Asociativní paměť: – jednoduchá paměť, kde $\vec{e} = \vec{x} \cdot W$

Heteroasociativní

přechází z n do k dimenzí, kde $n > k$ → pattern matching

Autoasociativní:

přechází z n do n dimenzí → noise reduction

Pattern recognition

přechází z n do 1 dimenze. → pattern recognition

Recurrent assoc. network:

výstup $x \cdot W$ je vstupem pro další iteraci.

Iteruji tak dlouho, než dojdu do stabilního stavu.

Eigenvector automata

– attractor → ten vektor, který bude převládat na výstupu

– mějme $W = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$, pak po dost iteracích

x_1 bude transformováno do $(2^t x_1, x_2)$, x_1 je tedy attractor

Associative learning

– activation = $\underline{\underline{sgn(x)}}$

Hebbian learning

$$W = [w_{ij}]_{n \times k} = [x_i \cdot y_j]_{n \times k}$$

→ je to proto correlation matrix mezi vstupem a výstupem

$$\vec{x}^1 \cdot W = \left(y_1^1 \cdot \sum_i^n x_i^1 x_i^1, y_2^1 \cdot \sum_i^n x_i^1 x_i^1, \dots, y_k^1 \cdot \sum_i^n x_i^1 \cdot x_i^1 \right)$$

$$= y^1 \cdot (x^1 \cdot x^1)$$

→ tohle všechno jsou vektory

Zároveň platí: $sgn(x \cdot w) = y$ a $sgn(-x \cdot w) = -y$

Hebbian inference:

$W = W^1 + W^2 + \dots + W^m$ (n inputs, m outputs)

$W^l = [x_i^l y_j^l]_{n \times h}$ — n dims., h dims

excitation:

$\vec{x^p} \cdot W = \vec{x^p} \cdot (W^1 + \dots + W^m)$

$\vec{x^p} \cdot W = \vec{x^p} \cdot W^p + \sum_{l \neq p} \vec{x^p} \cdot W^l$

$= \vec{y^p} \cdot (\vec{x^p} \cdot \vec{x^p}) + \sum_{l \neq p} \vec{y^p} \cdot (\vec{x^l} \cdot \vec{x^p})$

cross talk

Produkuje přesně output $y^p = x^p$, pokud cross talk je 0. $\Rightarrow$ to je pokud jsou ortogonální

$(\vec{x^p} \cdot \vec{x^p}) > 0 \Rightarrow$

perturbation

$$sgn(x \cdot W) = sgn\left(\vec{y^p} + \sum_{l \neq p} \vec{y^p} \frac{(\vec{x^l} \cdot \vec{x^p})}{(\vec{x^p} \cdot \vec{x^p})}\right)$$

$\Rightarrow$ obecně jde o projekci vstupu do vektorového prostoru vektorů uložených v asociativní paměti

$$\vec{z} \cdot W = \vec{z} \cdot W^1 + \vec{z} \cdot W^2 + \dots + \vec{z} \cdot W^m$$
$$= c_1 \vec{x}^1 + c_2 \vec{x}^2 + \dots + c_m \vec{x}^m$$

Capacity problem

$\Leftarrow$ pokud nejsou ortogonální, tak ještě méně

Associative network can hold up to $m \approx 0{,}18n$

Pseudoinverse learning

$\tilde{X}$ is pseudoinverse iff: $X\tilde{X}X = X$

$\tilde{X}X\tilde{X} = \tilde{X}$

$\tilde{X}X$ a $X\tilde{X}$ are symetrical

Chceme minimalizovat $\|Xw - y\|^2$, tedy $W = \tilde{X}y$

XX^T vždy nemusí inverz → přidáme proto malý šum $XX^T + \lambda I$

- lze najít pomocí BP network

BAM

- sgn activation again, bipolar values
- network which maps n dims to k dims

$\vec{y_0} = sgn(\vec{x_0} W)$, jako feedback $\vec{x_1} = sgn(W \cdot \vec{y_0}^T)$

$\vec{y_1} = sgn(\vec{x_1} W)$, ...

- pomocí Hebbian learningu může nalézt stabilní stavy, že:

$$y = (xW) \quad a \quad x = (Wy^T)$$

kde $W = x^T y$

$$y = sgn(xW) = sgn(xx^T y) = sgn(\|x\|^2 y) = sgn(y) = y$$

ekvivalentně pro x...

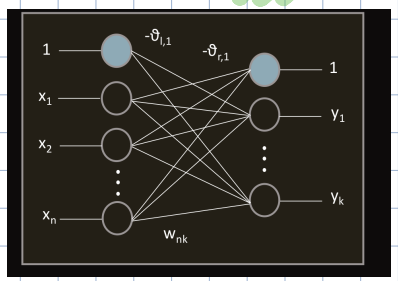
Energy function

$E \hat{=} -x_0 e^T = -x_0 W y^T$, protože $e^T = Wy^T$ $\Leftrightarrow x = sgn(e^T)$

→ takže čím blíž bude predikci, tím menší bude moje energy function

Minimum téhle funkce reprezentuje stabilní stavy

$E(x_i, y_i) = -\frac{1}{2} x_i W y_i^T$ ~ $\frac{\lambda}{2}$ pouze kvůli snadnější derivaci...



$$E(\vec{x}_i, \vec{y}_i) = -\frac{1}{2}\vec{x}_i W \vec{y}_i^T + \frac{1}{2}\vec{\theta}_l \vec{y}_i^T + \frac{1}{2}\vec{x}_i \vec{\theta}_r^T$$

$\vec{\theta}_l$ .... the vector of thresholds of the k neurons (in the left layer)

$\vec{\theta}_r^T$ .... the vector of thresholds of the n neurons (in the right layer)

$y_j = \sum_i w_i x_i - r_{ij}$

Hopfield network

$$w_{ij} = \begin{cases} \sum_{s=1}^{m} x_i^s x_j^s & i \neq j \\ 0 & i = j \end{cases}$$

$x_i^s \in \{-1, 1\}$, $\quad y_i(0) = x_i$

$$y_i(t+1) = f\left(\sum_{i=1}^{n} w_{ij}\, y_i(t) \right)$$

(sum: going over all neurons)

→ Hebbian doktrína nemůže output stabilizovat

→ výsledné neurony mají reprezentovat nejbližší uložený pattern k danému vstupu

Capacity:

$m < 0{,}15n$ — > ortogonalita zajišťuje stabilitu

Potentials:

diagonála

$$\xi = x_1 \cdot W = x_1 \left(x_1^T x_1 + \dots + x_m^T x_m - m I \right) =$$

$$= x_1 x_1^T x_1 + x_1 \cdot x_2^T x_2 + \dots + x_1 x_m^T x_m - m x_1 I$$

$x_1 x_1^T x_1 = n$, $\quad x_1 \cdot x_2^T x_2 = \alpha_{12}$, $\quad x_1 x_m^T x_m = \alpha_{1m}$ — > sk. součin

$$= (n - m) \cdot x_1 + \sum_{j=2}^{m} \alpha_{1j} x_j$$

perturbation

pokud tenhle term je malý, je síť stabilní

— opět pomáhá ortogonalita

Diagonála musí být 0_m, jinak periodicky měním patterny

Energy function

$$E(x) = -\frac{1}{2} x W x^T = -\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} w_{ij} x_i x_j$$

— pro thresholds:

$$E(x) = -\frac{1}{2} x W x^T + \theta x^T$$

$$= -\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} w_{ij} x_i x_j + \sum_{j=1}^{n} \theta_i x_i$$

Perceptron for Hebbian learning

$$\vec{v} = (\underbrace{w_{12}, w_{13}, \dots, w_{1n}}_{n-1 \text{ components}}, \underbrace{w_{23}, w_{24}, \dots, w_{2n}}_{n-2 \text{ components}}, \dots, \underbrace{w_{n-1,n}}_{1 \text{ component}}, \underbrace{-\vartheta_1, \dots, -\vartheta_n}_{n \text{ components}})$$

→ dimenze: $n-(n-1)/2$

→ horní pravý trojúhelník z matice W

$$\vec{z}_1 = (\underbrace{x_2, x_3, \dots, x_n}_{n-1 \text{ components}}, \underbrace{0, 0, \dots, 0}, \underbrace{1, 0, \dots, 0}_{n \text{ components}})$$

$$\vec{z}_2 = (\underbrace{x_1, 0, \dots, 0}_{n-1 \text{ components}}, \underbrace{x_3, \dots, x_n}_{n-2 \text{ components}}, 0, 0, \dots, \underbrace{0, 1, \dots, 0}_{n \text{ components}})$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$\vec{z}_n = (\underbrace{0, 0, \dots, x_1}_{n-1 \text{ components}}, \underbrace{0, 0, \dots, x_2}_{n-2 \text{ components}}, 0, 0, \dots, \underbrace{0, 0, \dots, 1}_{n \text{ components}})$$

→ dimenze $n + \frac{n-(n-1)}{2}$

$$\begin{aligned}
&\text{Neuron 1:} && \operatorname{sgn}(x_1) && \vec{z}_1 \cdot \vec{v} && > && 0 \\
&\text{Neuron 2:} && \operatorname{sgn}(x_2) && \vec{z}_2 \cdot \vec{v} && > && 0 \\
&\vdots && \vdots && \vdots && \vdots && \vdots \\
&\text{Neuron n:} && \operatorname{sgn}(x_n) && \vec{z}_n \cdot \vec{v} && > && 0
\end{aligned}$$

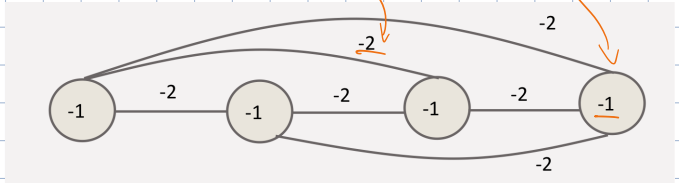
Optimization problems

Bitflip

Find minimum of: $E(x) = \left(\sum_{i=1}^{n} x_i - 1\right)^2$

→ bude 0 pouze pokud jeden je nastavený na 1 a ostatní na 0.

$$= -\frac{1}{2}\sum_{i \neq j}^{n} -2 x_i x_j + \sum_{i=1}^{n} -1 x_i + 1$$



$$\left(x_1 - 1 + x_2 - 1 + x_3 - 1 \dots\right)^2$$

$$\left(x_1 + x_2 + x_3 \cdots + x_n - n\right)^2$$

N-rooks problem

x_{ij} → rook on position ij

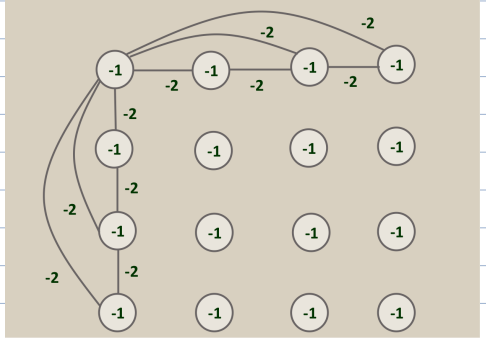
$$\sum_{i=1}^{n} x_{ij} = 1, \quad \sum_{j=1}^{n} x_{ij} = 1$$

→ only one in row/column

multiflop

find minimum of for columns: $E_1(x) = \sum_{j=1}^{n}\left(\sum_{i=1}^{n} x_{ij} - 1\right)^2$

find minimum of for rows: $E_2(x) = \sum_{i=1}^{n}\left(\sum_{j=1}^{n} x_{ij} - 1\right)^2$



TSP

city \ „the order" of visits	1	2	3	4
M_1	1	0	0	0
M_2	0	1	0	0
M_3	0	0	1	0
M_4	0	0	0	1

Convention:
the $(n+1)$-st column is the same one like the first column
→ **round trip**

- $x_{i,k}$ ..... neuron state ~ entry i, k of the matrix
- $x_{i,k} = x_{j,k+1} = 1$ ... city M_i is visited in step k and M_j in step $(k+1)$
- $d_{i,j}$ ..... distance between M_i and M_j
  → add $d_{i,j}$ to the length of the round trip

Tohle chci minimalizovat →

$$L = \frac{1}{2}\sum_{i,j,k}^{n} d_{i,j} x_{i,k} x_{j,k+1}$$

× at the same time, a single „1" is allowed in each column and row
→ *include the constraints for a valid round trip*
→ ***minimize*** E:

$$E = \frac{1}{2}\sum_{i,j,k}^{n} d_{i,j} x_{i,k} x_{j,k+1} + \frac{\gamma}{2}\left(\sum_{j=1}^{n}\left(\sum_{i=1}^{n} x_{i,j} - 1\right)^2 + \sum_{i=1}^{n}\left(\sum_{j=1}^{n} x_{i,j} - 1\right)^2\right)$$

$d_{ij} x_{ik} x_{jk+1}$ -> přešel jsem z města i do města j v čase $k(+1)$

$$w_{i,j,k,l} = \begin{cases} -d_{i,k} & \text{if } (i \neq k) \text{ and } [(j+1=l) \text{ or } (j=l+1)] \\ -\gamma & \text{if } (i=k) \text{ and } (j \neq l) \\ -\gamma & \text{if } (i \neq k) \text{ and } (j=l) \\ 0 & \text{otherwise} \end{cases}$$

$$\vartheta_{i,j} = -\gamma/2$$

Stochastic approach

- how to avoid bad minima?
  -> increase the number of paths in a search space
  -> allow updating a state even though energy function increases

Continuous hopfield

$$x_i = S(u_i) = \frac{1}{1+e^{-u_i}}, \qquad E(x) = -\frac{1}{2}\sum_{i}^{N}\sum_{j}^{N} w_{ij} x_i x_j + \sum_{i}^{N}\int_{0}^{x_i} S^{-1}(x)\,dx$$

Simulated annealing

$$P_{\Delta E} = \frac{1}{1+e^{\Delta E/T}}$$

-> pravděpodobnost, že uděláním update, přestože zapříčiní ΔE zvýšení energy funkce
-> pro velké T dostaneme $p \sim \frac{1}{2}$, naopak $T=0$ zajišťuje pouze zlepšující update

- dokáže najít optimum, nehodí se u velmi těžkých problémů

Boltzmann machine

$$x_i = \begin{cases} 1 & \text{with prob } p_i \\ 0 & \text{with prob } 1-p_i \end{cases} \qquad \text{where } p_i = \frac{1}{1+e^{-\left(\sum_{j=1}^{n} w_{ij} x_j - \theta_i\right)/T}}$$

pravděpodobnost přechodu není nikdy 0.

Never settles on single state

When T small, $p_i \sim 1$ pokud $\sum_{j=1}^{n} w_{ij} x_j - \theta_i > 0$
jinak $p_i \sim 0$

$$E(x) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i x_j + \sum_{j=1}^{N} \theta_j x_j$$