Cvičenia

Cvičenie 6.1. Aké sú základné postuláty holografického modelu pamäti?

Riešenie. Základné postuláty holografického prístupu k pamäti sú tieto:

1. Pamäť je v mozgu realizovaná distribuovaným spôsobom (je experimentálne zistené, že jednotlivé položky pamäti nemajú lokalistický charakter, lézie mozgu, Lashleyho experimenty na potkanoch). Distribuovaná reprezentácia poznatkov (vedomostí, koncepcií,...) používa také kódovaniu, ktoré je založené na neurónových sieťach - neurónové aktivity sú interpretované ako „neurálalne koreláty“ daných poznatkov.
2. Kognitívna aktivita spočíva v časovom slede distribuovaných reprezentácií, pričom existujú pravidlá, ako meniť vybranú reprezentáciu na inú reprezentáciu.
3. Tento prístup poskytuje umožnuje zaviesť metaforu čenia so schopnosťou generalizovať (indukcia).

Cvičenie 6.2. Ako je definovaný konceptuálny vektor a operácia konvolúcie pre tieto vektory?

Riešenie: Základný pojem holografického modelu pamäti je *konceptuálny vektor*, ktorý je reprezentovaný *n*-rozmerným vektorom

$$\boldsymbol{a} \in R^n \Rightarrow \boldsymbol{a} = \left(a_0, a_2, ..., a_{n-1}\right)$$

pričom jeho komponenty sú náhodné čísla so štandardným normálnym rozdelením

$$a_i = N\left(0, 1/n\right) \quad \forall \in \left\{0, 1, ..., n-1\right\}$$

kde $N(0,1/n)$ je náhodné číslo so stredom v 0 a so strednou odchýlkou $1/n$. Nad konceptuálnymi vektormi je definovaná binárna operácia „*konvolúciu*“, ktorá dvojici vektorov priradí tretí vektor, $\otimes \;:\; R^n \times R^n \to R^n$, alebo

$$\boldsymbol{c} = \boldsymbol{a} \otimes \boldsymbol{b}$$

Zložky výsledného vektora $c = \left(c_0, c_1, ..., c_{n-1}\right)$ sú určené vzťahom

$$c_i = \sum_{j=0}^{n-1} a_j b_{[i-j]} \quad \left(i = 0, 1, ..., n-1\right)$$

kde index v hranatej zátvorke, [*k*], je špecifikovaný pomocou operácie *modulo n* takto

$$k' = k \; mod \; n$$

$$\left[k\right] = \begin{cases} k' & \left(if \; k' \geq 0\right) \\ n + k' & \left(if \; k' < 0\right) \end{cases}$$

Konvolúcia dvoch vektorov $\boldsymbol{a}$ a $\boldsymbol{b}$ pre n=3 má tento tvar

$$c_0 = a_0 b_0 + a_1 b_2 + a_2 b_1$$

$$c_1 = a_0 b_1 + a_1 b_0 + a_2 b_2$$

$$c_2 = a_0 b_2 + a_1 b_1 + a_2 b_0$$

Konvolúcia vyhovuje týmto vlastnostiam:

(1) komutatívnosť, $\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{b} \otimes \boldsymbol{a}$

(2) asociatívnosť, $\left(\boldsymbol{a} \otimes \boldsymbol{b}\right) \otimes \boldsymbol{c} = \boldsymbol{a} \otimes \left(\boldsymbol{b} \otimes \boldsymbol{c}\right)$

(3) distributívnosť, $\boldsymbol{a} \otimes \left(\alpha \boldsymbol{b} + \beta \boldsymbol{c}\right) = \alpha\left(\boldsymbol{a} \otimes \boldsymbol{b}\right) + \beta\left(\boldsymbol{a} \otimes \boldsymbol{c}\right)$

(4) existencia jednotkového vektora, $\mathbf{1} \otimes \boldsymbol{a} = \boldsymbol{a} \quad \left(\mathbf{1} = \left(1, 0, ..., 0\right)\right)$

Časová zložitosť operácie konvolúcie je $o\left(n^2\right)$

Cvičenie 6.3. Ako je definovaná operácia *involúcie* a aké sú jej základné vlastnosti?

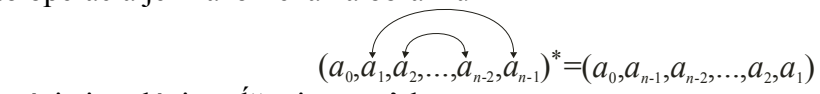
Riešenie: Operácia involúcie

$$\left(\ \right)^* \ : \ R^n \to R^n$$

Je definovaná vzťahom

$$b = a^* = \left(a_{[0]}, a_{[-1]}, \ldots, a_{[-n+2]}, a_{[-n+1]}\right)$$

Táto operácia je znázornená na obrázku



Operácia involúcie spĺňa tieto vzťahy

$$\left(\boldsymbol{a}+\boldsymbol{b}\right)^* = \boldsymbol{a}^* + \boldsymbol{b}^*$$

$$\left(\boldsymbol{a}\otimes\boldsymbol{b}\right)^* = \boldsymbol{a}^* \otimes \boldsymbol{b}^*$$

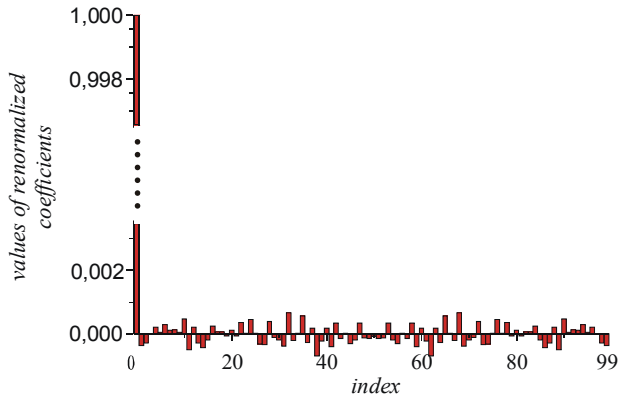
$$\left(\boldsymbol{a}\otimes\boldsymbol{b}^*\right)\cdot\boldsymbol{c} = \boldsymbol{a}\cdot\left(\boldsymbol{b}\otimes\boldsymbol{c}\right)$$

$$\boldsymbol{a}^{**} = \boldsymbol{a}$$

Dôležitá vlastnosť involúcie ***c**** spočíva v tom, že sa približne rovná „inverznému“ vektoru, $\boldsymbol{c}^* \otimes \boldsymbol{c} \approx \mathbf{1}$

$$\begin{aligned}\left(\boldsymbol{c}^* \otimes \boldsymbol{c}\right)_i &= \sum_{k=0}^{n-1} c_k^* c_{[i-k]} = \sum_{k=0}^{n-1} c_{[-k]} c_{[i-k]} \\ &= \begin{cases} \boldsymbol{c}\cdot\boldsymbol{c} & \left(\text{pre } i=0\right) \\ \sum_{k=0}^{n-1} c_{[-k]} c_{[i-k]} & \left(\text{pre } i>0\right) \end{cases} \\ &= \frac{1}{\boldsymbol{c}\cdot\boldsymbol{c}} \begin{cases} 1 & \left(\text{pre } i=0\right) \\ \left(\boldsymbol{c}\cdot\boldsymbol{c}\right)\sum_{k=0}^{n-1} c_{[-k]} c_{[i-k]} & \left(\text{pre } i>0\right) \end{cases} \\ &\approx \begin{cases} 1 & \left(\text{pre } i=0\right) \\ \varepsilon_i & \left(\text{pre } i>0\right) \end{cases}\end{aligned}$$

kde ε_i sú náhodné malé čísla. Nultá zložka konvolúcie $\left(\boldsymbol{c}^* \otimes \boldsymbol{c}\right)_0$ sa rovná skalárnemu súčinu $\boldsymbol{c}\cdot\boldsymbol{c}$, ktorý je vyjadrený pomocou kladných “diagonálnych“ členov c_i^2, zatiaľ čo ostatné zložky $\left(\boldsymbol{c}^* \otimes \boldsymbol{c}\right)_i$, pre $i \geq 1$, sú určené sumami „nediagonálnych“ členov $c_i c_j$, ktoré majú náhodné znamienko. Dôsledkom tejto skutočnosti je, že $\left(\boldsymbol{c}^* \otimes \boldsymbol{c}\right)_0$ je podstatne väčšia než ako $\left(\boldsymbol{c}^* \otimes \boldsymbol{c}\right)_i$, pre $i \geq 1$, z čoho priamo vyplýva požadovaná skutočnosť $\boldsymbol{c}^* \otimes \boldsymbol{c} \approx \mathbf{1}$, pozri obrázok.



Tento obrázok znázorňuje histogram jednotlivých zložiek súčinu $c^* \otimes c$, kde c je náhodne generovaný conceptuálny vektor pre n=100. Absolútna hodnota „prvej“ zložky $(c^* \otimes c)_0$ je väčšia o niekoľko rádov ako absolútne hodnoty ostatných zložiek $(c^* \otimes c)_i$, pre $i \geq 1$. To znamená, že súčin $c^* \otimes c$ po vhodnej normalizácii je približne rovný jednotkovému vektoru $(c \cdot c)^{-1}(c^* \otimes c) \square \mathbf{1} = (1,0,...,0,0)$.

Príklad 6.4. Aký má význam operácia involúcie?

Riešenie. Pomocou operácie involúcie môžeme zrekonštruovať pôvodné konceptuálne vektory, ktoré boli použité pri konštrukcii konvolúcie dvoch vektorov. Táto možnosť je veľmi dôležitá, pretože umožňuje dekódovať pôvodnú informáciu zo zložených konceptuálnych vektorov. *Rekonštrukcia* $\boldsymbol{x}$ z $\boldsymbol{c}\otimes\boldsymbol{x}$ je založená na skutočnosti, že že involúcia vektora je približne rovná jeho inverznej hodnote, t.j. $\boldsymbol{c}^* \otimes \boldsymbol{c} \approx \mathbf{1}$ (čo bolo ukázane v predchádzajúcom príklade)

$$\tilde{\boldsymbol{x}} = \boldsymbol{c}^* \otimes (\boldsymbol{c} \otimes \boldsymbol{x}) = (\boldsymbol{c}^* \otimes \boldsymbol{c}) \otimes \boldsymbol{x} \approx \frac{1}{\boldsymbol{c} \cdot \boldsymbol{c}} \mathbf{1} \otimes \boldsymbol{x} = \frac{1}{\boldsymbol{c} \cdot \boldsymbol{c}} \boldsymbol{x} \tag{19}$$

podľa ktorej, konvolúcia $\boldsymbol{c}^*$ s vektorom $\boldsymbol{c}\otimes\boldsymbol{x}$ produkuje vektor $\tilde{\boldsymbol{x}}$, ktorý je podobný pôvodnému vektoru $\boldsymbol{x}$, $\tilde{\boldsymbol{x}} \approx \boldsymbol{x}$. Tento výsledok vyjadríme takto

$$\frac{1}{(\boldsymbol{c} \cdot \boldsymbol{c})} \tilde{\boldsymbol{x}} = \begin{pmatrix} x_0 \\ x_1 + \eta_1 \\ ..... \\ x_{n-1} + \eta_{n-1} \end{pmatrix} = \boldsymbol{x} + \boldsymbol{\eta} \tag{20}$$

kde vektor $\boldsymbol{\eta}$ je interpretovaný ako náhodný šum s normálnou distribúciou so strtednou hodnotou v nule a so štandardnou odchýlkou omnoho menšou než akú má $\boldsymbol{x}$.

Prekryv rezultujúceho vektora $\tilde{\boldsymbol{x}}$ s pôvodným vektorom $\boldsymbol{x}$ je určený pomocou skalárneho súčinu takto

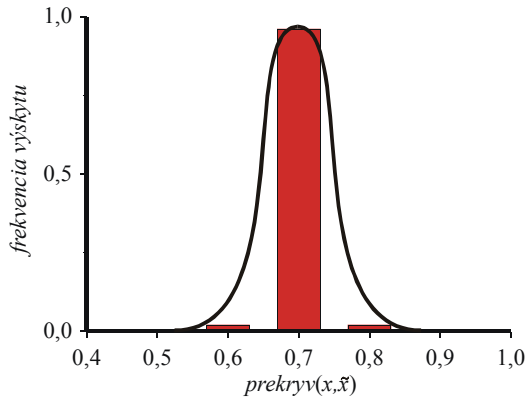
$$-1 \leq prekryv(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{\boldsymbol{x} \cdot \tilde{\boldsymbol{x}}}{|\boldsymbol{x}||\tilde{\boldsymbol{x}}|} \leq 1 \tag{21}$$

kde nerovnosti vyplývajú priamo zo Schwartzovej nerovnosti z lineárnej algebry. Čím je táto hodnota bližšie k maximálnej hodnote 1, tým sú si vektory $\tilde{\boldsymbol{x}}$ a $\boldsymbol{x}$ *podobnejšie*[1].

Na obrázku je znázornený histogram prekryvov pre súčin $\boldsymbol{c}\otimes\boldsymbol{x}$, obsahujúci dvojicu náhodne vygenerovaných rôznych konceptuálnych vektorov $\boldsymbol{c}$ a $\boldsymbol{x}$ dimenzie n=1000.

[1] V prípade, že prekryv sa blíži k hodnote -1, potom vektory $\tilde{\boldsymbol{x}}$ a $\boldsymbol{x}$ sú si taktiež podobné, aj keď sú opačne orientované (antikolineárne).

Z obrázku vyplýva, že najčastejší prekryv medzi $\tilde{\boldsymbol{x}} = \boldsymbol{c}^* \otimes \boldsymbol{c} \otimes \boldsymbol{x}$ a $\boldsymbol{x}$ je okolo 0.7, z čoho vyplýva, že vektory $\tilde{\boldsymbol{x}}$ a $\boldsymbol{x}$ sú si podobné, $\tilde{\boldsymbol{x}} \approx \boldsymbol{x}$.



Príklad 6.5. Ako je špecifikovaná aditívna pamäť?

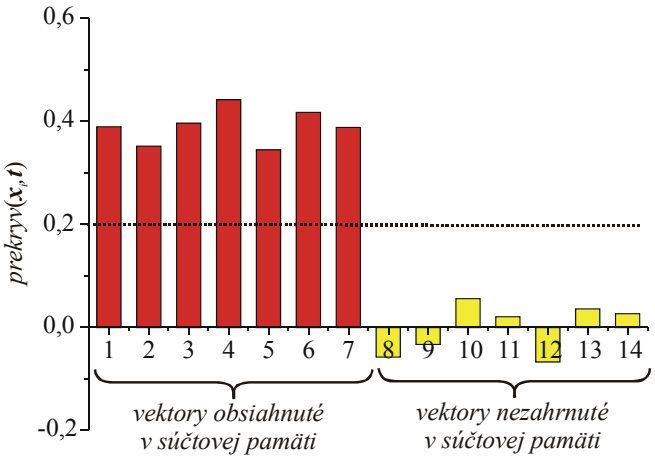
Riešenie. Majme množinu obsahujúcu $p+q$ náhodne vygenerovaných konceptuálnych vektorov, $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_p, \boldsymbol{x}_{p+1}, ..., \boldsymbol{x}_{p+q}\}$, pričom $p<q$. Pomocou prvých p vektorov z X definujme *pamäťový vektor* $\boldsymbol{t}$ ako ich súčet

$$\boldsymbol{t} = \sum_{i=1}^{p} \boldsymbol{x}_i \tag{22}$$

Vektor $\boldsymbol{t}$ reprezentuje ***aditívnu pamäť'***, ktorá jednoduchým aditívnym spôsobom obsahuje vektory z množiny X. Rozhodnutie o tom, či nejaký vektor $\boldsymbol{x} \in X$ je obsiahnutý v $\boldsymbol{t}$ musí byť založené na hodnote prekryvu (21)

$$prekryv(\boldsymbol{x}, \boldsymbol{t}) = \frac{\boldsymbol{x} \cdot \boldsymbol{t}}{|\boldsymbol{x}||\boldsymbol{t}|} \tag{23}$$

Ak je táto hodnota väčšia ako vopred zvolená prahová hodnota, $prekryv(\boldsymbol{x}, \boldsymbol{t}) \geq \vartheta$, potom vektor $\boldsymbol{x}$ je zahrnutý v aditívnej pamäti $\boldsymbol{t}$, v opačnom prípade, ak $prekryv(\boldsymbol{x}, \boldsymbol{t}) < \vartheta$, potom vektor $\boldsymbol{x}$ nie je zahrnutý v $\boldsymbol{t}$, pozri obrázok



Znázornenie súčtovej pamäti pre prvých 7 vektorov množiny X, ktorá obsahuje 14 náhodne vygenerovaných konceptuálnych vektorov dimenzie n=1000. Prahovú hodnotu ϑ v tomto prípade môžeme zvoliť 0.2.

Výpočty ukazujú, že ak zmenšujeme počet vektorov v aditívnej pamäti, potom prahová hodnota rastie k jednotkovej hodnore, v opačnom prípade, ak zväčšujeme počet vektorov v aditívnej pamäti, potom práová hodnota prudko klesa k hodnotám blízkym nule. To znamená, že v prípade aditívnej pamäti, maximálny počet konceptuálnych vektorov, ktoré je možne „zapamätať si“ je okolo 7.

Príklad 6.6. Čo je to „čistenie“ (angl. *clean-up*)?

Riešenie. Je to postupný proces rekonštrukcie z aditívneho pamäťového vektora $\boldsymbol{t}$ jeho pôvodné zložky – konceptuálne vektory. Majme množinu vektorov $X=\{\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_n\}$ a nejaký vektor $\boldsymbol{t}$. Stojíme pre problémom, rozhodnúť, či pamäťový vektor $\boldsymbol{t}$ obsahuje aditívnu komponentu, ktorá je podobná (alebo nie je podobná) nejakému vektoru z množiny X. Tento problém riešime pomocou výpočtu prekryvu

$$\boldsymbol{x}\approx\boldsymbol{t}=\begin{cases}áno & \left(prekryv\left(\boldsymbol{x},\boldsymbol{t}\right)\geq\vartheta\right)\\ nie & \left(prekryv\left(\boldsymbol{x},\boldsymbol{t}\right)<\vartheta\right)\end{cases} \tag{24}$$

kde ϑ je zvolená *prahová hodnota* akceptovania veľkosti prekryvu ako pozitívnej odpovedi. Výsledkom tohto procesu čistenia je podmnožina vektorov

$$X\left(\boldsymbol{t}\right)=\left\{\boldsymbol{x}\in X;\boldsymbol{x}\approx\boldsymbol{t}\right\}\subseteq X \tag{25}$$

Môžeme si položiť otázku aj v trochu inej podobe, a to, či pamäťový vektor $\boldsymbol{t}$ je podobný nejakému vektoru z množiny X? Odpoveď na túto všeobecnejšiu otázku rozhodneme podľa maximálnej hodnoty prekryvu

$$prekryv\left(\boldsymbol{t},X\right)=\max_{x\in X}prekryv\left(\boldsymbol{t},\boldsymbol{x}\right) \tag{26}$$

Potom môžeme prepísať (24) do tvaru

$$\boldsymbol{x}\approx X=\begin{cases}áno & \left(prekryv\left(\boldsymbol{x},X\right)\geq\vartheta\right)\\ nie & \left(prekryv\left(\boldsymbol{x},X\right)<\vartheta\right)\end{cases} \tag{27}$$

Príklad 6.7. Ako je špecifikovaná asočiačná pamäť?

Riešenie. Konštrukcia asociačnej pamäti patrí medzi hlavné výsledky holografickej redukovanej reprezentácie. Majme množinu konceptuálnych vektorov $X=\{\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_n\}$ a tréningovú množinu $A_{train}=\{\boldsymbol{c}_i/\boldsymbol{x}_i;i=1,2,...,m\}$, ktorá obsahuje $m<n$ asociačných dvojíc konceptuálnych vektorov $\boldsymbol{c}_i/\boldsymbol{x}_i$, kde c_i je *narážka* do asociačnej pamäti (angl. *cue*) a x_i je výstup z pamäti. Zostrojíme pamäťový vektor t, ktorý reprezentuje *asociačnú pamäť* nad tréningovou množinou A_{train}

$$\boldsymbol{t}=\boldsymbol{c}_1\otimes\boldsymbol{x}_1+...+\boldsymbol{c}_m\otimes\boldsymbol{x}_m=\sum_{i=1}^{m}\boldsymbol{c}_i\otimes\boldsymbol{x}_i \tag{28}$$

Predpokladajme, že vopred poznáme len narážky asociačnej pamäti $\boldsymbol{c}_i$, nepoznáme možné výstupy z množiny $X_{train}=\{\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_m\}$. Odozva asociačnej pamäti na vstup – narážku $\boldsymbol{c}_i$ je určená pomocou procesu „čistenia“. V prvom kroku spočítame vektor $\tilde{\boldsymbol{x}}_i=\boldsymbol{c}_i^*\otimes\boldsymbol{t}$, potom pomocou procesu založenom na maximálnej hodnoty prekryvu zistíme či $\tilde{\boldsymbol{x}}_i\approx\boldsymbol{x}_i\in X$

$$prekryv\left(\tilde{\boldsymbol{x}}_i,X\right)=\max_{x\in X_{train}}prekryv\left(\tilde{\boldsymbol{x}}_i,\boldsymbol{x}\right) \tag{29}$$

Pre ilustráciu tejto pamäti majme tréningovú množinu $A_{train} = \{\boldsymbol{c}_i / \boldsymbol{x}_i \,;\, i = 1,2,...,m\}$, ktorá je náhodne vygenererovaná pre m=8, pričom dimenzia konceptuálnych vektorov je n=1000. Pre každú asociačnú dvojicu $\boldsymbol{c}_i / \boldsymbol{x}_i$ spočítame $\boldsymbol{t}_i = \boldsymbol{c}_i \otimes \boldsymbol{x}_i$. Hodnoty $prekryv\left(\boldsymbol{c}_i^* \otimes \boldsymbol{t}_i, \boldsymbol{x}_j\right)$ sú uvedené v tabuľke

	1	2	3	4	5	6	7	8
1	0.71703	-0.01820	0.01452	0.02776	-0.01488	-0.01922	-0.02442	0.01358
2	-0.03998	0.73804	0.01510	0.01430	0.00276	0.02346	-0.00545	-0.01626
3	-0.02757	-0.01736	0.64667	0.00474	-0.11580	-0.00812	0.01476	0.00379
4	0.00785	0.00374	-0.01899	0.68728	-0.15340	0.00005	-0.00561	0.00136
5	-0.00466	0.00426	-0.01831	-0.00827	0.70767	0.04175	-0.03384	-0.00668
6	-0.01467	0.02522	-0.01403	-0.01316	-0.03000	0.71444	0.00078	-0.00526
7	0.02966	0.00892	-0.00301	-0.00358	0.01285	0.00971	0.70790	0.01816
8	-0.00344	-0.01080	0.00843	-0.01871	0.00324	-0.02629	0.00851	0.58957

Z tejto tabuľky jasne vyplýva, že prekryvy sú dostatočne veľké len pre diagonálne hodnoty, zatiaľ čo nediagonálne prekryvy sú rádovo menšie. Môžeme teda jednoznačne rozhodnúť pomocou prekryvu, či $\boldsymbol{c}_i^* \otimes \boldsymbol{t}_i \approx \boldsymbol{x}_i$ je asociantom s „narážkou“ $\mathbf{c}_i$.

Príklad 6.8. Ako je modelovaný pamäťový vektor pre postupnosť znakov?

Riešenie. Ukážeme, že daný prístup je schopný spracovať aj lineárny reťazec znakov

$$postupnosť = \{\boldsymbol{a} \to \boldsymbol{b} \to \boldsymbol{c} \to \boldsymbol{d} \to \boldsymbol{e} \to \boldsymbol{f}\}$$

Pre tieto vektory zostrojíme pamäťový vektor

$$\boldsymbol{t}_0 = \boldsymbol{a} + \boldsymbol{a} \otimes \boldsymbol{b} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e} \otimes \boldsymbol{f}$$

Vieme, že tento vektor obsahuje postupnosť vektorov zakódovanú vektorom, avšak nevieme ktorých vektorov a v akom poradí. Pomocou procedúry „čistenia“ z vektora $\boldsymbol{t}_0$ môžeme postupne rekonštruovať pôvodnú postupnosť nasledujúcou procedúrou (pozri obr. 7):

1. krok: $\boldsymbol{a} = clean\_up\left(\boldsymbol{t}_0\right)$, $\boldsymbol{t}_1 := \boldsymbol{t}_0 - \boldsymbol{a}$,

   $\tilde{\boldsymbol{t}}_1 := \boldsymbol{a}^* \otimes \boldsymbol{t}_1$,

2. krok: $\boldsymbol{b} = clean\_up\left(\tilde{\boldsymbol{t}}_1\right)$, $\boldsymbol{t}_2 := \boldsymbol{t}_1 - \boldsymbol{a} \otimes \boldsymbol{b}$,

   $\tilde{\boldsymbol{t}}_2 := \left(\boldsymbol{a} \otimes \boldsymbol{b}\right)^* \otimes \boldsymbol{t}_2$,

3. krok: $\boldsymbol{c} = clean\_up\left(\tilde{\boldsymbol{t}}_2\right)$, $\boldsymbol{t}_3 := \boldsymbol{t}_2 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c}$,

   $\tilde{\boldsymbol{t}}_3 := \left(\boldsymbol{y}_1 \otimes \boldsymbol{y}_2 \otimes \boldsymbol{y}_3\right)^* \otimes \boldsymbol{t}_3$,

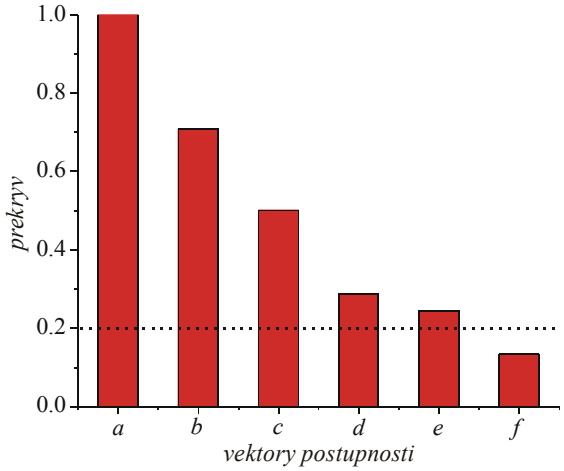
4. krok: $\boldsymbol{d} = clean\_up\left(\tilde{\boldsymbol{t}}_3\right)$, $\boldsymbol{t}_4 := \boldsymbol{t}_3 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d}$,

   $\tilde{\boldsymbol{t}}_4 := \left(\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d}\right)^* \otimes \boldsymbol{t}_4$,

5. krok: $\boldsymbol{e} = clean\_up\left(\tilde{\boldsymbol{t}}_4\right)$, $\boldsymbol{t}_5 := \boldsymbol{t}_4 - \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e}$,

   $\tilde{\boldsymbol{t}}_5 := \left(\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \otimes \boldsymbol{d} \otimes \boldsymbol{e}\right)^* \otimes \boldsymbol{t}_5$,

6. krok: $\boldsymbol{f} = clean\_up\left(\tilde{\boldsymbol{t}}_5\right)$.



Prekryv pre jednotlivé vektory s postupnosti z (30), ktoré boli získané rekonštrukciou z vektora $\boldsymbol{t}_0$. Z obrázku vyplýva dôležitý poznatok, že nastáva pomerne rýchla degradácia rekonštrukcie, už šiesty vektor f je rekonštruovaný s pravdepodobnosťou menšou ako 0.20.

Postupnosť symbolov môže byť kódovaná taktiež pomocou asociačnej pamäti, kde vektor vstupu $\boldsymbol{c}_i$ špecifikuje i-tú pozíciu daného symbola. Vyššie uvedený ilustračný príklad je reprezentovaný pamäťovým vektorom

$$\boldsymbol{t} = \boldsymbol{c}_1 \otimes \boldsymbol{a} + \boldsymbol{c}_2 \otimes \boldsymbol{b} + \boldsymbol{c}_3 \otimes \boldsymbol{c} + \boldsymbol{c}_4 \otimes \boldsymbol{d} + \boldsymbol{c}_5 \otimes \boldsymbol{e} + \boldsymbol{c}_6 \otimes \boldsymbol{f} \tag{34}$$

Potom rekognoskácia tejto postupnosti spočíva v hľadaní asocianta k vstupnému vektoru $\boldsymbol{c}_i$, pomocou procesu čistenia zostrojíme „tréningovú množinu“

$$A_{train} = \{\boldsymbol{c}_1/\boldsymbol{a}, \boldsymbol{c}_2/\boldsymbol{b}, \boldsymbol{c}_3/\boldsymbol{c}, \boldsymbol{c}_4/\boldsymbol{d}, \boldsymbol{c}_5/\boldsymbol{e}, \boldsymbol{c}_6/\boldsymbol{f}\} \tag{35}$$

ktorá jednoznačne špecifikuje postupnosť vektorov. Výhoda tohto postupu je v tom, že presnosť rekognoskácia nedegraduje tak rýchlo ako pri pôvodnom postupe založenom na pamäťovom vektore (33).

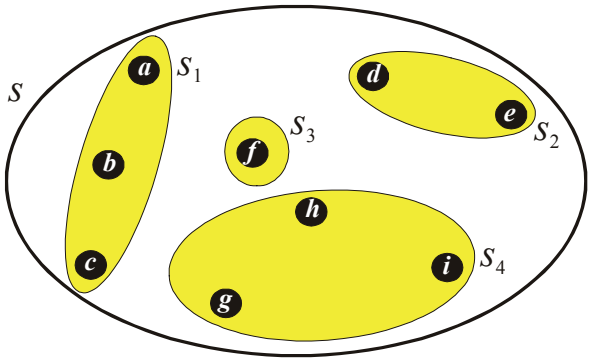
Príklad 6.9. Čo je to agregovaná pamäť?

Riešenie. Agregovaná pamäť pomáha prekonávať problémy s degradáciou pamäti pre postupnosť symbolov. Majme množinu konceptuálnych vektorov $S = \{\boldsymbol{a}, \boldsymbol{b}, ..., \boldsymbol{k}, \boldsymbol{l}, ...\}$, túto množinu rozložíme na disjunktné podmnožiny - *agregáty*

$$S = S_1 \cup S_2 \cup S_3 \cup S_4 \cup ... \qquad \left(S_i \cup S_j = \varnothing, \text{pre } i \neq j\right) \tag{36}$$

Študujme množinu $S = \{a, b, c, d, e, f, g, h\}$, jej rozklad na agregáty vypadá takto (pozri obr. 8)

$$S_1 = \{a, b, c\},\ S_2 = \{d, e\},\ S_3 = \{f\}, \text{ and } S_4 = \{g, h\} \tag{37}$$



Agregovaná pamäť je reprezentovaná pamäťovým vektorom, ktorý reprezentuje postupnosť agregátov $\{\boldsymbol{s}_1 \to \boldsymbol{s}_2 \to \boldsymbol{s}_3 \to \boldsymbol{s}_3\}$

$$\boldsymbol{t} = \boldsymbol{s}_1 + \boldsymbol{s}_1 \otimes \boldsymbol{s}_2 + \boldsymbol{s}_1 \otimes \boldsymbol{s}_2 \otimes \boldsymbol{s}_3 + \boldsymbol{s}_1 \otimes \boldsymbol{s}_2 \otimes \boldsymbol{s}_3 \otimes \boldsymbol{s}_4 \qquad (38b)$$

pričom jednotlivé agregáty sú definované takto pomocou príslušných postupnosti vektorov (pozri obr. 9)

$$\boldsymbol{s}_1 = \boldsymbol{a} + \boldsymbol{a} \otimes \boldsymbol{b} + \boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c} \qquad (38c)$$

$$\boldsymbol{s}_2 = \boldsymbol{d} + \boldsymbol{d} \otimes \boldsymbol{e} \qquad (38d)$$

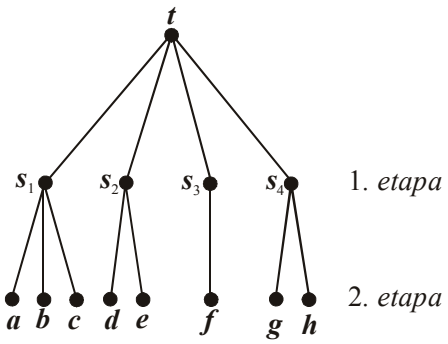
$$\boldsymbol{s}_3 = \boldsymbol{f} \qquad (38e)$$

$$\boldsymbol{s}_4 = \boldsymbol{g} + \boldsymbol{g} \otimes \boldsymbol{h} \qquad (38f)$$

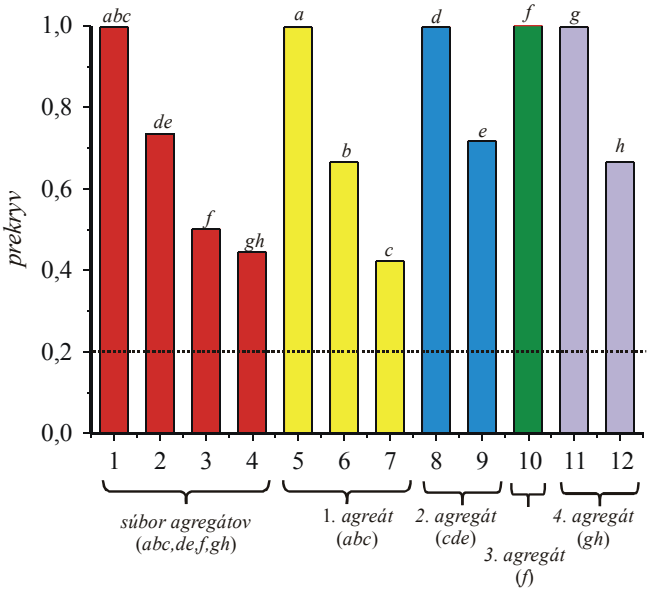
Proces spracovania agregovanej pamäti obsahuje dve etapy:

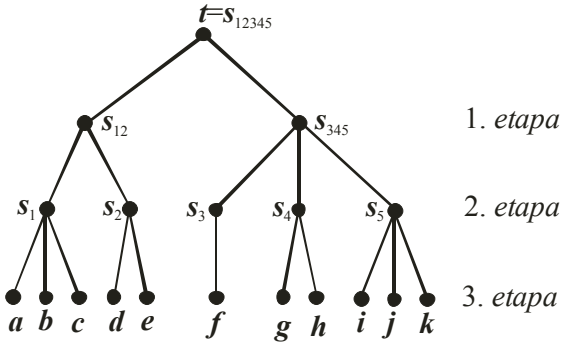
1. etapa – pomocou procesu „čistenia“ identifikujeme agregáty vyskytujúce sa v $\boldsymbol{t}$ (predpokladáme, že „upratovanie“ má množinu $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ z konca 2. kapitoly, kde bol tento proces špecifikovaný, rozšírenú aj o agregáty $\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4$, t.j. v našom ilustračnom príklade $X = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{h}, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4\}$).

2. etapa – identifikované agregáty sú ďalej analyzované pomocou procesu upratovania.



Výsledky dvoj-etapového procesu „čistenia“ sú znázornené na nasledujúcom obrázku. Z tohto obrázku vyplýva, že v prípade dlhej sekvencie konceptuálnych vektorov rýchla degradácia procesu „čistenia“ môže čiastočne prekonaná pomocou agregácia konceptov na agregáty, ktoré sú na najvyššej úrovni samostatne kódované.





Z uvedených ilustračných príkladov vyplýva, že prístup agregovanej pamäti reprezentuje efektívny spôsob prekonania rýchlej degradácie pôvodnej verzie postupnej rekognoskácie vektora (33). Tým, že združíme niekoľko konceptuálnych vektorov do agregátu, získame jednoduchú možnosť rozšírenia našich možností korektne rekognoskovať väčšie množiny konceptuálnych vektorov. Proces agregácie môže mať niekoľko hierarchických úrovní, čím sa naše možnosti zapamätania a rekognoskácie konceptuálnych vektorov stávajú skoro neohraničené.