Neurónové siete
vypracované otázky ku skúške 2011

Zdroje:
1. Igor Farkaša – http://ii.fmph.uniba.sk/~farkas/Courses/ns.html
2. Wikipedia

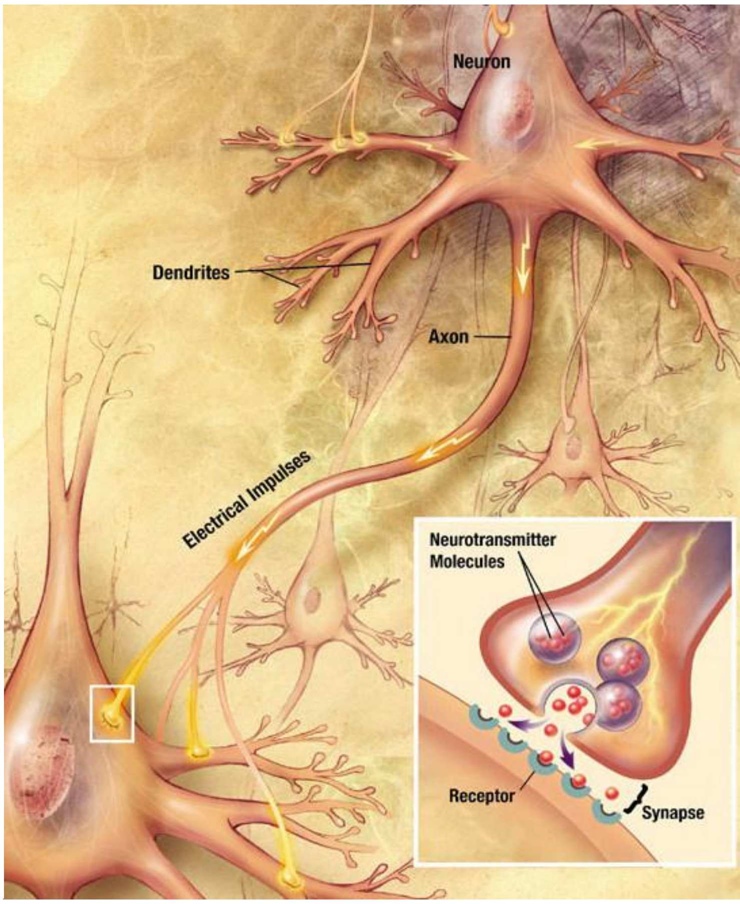
(poznámka: červeným zvýraznené časti otázok nie sú vypracované)

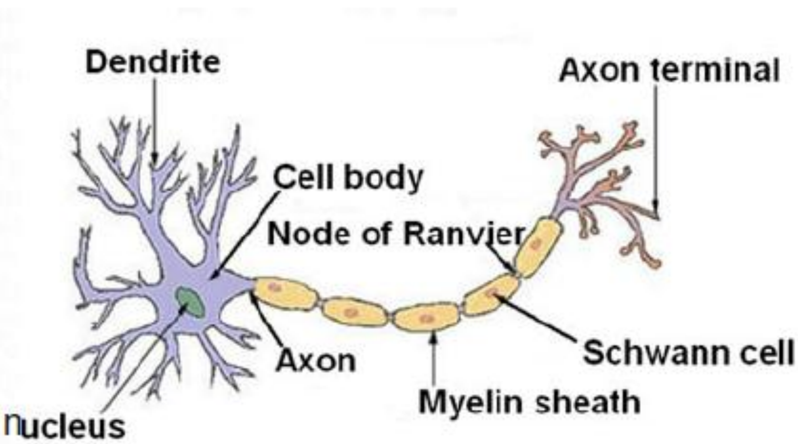
1. Stručná história konekcionizmu, základné časti biologických neurónov a ich vlastnosti, typické vlastnosti konekcionistického modelu, klasický konekcionizmus, model s prahovou logikou, implementácia Booleových funkcií.

Konekcionizmus je inšpirovaný biológiou - mozgom. Je založený na umelých neurónových sieťach. Modeluje mentálne procesy, aplikácie v praktických problémoch. Mozog má 10^11 neurónov a 10^14 synáps.

História:
- do 40-tych rokov – klasický – filozofia, psychológia,
- do 70-tych rokov – starý – začiatok počítačov, umelých NN, spája sa s kognitívnou vedou,
- od 1986 – nový – paralelné distribuované subsymbolové, multi-layer, rekurentné,
- od konca 90-tych rokov – pravdepodobnostné metódy.

Neurónové siete - najpoužívanejší konekcionistický model v súčasnosti. Vychádzajú z nasledovných princípov:
- každý mentálny stav možno reprezentovať n-dimenzionálnym vektorom aktivačných hodnôt,
- pamäť sa tvorí modifikáciou sily prepojení medzi neurónmi.

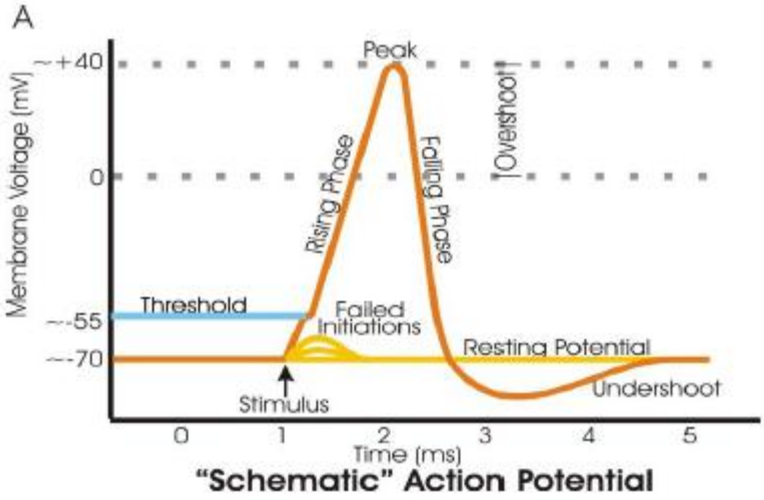




Model s prahovou logikou:

Neurón páli (vysiela signál) ak signál, ktorý dostane cez axóny je dostatočne silný (tzn. že presiahne prah). To zvyčajne znamená jednu z nasledujúcich situácii:

- *časová sumácia* – v (krátkom) čase prídu viaceré impulzy po tej istej synapse
- *priestorová sumácia* – naraz prídu impulzy po viacerých synapsách.

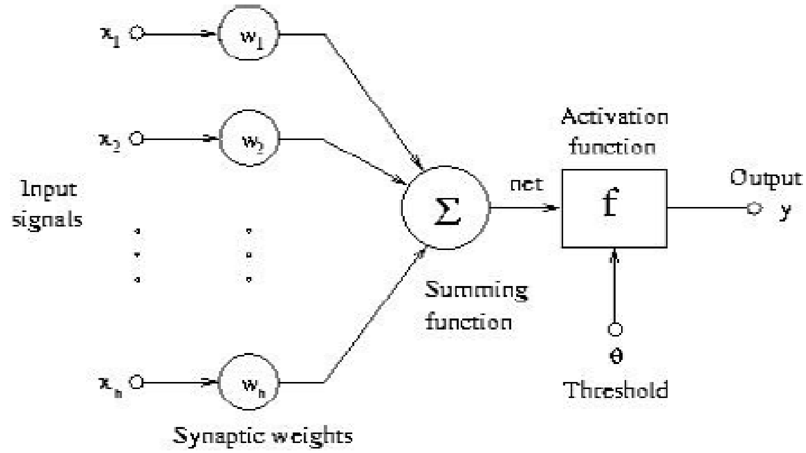


Typický model umelého neurónu:

1. prijme signál z ostatných neurónov,
2. spracuje prijatý signál,
3. pošle spracovaný signál ostatným neurónom.

Deterministic model

$$y = f\left(\sum_i w_i x_i - \theta\right)$$



Stochastic model $P(s=+1) = 1/(1+\exp(-\sum_i w_i x_i + \theta))$

Implementácia Booleových funkcií

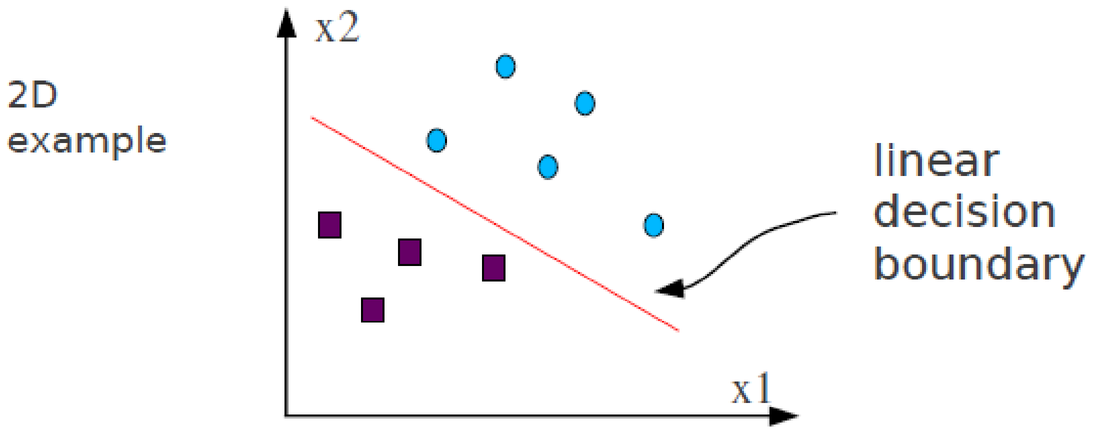
Neurón vie simulovať každú lineárne separovateľnú booleovskú funkciu. Na ostatné treba dvojvrstvovú sieť.

2. Binárny perceptrón: pojem učenia s učiteľom, učiace pravidlo, algoritmus trénovania, deliaca nadrovina, klasifikácia vzorov, lineárne separovateľné problémy, definícia a príklad.

Binárny perceptrón – vracia len hodnoty 0 a 1 (prípadne -1, +1) – teda delí vstupné dáta na dve podmnožiny. Z toho vyplýva, že dokáže riešiť len lineárne separovateľné problémy – teda také, ktoré ak si zobrazíme v k-dimenzionálnom priestore (kde k je dĺžka vstupného vektora), tak medzi tieto dve skupiny musíme vedieť skonštruovať deliacu nadrovinu.

$w_1x_1 + w_2x_2 + \ldots + w_nx_n = \theta$

linear separability of two classes



Učenie s učiteľom: máme k dispozícii cieľovú hodnotu (prípadne vektor) y pre každú vstupnú hodnotu x.

Algoritmus:

Vstup: množina dvojíc {x, d} kde x je vektor a d je cieľová hodnota.
inicializácia: vygeneruj náhodné váhy, nastav koeficient učenia.

1. nastav celkovú chybu E := 0, náhodne zamiešaj vstupnú množinu vzoriek
2. pre všetky vzorky x zo vstupnej množiny
    a. vyber vstupnú vzorku x, vypočítaj výstup y
    b. vypočítaj kvadratickú chybu učenia: $e(t) = ½(d - y)^2$, nastav E += e(t)
    c. ak e(t) > 0 zmeň váhy pomocou učiaceho pravidla
3. Ak E = 0 (nulová chyba) tak skonči, inak choď na bod 1.

Učiace pravidlo $\mathrm{w_j(t+1) = w_j(t) + \alpha(d-y)x_j}$

Aktivačná funkcia $\mathrm{y} = f\left(\sum_{j=1}^{n+1} w_j x_j\right) \quad x_{n+1} = -1$

kde *f* je nejaká prahová funkcia: unipolárna {0,1} alebo bipolárna {-1,+1}

3. Spojitý perceptrón: Rôzne aktivačné funkcie perceptrónu, chybová funkcia a spôsob jej minimalizácie, pojem učenia s učiteľom, učiace pravidlo, algoritmus trénovania perceptrónu.

Spojitý perceptrón: Nelineárna jednotka, štandardne používa sigmoidnú aktivačnú funkciu.

Algoritmus trénovania – rovnaký ako pri binárnom perceptróne.

Alternatívne aktivačné funkcie:

sigmoida: $y = 1 / (1 + e^{-net})$ softmax: $\mathrm{y}_i = \frac{e^{net_i}}{\sum_j e^{net_j}}$ hypertangens: tanh(net)

Chyba učenia – kvadratická rovnako ako pri binárnom perceptróne: $e(t) = ½(d - y)^2$

Učiace pravidlo – dve možnosti (p je zvolená vzorka dát):

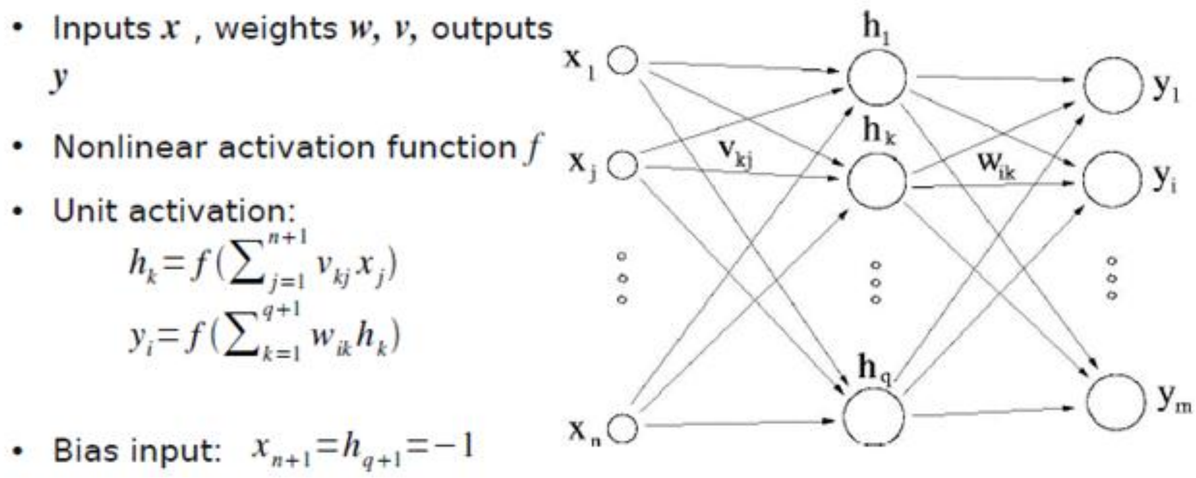
(stochastické, online) gradient descent: $w_j(t+1) = w_j(t) + \alpha(d-y)f' x_j$ ***kde*** $f' = y(1-y)$

(alternatívne) batch: $w_j(t+1) = w_j(t) + \alpha \sum_p \left(d^{(p)} - y^{(p)}\right) x_j^{(p)}$

Učenie s učiteľom: máme k dispozícii cieľovú hodnotu (prípadne vektor) d pre každú vstupnú hodnotu x.

4. Viacvrstvové dopredné neurónové siete: architektúra a aktivačné vzorce, odvodenie metódy učenia pomocou spätného šírenia chýb (back-propagation) pre dvojvrstvovú doprednú NS, modifikácie BP, typy úloh pre použitie doprednej NS.

Sú zovšeobecnením jednoduchých perceptrónov. Obsahujú skryté vrstvy, neuróny majú nelineárnu aktivačnú funkciu.

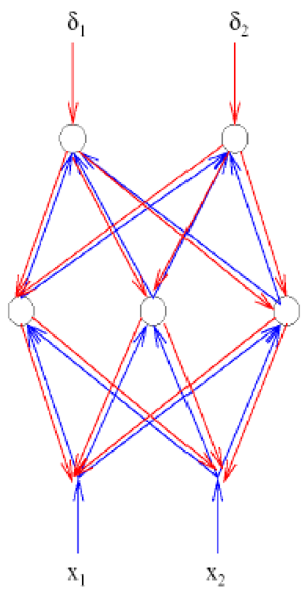


Alternatívne aktivačné funkcie:

sigmoida: $y = 1 / (1 + e^{-net})$ softmax: $y_i = \frac{e^{net_i}}{\sum_j e^{net_j}}$ hypertangens: tanh(net)

Chyba učenia – kvadratická, počíta sa cez všetky výstupné neuróny: $e^{(p)} = \frac{1}{2}\sum_i (d_i^{(p)} - y_i^{(p)})^2$

Back-propagation



Vstup: množina dvojíc $\{x^{(p)}, d^{(p)}\}$ kde x je vektor a d je cieľová hodnota.
inicializácia: vygeneruj náhodné váhy, nastav koeficient učenia.

1. nastav celkovú chybu E := 0
2. pre všetky vzorky x zo vstupnej množiny
   a. vyber vstupnú vzorku x, vypočítaj výstup y
   b. vypočítaj chybu a pripočítaj ju k celkovej E += $e^{(p)}$
   c. vypočítaj δ_i a δ_k (spätný prechod)
   d. uprav všetky váhy w_{ik} a v_{kj}
3. ak je splnené kritérium zastavenia tak skonči, inak pokračuj bodom 1.

Učiace pravidlo

Hidden-output weights:

$$w_{ik}(t+1) = w_{ik}(t) + \alpha\delta_i h_k \quad \text{where} \quad \delta_i = (d_i - y_i) f_i'$$

Input-hidden weights:

$$v_{kj}(t+1) = v_{kj}(t) + \alpha\, \delta_k x_j \quad \text{where} \quad \delta_k = (\Sigma_i w_{ik}\, \delta_i) f_k'$$

Kritérium zastavenia (napríklad):

- zmena celkovej chyby E dostatočne malá (< 1 % oproti predchádzajúcej epoche),
- ak sieť dobre generalizuje.

Modifikácia BP

- **momentum** – na preskočenie lokálnych miním,
- **permutácia vstupu** – náhodné preusporiadanie pred každou epochou,
- **weight decay** – váhy sa vynásobia konštantou.

Príklady použitia

- XOR – lineárne neseparovateľne problémy (2-class),
- kompresia obrázkov,
- čítanie anglického textu,
- rozpoznávanie ručné písaných PSČ,
- predikcia časových radov.

5. Viacvrstvová dopredná NS ako univerzálny aproximátor funkcií (formulácia teorémy), trénovacia a testovacia množina, zovšeobecňovanie, preučenie, skoré zastavenie učenia, selekcia modelu, prekrížená validácia.

Trojvrstvová sieť je schopná s danou presnosťou aproximovať ľubovoľnú spojitú funkciu.

Veta: Majme $A_{train} = \{x^{(1)}, ..., x^{(p)}, ..., x^{(N)}\}$, $x^{(p)} \in \mathbf{R}^n$. Pre $\varepsilon > 0$ a ľubovoľnú spojitú funkcu F: $\mathbf{R}^n \rightarrow (0,1)$ definovanú na diskrétnej množine A_{train} existuje funkcia G:

$$G(x^{(p)}) = f\left(\sum_{k=1}^{q+1} w_k f\left(\sum_{j=1}^{n+1} v_{kj} x_j^{(p)}\right)\right)$$

kde parametre w_k, $v_{kj} \in \mathbf{R}$ a $f(z) = \mathbf{R}^n \rightarrow (0,1)$ je spojitá monotónne rastúca funkcia spĺňajúca

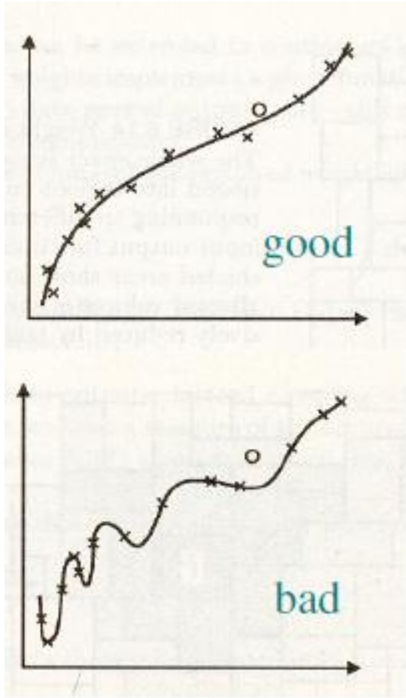
$f(-\infty) = 0$ a $f(\infty) = 0$ taká že $\sum_p | F(x^{(p)}) - G(x^{(p)}) | < \epsilon.$

Hovoríme, že G aproximuje F na A_{train} s presnosťou ε.

G môžeme interpretovať ako dvojvrstvovú doprednú NS s jedným výstupným neurónom.

Zovšeobecnenie, trénovacia a testovacia množina

Množinu všetkých dát rozdelíme na dve: $A = A_{train} \cup A_{test}$



Kvalita generalizácie je ovplyvnená:

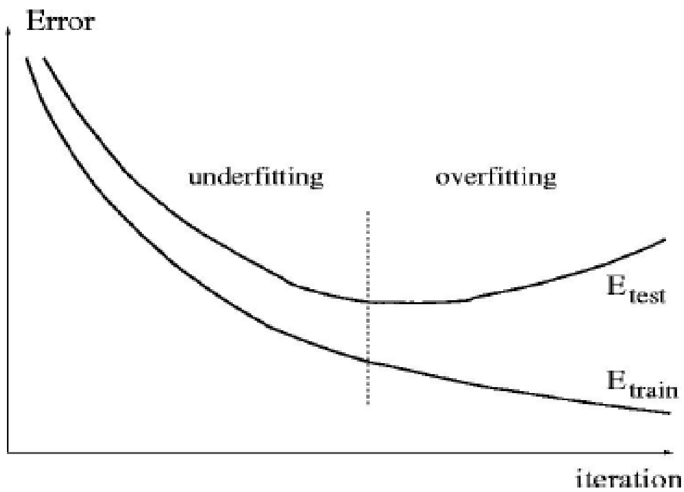
- reprezentatívnosťou vybranej testovacej množiny,
- architektúrou siete,
- zložitosťou problému.

Selekcia modelu – v princípe máme dve možnosti:

- zafixovať veľkosť trénovacej množiny a hľadať optimálnu sieť (network pruning, Occam's razor),
- zafixovať sieť, hľadať optimálnu veľkosť trénovacej množiny (teória založená na VC dimenzii).

Early stopping

Trénovanie sa snažíme zastaviť na hranici medzi podučením a preučením – teda skôr ako sa začne generalizačná sila zhoršovať.



Cross-validation

Trénovaciu množinu rozdelíme ďalej na estimačnú a validačnú podmnožinu. Najlepší model potom hľadáme tak, že modely trénujeme a validujeme na estimačnej a validačnej podmnožine. Najlepší z nich potom môžeme použiť na testovacej množine a overiť tak jeho kvalitu. Používa sa early stopping.

K-fold cross-validation

Rozdelíme A_{train} na k podmnožín A_1 .. A_k. Následne k-krát trénujeme (použijeme early stopping) každý model NS na množine A_{train} - A_i a validujeme na množine A_i. Ako najlepší vyberieme ten ktorý je má najlepšie validačné výsledky. *Extrémny prípad:* veľkosť A_i = 1.

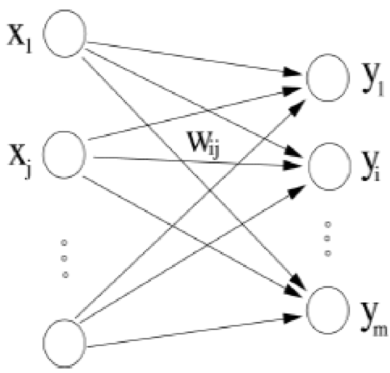
6. Lineárne modely NS: vzťah pre riešenie systému lineárnych rovníc v jednovrstvovej sieti, pojem pseudoinverzie matice, autoasociatívna pamäť: lineárny obal, princíp funkcie modelu, detektor novosti.

Input vector: $\boldsymbol{x} = [x_1, x_2, .., x_n]^T$

Output vector: $\boldsymbol{y} = [y_1, y_2, .., y_m]^T$

Weight matrix: $\boldsymbol{W} \sim$ type $[m \times n]$

Linear transformation $\varphi \ : \Re^n \to \Re^m \,,\ \boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$



Pridávanie vrstiev je zbytočné, lebo lineárne zobrazenie je uzatvorené na skladanie.

Pseudoinverzia matice

Trénovacia množina (veľkosti N) má vstupy X a výstupy Y. Platí $y^{(p)} = W\, x^{(p)}$ pre každý prípad p, v maticovej notácii Y = WX.

$$\underset{(m\times N)}{[y^{(1)}\, y^{(2)} \dots y^{(N)}]} = \underset{(m\times n)}{W} \times \underset{(n\times N)}{[x^{(1)}\, x^{(2)} \dots x^{(N)}]}$$

Ak je X regulárna tak $W = YX^{-1}$. Inak $W = YX^{+}$, kde X^{+} je pseudoinverzná matica k matici X.

a) $X^{+} = X^{T}(XX^{T})^{-1}$, if $n < N$ and $rank(X) = n$.
b) $X^{+} = (X^{T}X)^{-1}X^{T}$, if $n > N$ and $rank(X) = N$.

Autoasociatívny prípad

Uvažujme N < n a autoasociatívny prípad $y^{(p)} = W\, x^{(p)}$, m = n.
Model NS by si mal zapamätať/natrénovať N prototypov $[x^{(1)}\, x^{(2)} \dots x^{(N)}]$.

Cieľ: natrénovať na prototypoch a potom pustiť na poškodenej verzii prototypu. Model by mal byť schopný prototyp zrekonštruovať.

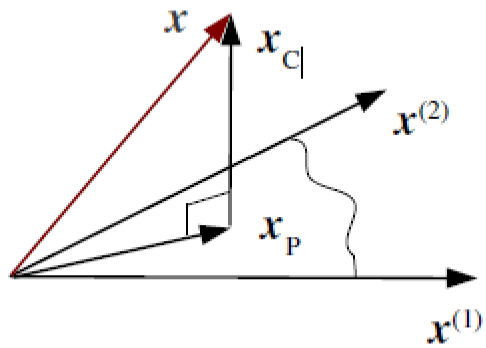
V špeciálnom prípade ak N = n existuje triviálne riešenie – W = I (identita).

Linear manifold $\mathscr{L} = \{x \in \Re^{n} \mid x = a_1 x^{(1)} + a_2 x^{(2)} + \dots + a_N x^{(N)},\ a_p \neq 0\}$

$\mathscr{L} \subset \Re^{n}$

Orthogonal complement $\mathscr{L}^{\perp} = \{x \in \Re^{n} \mid x \perp \mathscr{L}\}$

Každý vektor $x \in R^{n}$ vieme jednoznačne rozložiť:
$x = x_p + x_c$ kde $x_p \in \mathscr{L}$ a $x_c \in \mathscr{L}^{\perp}$.



Čo robí autoasociacívna NS?
Tréningová množina $A_{train} = \{x^{(1)}, \dots, x^{(p)}, \dots, x^{(N)}\}$ tvorí Lineárny obal $\mathscr{L}$. NS považuje každé smer každého x z $\mathscr{L}$ ako šum, ktorý je potrebné odfiltrovať pred projekciou x do $\mathscr{L}$. Z odchýliek sa spraví ortogonálna projekcia(odstráni sa šum) vynásobením zľava $W = XX^{+}$. Keby sa to zľava vynásobilo ($W = I - XX^{+}$), máme bázový vektor z R^{n} – L, kolmý na L – **detektor novosti** je model s týmto operátorom.

7. Lineárne modely NS: účel Grammovho-Schmidtovho ortogonalizačného procesu, GI model. Pamäť korelačnej matice ako autoasociatívna pamäť, vzťah pre výpočet váh, presluch, porovnanie s GI.

Pridanie novej vzorky nemusí byť až tak výpočtovo náročné ako rátať znovu pseudoinverziu.

General Inverse (GI) model: Ortogonalizačný proces: máme bázu u^1..u^k. Chceme ortogonálnu bázu v^1..v^k. v^1:=u^1. Keď už máme p vektorov určených, ďalší bude z priestoru gen. v^1..v^p,u^(p+1).v^(p+1) = u^(p+1) - ▯_i=1^p{(v^i^T * u^(p+1) / ||v^i||^2) * v^i}

Keď príde nový vektor x, čo nepatrí do priestoru, týmto vzorcom získame zložku kolmú na priestor.

W_0 = 0. W_(N+1) = W_N +(x_(N+1) * x_(N+1)^T / |x_(N+1)|^2).x_(N+1) = x – W_N*x.

Korelačná matica: Váha z j-teho vstupu do i-teho výstupu w_ij je úmerná sume x_j*y_i pre všetky x,y z trénovacej množiny. W=YX^T. Ak sú vstupy X ortonormálne, X^T=X^-1=X^+. Potom CMM=GI.

Autoasociácia: W=XX^T. Pre vstup x_p: Wx_p = x_p|x_p|^2 + C(p) – presluch z iných zložiek. Ak bybol nulový, x_1..x_N sú ortogonálne. Vo všeobecnosti teda nie je. Dá sa ale znížiť posunutím strednej hodnoty zložiek vstupov do nuly. x_k := x_k – 1/N �x_i. (obr.)

Porovnanie: Ak vstupy nie sú ortogonálne, GI s detektorom novosti je lepší. (obr. rozp. tvárí).

8. Samoorganizácia v NS, základné princípy, pojem učenia bez učiteľa, typy úloh použitia, Ojovo pravidlo učenia – vzťah pre adaptáciu váh a vysvetlenie konvergencie váhového vektora, pojem vlastného vektora a vlastného čísla.

Učenie bez učiteľa – algoritmus učenia nemá informáciu o požadovaných aktivitách výstupných neurónov v priebehu trénovania. Súperenie medzi váhami, aktualizácia váh blízkych víťazovi. Tendencia samozosilňovania – Hebbovo pravidlo. Využíva sa redundancia vo vstupných dátach (využíva sa klasterizácia – vektorová kvantizácia). Topologické zobrazenie príznakov.

Ak netreba hlavné komponenty, ale stačí podpriestor nimi generovaný, je tu Ojovo pravidlo: w_ij += �y_i(x_j -�{y_k*w_kj}), máme n vstupov a p výstupov, i=1..p, k=1..p

Výstupy nie sú usporiadané, variancia je cca rovnaká. Výsledky závisia od počiatočných podmienok a poradiu vstupov. Váhový vektor konverguje k vlastnému vektoru korelačnej matice vstupov xx^T.

Matica A má vlastné číslo v, ak je determinant A-vI nulový. Vlastné čísla trojuholníkovej a diagonálnej matice sú na diagonále. X je vlastný vektor prislúchajúci vlastnému číslu v, ak Ax=vx.

Architektúry – dopredné (+ laterálne spojenia)

9. Metóda hlavných komponentov pomocou algoritmu GHA a APEX, architektúra modelu, vzťah pre adaptáciu váh, pojem vlastných vektorov a vlastných čísel, redukcia dimenzie, aplikácia na kompresiu obrazu.

PCA = lineárna transformácia do priestoru príznakov. S tým súvisí redukcia dimenzie. Slúži na predspracovanie dát, ktoré majú cca Gaussovské rozloženie.

Príznakový priestor je n-dimenzionálny priestor, kde každý vzor je bod s n súradnicami. n je počet príznakov vzoru. Podobné vzory sú blízko pri sebe.

Matica A má vlastné číslo v, ak je determinant A-vI nulový. Vlastné čísla trojuholníkovej a diagonálnej matice sú na diagonále. X je vlastný vektor prislúchajúci vlastnému číslu v, ak Ax=vx. Majme vstupný vektor x, premietneme ho do priestoru príznakov na vektor u.

Variancia je u^T*Ru, kdeR=xx^T je korelačná matica vstupu. Keď variancia nadobúda extrém, Ru = vu, kde v sú vlastné čísla Ra u sú vlastné vektory R. Pre n rôznych vlastných vektorov dostaneme n rôznych projekcií a_i =x^T*u_i. a_i sú hlavné komponenty(príznaky). Rekonštrukcia: x = ▯{a_i*u_i}. Keď to zosumujeme len pre p najväčších vlastných čísel, dostaneme akceptovateľnú aproximáciu s redukovanou dimenziou.

GHA:

Majme n vstupov, p výstupov. Pre i=1..p w_ij += ▯y_i(x_j - ▯_k=1^i{y_k*w_kj}). Takto sa extrahuje p hlavných komponentov korelačnej matice vstupov usporiadaných podľa veľkosti. Netreba počítať korelačnú maticu. Výpočtovo nenáročný, ak p<<n. Je to reestimačný algoritmus.

Použitie: kódovanie dát, kompresia.

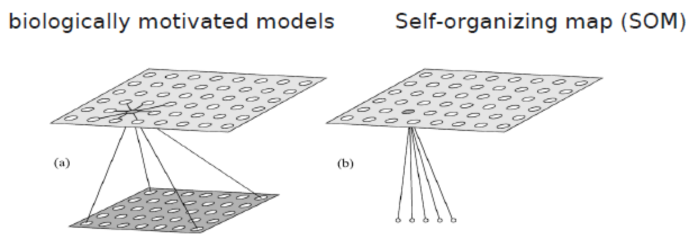
APEX:

Má aj laterálne spojenia, ktoré majú inhibičný účinok, ale konvergujú k 0. Váhy konvergujú k vlastnýmvektorom. Je to dekorelačný algoritmus.

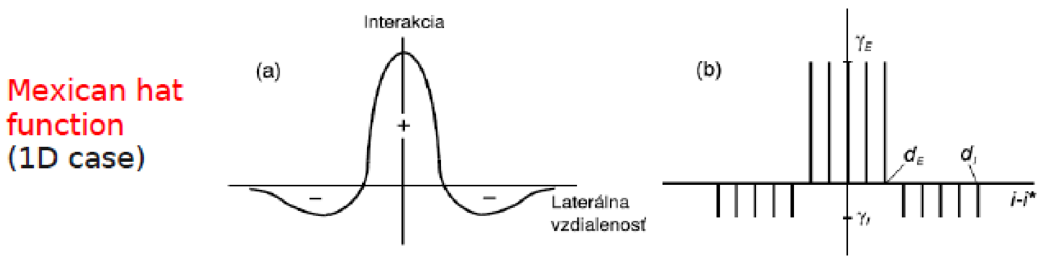
10. Učenie so súťažením (typu "winner-take-all"), nevýhody. Neurobiologická motivácia algoritmu SOM, laterálna interakcia a jej náhrada v SOM, sumarizácia algoritmu, voľba parametrov modelu, DP verzia algoritmu.

Winner-take-all v nelineárnej sieti – neuróny sa ovplyvňujú navzájom (kooperácia neurónov) a zároveň aktivujú samé seba (rekurencia) – postupne sieť dospeje do štádia kedy je aktívny len jeden výstupný neurón – korešpondujúci najsilnejšiemu vstupu.

Neurobiologická motivácia:
oko - sietnica je prepojená s mozgovou kôrou.

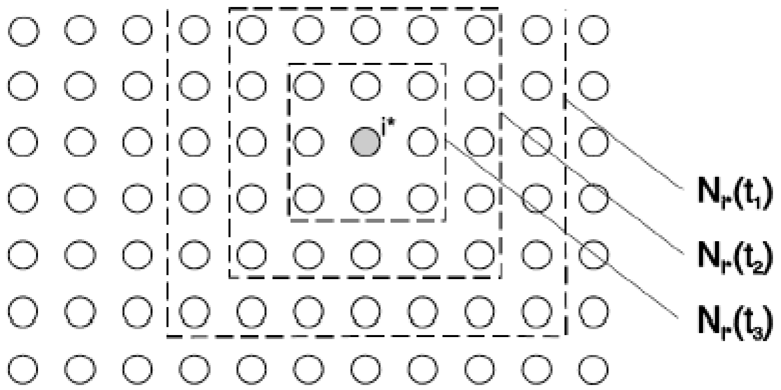


Laterálna interakcia: každý s každým, vplyv v závislosti od vzdialenosti má tvar mexického klobúka. Šírka mexického klobúka má byť dostatočne široká v porovnaní s šírkou lokálnych vstupov.



Náhrada – použijeme **funkciu susedstva:** neuróny sa aktualizujú len v susedstve víťaza. Veľkosť susedstva časom klesá. Susedstvo: *štvorcové* alebo *gaussovské*.

• rectangular neighborhood (below)



• alternative: gaussian neighborhood

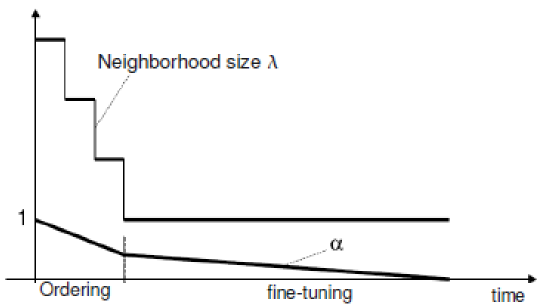
$$h(i^*, i) = \exp\{-\frac{d_E^2(i^*, i)}{\lambda^2(t)}\}$$

$$\lambda(t) = \lambda_i.(\lambda_f/\lambda_i)^{t/t_{max}}$$

Algoritmus

1. náhodne vyber vstup x
2. nájdi víťazný neurón i* pre x $i^* = \operatorname{argmin}_i \|x - w_i\|$
3. uprav váhy v susedstve víťazného neurónu
$$w_i(t+1) = w_i(t) + \alpha(t) * \mathrm{h}(i^*, i) * [x(t) - w_i(t)]$$
4. aktualizuj parametre – susedstvo, koeficient učenia
5. opakuj až kým nie je splnené kritérium zastavenia

Voľba parametrov modelu - proces učenia možno rozlíšiť dve fázy.
1. usporiadanie - klesá veľkosť okolia diskrétne s časom.
2. doladenie – možno ponechať najbližších susedov súčasťou okolia, až kým učenie neskončí.
Na funkcii poklesu parametra učenia **α** v praxi až tak veľmi nezáleží - monotónne klesajúca funkcia **z hodnoty blízkej 1**, s malými hodnotami (**do rádovo 0,1–0,01**) - doladenie (napr. lineárna lomená funkcia, exponenciálna funkcia atď.).

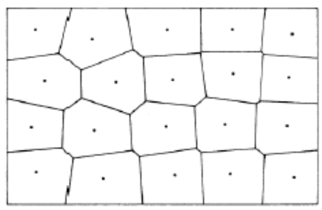


Počet iterácií: aspoň 500 * počet neurónov (Kohonen)

DP verzia (dot product, skalárny súčin) – použitá metrika – $d_i = w_i^t * x$, spočíva to v rotácii váhového vektora smerom k vstupnému vektoru. Víťaz má najmenší uhol. Váhy ležia na povrchu hypergule.

11. SOM: vektorová kvantizácia, topografické zobrazenie príznakov, redukcia dimenzie, magnifikačný faktor, náčrt matematických problémov analýzy algoritmu, príklad použitia.

SOM umožňuje realizovať zobrazenie zachovávajúce topológiu a zobraziť tak charakteristické príznaky vstupných dát. SOM je schopná dáta klasterizovať, redukovať ich dimenziu.

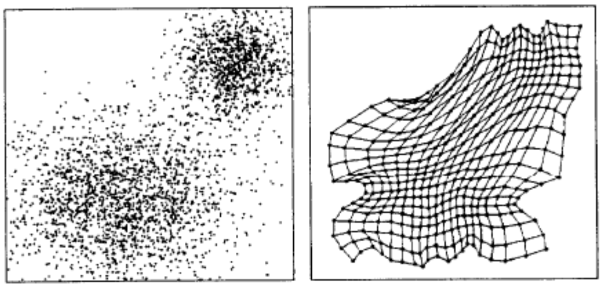


Voronoi tessellation

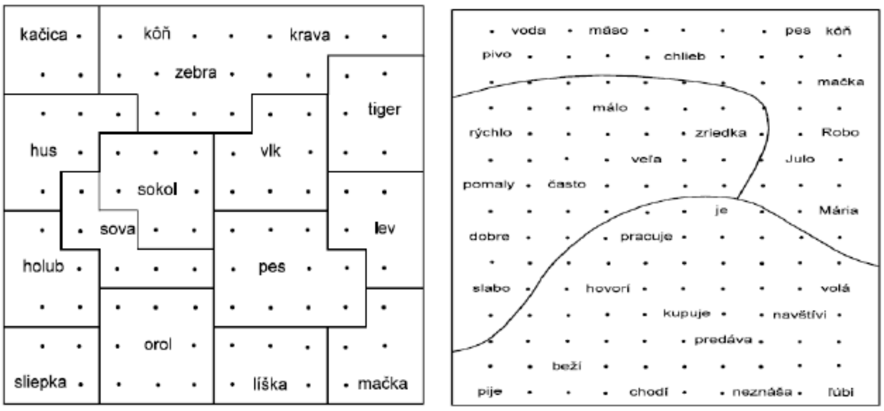
Vektorová kvantizácia – nahradenie množiny vstupných vektorov za menšiu množinu prototypov (v SOM – váhových vektorov). Vstupný priestor sa rozdelí na Voronoiov diagram tak, aby sa minimalizovala chyba rekonštrukcie. Použitie: napr. kompresia dát.

$$V_i = \{x \mid \|x-w_i\| < \|x-w_j\|, \ \forall j \neq i$$

Magnifikačný faktor – počet váhových vektorov pripadajúcich na jednotku plochy vstupného priestoru. SOM má tendenciu rovnomerne pokryť trénovacie dáta – hustejšie vstupy = hustejšie váhové vektory.



Attribute map — Contextual map



- words represented by features — - words represented by contexts

12. Hybridné modely NS, RBF model: aktivačné vzorce, bázové funkcie, príznakový priestor, problém interpolácie, aproximačné vlastnosti RBF siete.

Hybridné modely

– kombinácia učenia s učiteľom a učenia bez učiteľa,
– môže byť oveľa rýchlejšia ako gradient descent ale s podobnými výsledkami,
– fungujú správne ak podobné vstupy majú dať podobné výstupy,
– môže byť potrebných viac skrytých neurónov ako v prípade učenia s učiteľom.

RBF siete sú také, v ktorých je aktivačnou funkciou skrytých neurónov tzv. radiálna bázová funkcia (RBF) - najčastejšie sa používa gaussovská funkcia.

- Inputs $\boldsymbol{x}$, weights $\boldsymbol{w}$, outputs $\boldsymbol{y}$
- Output activation:

$$y_i = \sum_{k=1}^{q} w_{ik} h_k(x) + w_{i0}$$

- h_k = radial activ. function, e.g.

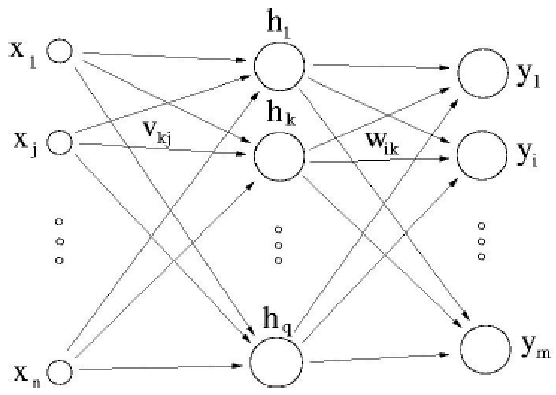
  $h_k(\boldsymbol{x}) = \varphi(x) = \exp(-\|\boldsymbol{x}-\boldsymbol{v}_k\|^2/\sigma_k^2)$

  $\boldsymbol{v}_k$ ~ center k, σ_k ~ its width

  $\varphi(d)$ are (usually) local functions because for $d \to \infty$ $\varphi(d) \to 0$

  σ affects generalization

- $\boldsymbol{v}_k$ used for approximation of unconditional probability density of input data $p(\boldsymbol{x})$



Celá sieť reprezentuje nasledovnú sumu (podľa wikipedie):

$$y(\mathbf{x}) = \sum_{i=1}^{N} w_i\, h(\|\mathbf{x} - \mathbf{x}_i\|),$$

Typy bázových funkcií

Gaussian: $\varphi(r) = \exp(-r^2/\sigma^2)$

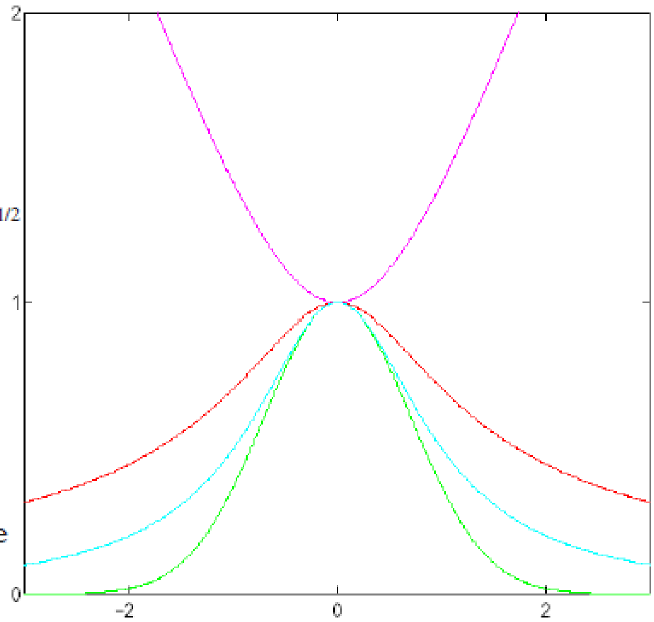
Multiquadrics: $\varphi(r) = (r^2+c^2)^{1/2}$

Inverse multiquadrics: $\varphi(r) = (r^2+c^2)^{-1/2}$

Cauchy: $\varphi(r) = 1/(1+r^2)$

$r \in \Re,\ c > 0$

RBFs that grow at infinity (multiquadrics) can be used to approximate a smooth I/O mapping with greater accuracy than those that yield positive-definite interpolation matrix (Powell, 1988).



Príznakový priestor je n-dimenzionálny priestor, kde každý vzor je bod s n súradnicami. n je počet príznakov vzoru. Podobné vzory sú blízko pri sebe.

Problém interpolácie: RBF umožňuje interpolovať zobrazenie $R^n \rightarrow R$. Interpolácia je metóda umožňujúca konštruovať nové body z konečnej množiny známych bodov. Napríklad danými bodmi preložiť polynóm. Váhy sa dajú vypočítať pomocou lineárnej algebry.

Problém: zobrazenie nemusí byť všadedefinované ani jednoznačné ani spojité. (šum..)

13. Hybridné modely NS, RBF model: spôsoby trénovania váh. Základné vlastnosti dynamických modelov NS na online aproximáciu dátových množín (TRN, DCS). Porovnanie RBF a MLP.

Hybridné modely, RBF – viď vyššie.

Trénovanie váh RBF je dvojfázový proces - najprv sa upravujú centrá a šírky, potom váhy.

Centrá: vyberú sa náhodne zo vstupov alebo klasterizáciou vstupov (napr. k-means) alebo supervised (vzdialenosť – najmenšie štvorce).

Šírky: sa vyberú podľa najväčšej vzdialenosti centier.

Váhy: zráta sa matica G ($g_{11}..g_{1n}..\ ..g_{m1}..g_{mn}$), $g_{ij} = h(|x_i - x_j|)$. $w = G^+y$

Dá sa to, lebo na rozdiel od MLP majú RBF siete jedno lokálne minimum, keď sú centrá fixnuté. Ale tiež sa to dá aj gradient descentom.

TRN = topology representing network, neorientovaný graf, po určení víťaza sa zvýši vek hrán z neho idúcich, staré hrany sa rozpoja.

DCS = dynamic cell structures, neorientovaný graf, môžu sa vkladať vrcholy. Odstraňujú sa vrcholy bez spojenia.

RBF vs. MLP: obe sú nelineárne dopredné viac-vrstvové univerzálne aproximátory,

- RBF rýchlejšie konverguje,
- reprezentácia skrytej vrstvy: MLP – globálna, RBF – lokálna => MLP potrebuje menej parametrov
- MLP – stochastický aproximačný problém,
- RBF – fitovanie hyperpovrchu vo vysoko-dimenzionálnom priestore,
- MLP – jednofázové; RBF – dvojfázové trénovanie.

14. Rekurentné NS: spôsoby reprezentácie času, typy úloh pre rekurentné NS. Modely s časovým oknom do minulosti, výhody a nedostatky, príklad použitia.

V trénovacej množine môžu byť pre rovnaké vstupy rôzne výstupy – menia sa v čase.

Reprezentácia času

- *tapped-delay input* – čas ako priestorový rozmer – „okno do minulosti“,
- *rekurentná architektúra*
  - dočasné vstupno-výstupné mapovanie,
  - asociatívna pamäť.

Typy úloh

- klasifikácia sekvencií ,
- predikcia sekvencií,
- generovanie sekvencií.

Nevýhody - spotrebujú veľa pamäti; nemajú spätnú väzbu.

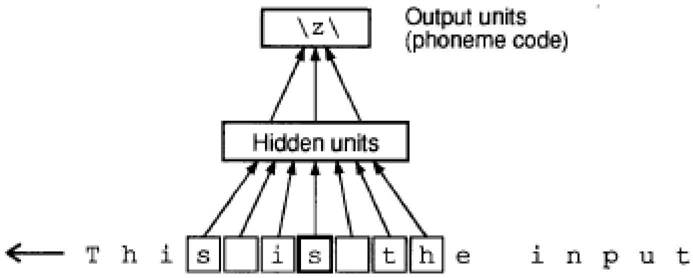
Príklady použitia – predikcia a modelovanie časových radov; odstraňovanie šumu; rozpoznávanie hlasu.

Okno do minulosti – reprezentuje kontext - vstupom NS je postupnosť niekoľkých reálnych vstupov tak ako za sebou nasledujú v čase. Napr. niekoľko po sebe nasledujúcich písmen pri ‚čítaní anglického textu‘.

Modely využívajúce okno do minulosti

NETtalk: čas ako priestorový rozmer

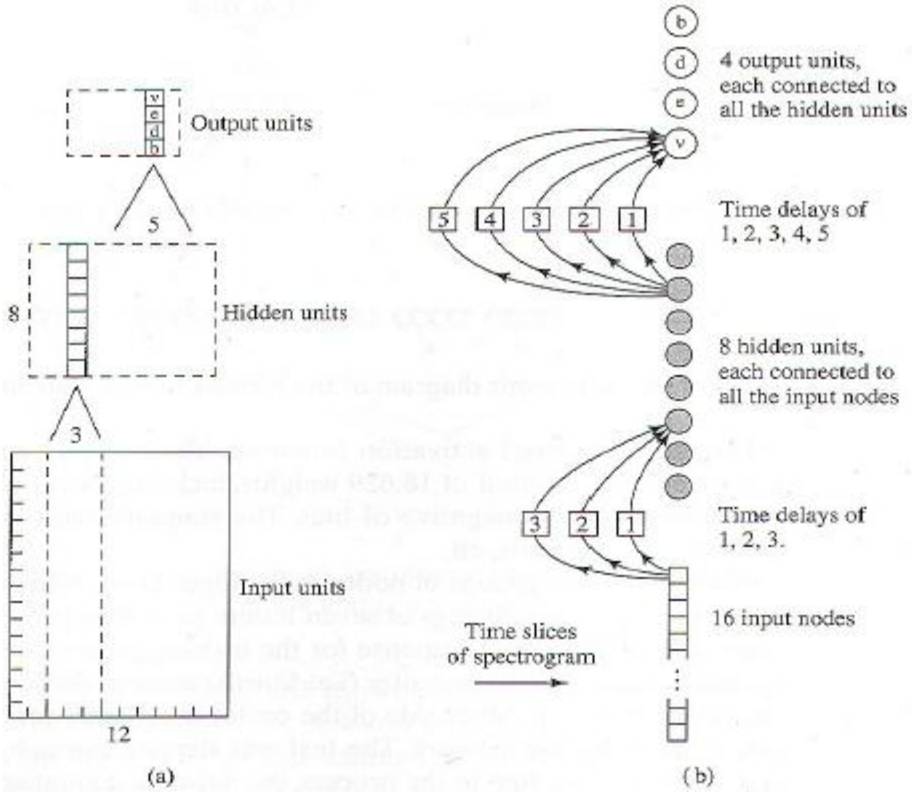
Čítanie anglického textu. Vstup = 7 × 29 neurónov (7 znakov), 80 skrytých a 26 výstupných (fonémy).



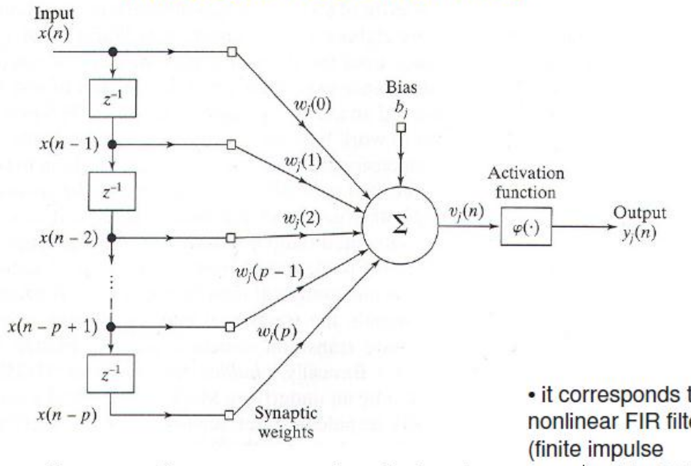
Time-Delay Neural Network

Vstup: spektogram (16 × 12).
Skrytá vrstva: 10 kópií 8 neurónov.
Výstup: 6 kópií 4 jednotiek

Trénuje sa ako dopredná NS.
Nevhodná pre úlohy vyžadujúce dlhú pamäť.



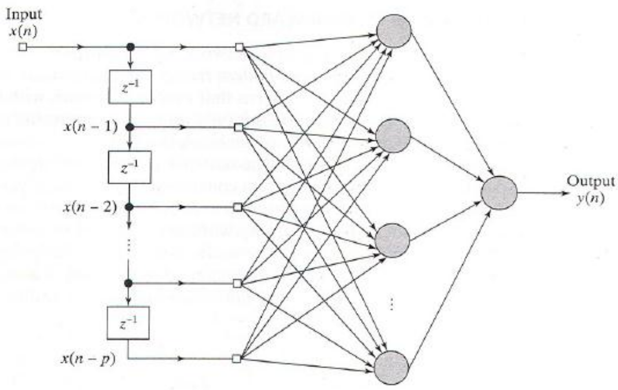
Focused neuronal filter



• it corresponds to nonlinear FIR filter (finite impulse response) in DSP

• focused because all memory comes from the input
• can be trained as ordinary feedforward NN

Focused time feedforward NN



• usable for stationary input-output mapping tasks
• can be trained as ordinary feedforward NN

Od tohto miesta nespracované...

15. Rekurentné NS: opis architektúry a princíp trénovania RNN pomocou algoritmu BPTT a pomocou RTRL. Príklad použitia.

Aktiváciu majú rovnakú ako MLP: S_i^(t+1) = g(□w_ij*S_j^t + I_i^t) – I je vstup.

BPTT: Chyba sa šíri po časovo rozvinutej sieti takisto ako v BP. Veľké pamäťové nároky. Moc sa neujala.

RTRL: Netreba vedieť dopredu dĺžku postupnosti, môže sa meniť a netreba alokovať toľko pamäti. Výpočtovo náročnejšia. Požadovaná odpoveď v čase T+1 je O. Chyba k-teho neurónu v čase t=T+1 je E_k^t = O_k – S_k^t.

V čase t z {1..T} je E_k^t = 0. Celková chyba je E^t = 1/2{□(E_k^t)^2}. Váhy sa updatujú ako v MLP. delta w_ij^t = □□(E_k^t*deriv.S_k^t podla w_ij).

16. Elmanova sieť: interné reprezentácie pri symbolovej dynamike, markovovské správanie, architektúrny bias, vzťah k IFS, sieť s echo stavmi architektúra, trénovanie.

Elman: predikuje postupnosti z kontextu, rozpoznáva, dopĺňa, má hierarchickú reprezentáciu. Stavový priestor siete má fraktálovú reprezentáciu. Trénovať sa môže BP bez rekurentných váh, alebo BPTT alebo RTRL.

IFS – množina transformácií. Príklad – žaba v sierinskeho trojuholníku. Postupnosť transformácií je adresa v priestore. Dve postupnosti ak majú dlhý spoločný suffix, sú blízko seba.

Markovovská vlastnosť: budúce stavy sú nezávislé na minulých stavoch.

Architektúrny bias: je fenomén – štruktúra klastrov odráža históriu vstupov.

17. Hopfieldov model NS: deterministická verzia, typy dynamiky modelu, energia systému, relaxácia, možné typy atraktorov, autoasociatívna pamäť – nastavenie váh, kapacita pamäte.

Každý neurón je v jednom z dvoch stavov S_i □{-1,+1}. Spojené sú každý s každým. Váha synapsy J_ij. Postsynaptický potenciál h_i^int=□_j=1^N{J_ij*S_j}. Ak prekročí prah excitácie h_i^ext, neuróny sa aktivuje. S_i = sgn(h_i^int – h_i^ext).

Paralelná dynamika: Všetky menia svoj stav naraz. Najprv sa zrátajú zmeny, potom sa aplikujú.

Sekvenčná dynamika: Vždy sa mení len jeden náhodne vybratý neurón. Zodpovedá to pohybu po hranách hyperkocky.

Energia systému: E(S)=-1/2(□J_ij*S_i*S_j) - □S_i*h_i^ext.

Atraktory: pravé/falošné

Správanie: chaotické trajektórie(synchrónna dyn., asymetrická J), limitné cykly (pri synch. dyn.), bodové atraktory. (energia klesá, kým nedosiahne minimum, pri async. dyn.)

Autoasociátor: nastavenie váh: J_ij=□x_i^p*x_j^p pre jednotlivé pamäťové vzory p. Pri relaxácii keď sa dostane do nejakého vzoru (S_i=x_i^p) (recall), podmienka stability: 1+C_i^p = x_i^p*h_i^p>0.C_i^p je presluch, rovná sa nule, ak sú pamäťové vzory ortogonálne. Pamäťová kapacita sa potom blíži k počtu neurónov.

18. Hopfieldov model NS: stochastická verzia, opis dynamiky, parameter inverznej teploty, odstránenie falošných atraktorov. Porovnanie s deterministickou verziou. Typy úloh pre použitie Hopfieldovho modelu.

Každý neurón je v jednom z dvoch stavov S_i □{-1,+1}. Spojené sú každý s každým. Váha synapsyJ_ij. Postsynaptický potenciál h_i^int=□_j=1^N{J_ij*S_j}. Ak prekročí prah excitácie h_i^int, neurón i sa aktivuje.

Sú tu reverzné atraktory, ale tie nás netrápia. Chceme odstrániť zmiešané atraktory. Pridáme šum(teplotu). Každý zmiešaný stav má kritickú teplotu do 0.46, od ktorej už nie je stabilný.

Použitie: rozoznávanie vzorov(písmo, tváre..), generovanie, rozpoznávanie postupností(reč, zvuky, video..), autoasociácia, klasifikácia, rekonštrukcia, optimalizačné problémy, obch. cestujúci, párovanie,