4 oblasti
8 oblastí
16 oblastí



Abstrakční strom

Uzavřený uzel: ●
Segmentované oblasti:
$\{ R_1 , R_2 , \ldots , R_7 \}$

4. úroveň
3. úroveň
2. úroveň
1. úroveň