Neurónové siete

Peter Kovács, Marián Margeta, Jozef Brandys, Jakub Pospíchal

Contents

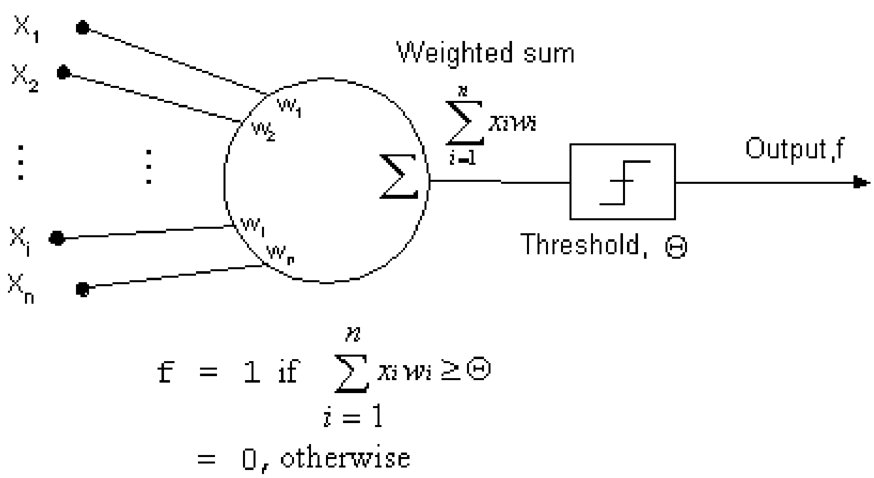
1 Dopredné modely. Otázky 1 až 7.

1.1 Stručná história konekcionizmu, vlastnosti biologického neurónu, model neurónu s prahovou logikou, implementácia Booleových funkcií. Paradigmy učenia a typy úloh pre NS.

Stručná história konekcionizmu sa začína v 40tych rokoch v psychologii a filozofii. McCullogh a Pitts vymyslia neuron s aktivacnym prahom. Neskor v 60. až 70. rokoch sa k ich nápadu vyjadrí Minsky, zrozumitelnejšie to popíše a dá to do kontextu s teóriou formálnych jazykov a automatov. V 90. rokoch prichádzajú na výslnie viacvrstvové generatívne modely a od roku 2000 prichádza druhá renesancia - deep learning, rekurentné a konvolučné neurónové siete a tiež siete s echo stavmi.

Nervová bunka sa skladá z tela a niekoľkých výbežkov. Tieto možno rozdeliť na dva typy" dendrity, ktoré predstavujú z informatického hľadiska vstupnú časť (predovšetkým na ne prechádza vzruch z iných buniek) a jeden axón, po ktorom sa vzruch šíri k iným bunkám.[Uvod do teórie neurónových sietí.]

Neurón s prahovou logikou vyzerá nasledovne,



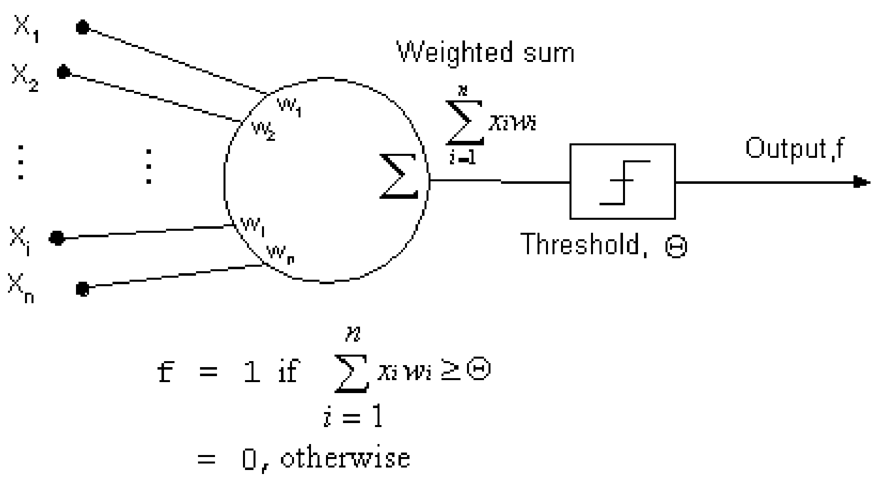
v prípade, že by sme pomocou neho chceli implementovať booleovské funkcie, napríklad logický AND jednoducho pre každú premennú použijeme jeden vstup a threshold bude rovný počtu premenných. V prípade OR by stačilo aby threshold bol 1. Ešte by bolo vhodné podotknúť, že každá booleovská funkcia môže byť simulovaná pomocou dvojvrstvovej NN s logickými jednotkami.

Medzi paradigmy učenia v neurónových sietiach patrí určite učenie s učiteľom a učenie bez učiteľa. Ako príklad si uveďme cenu domov v Bratislave. Vstupom sú dáta typu rozloha/cena/počet izieb a výstupom cena. V prípade učenia s učiteľom máme dostupnú cenu ktorú chceme predikovať. V prípade učenia bez učiteľa máme k dispozícii iba prvé tri vstupy a na základe nich môžeme byty zhlukovať do kategórií veľký/malý etc. Poslednou paradigmou učenia je učenie posilnovaním, ktoré funguje na základe nejakej funkcie odmeny. Po každej akcii sa posunieme do nového stavu a prostredie nám poskytne informáciu či to čo sme spravili je správne alebo nie.

V dnešnej dobe sa neurónové siete používajú na kadečo významné úspechy dosiahli v oblasti spracovania obrazu(konvolučné siete), spracovaní prirodzeného jazyka, zvuku alebo iných sekvenčných dát(rekurentné)...

1.2 Binárny perceptrón: pojem učenia s učiteľom, učiace pravidlo, algoritmus trénovania, deliaca nadrovina, klasifikácia vzorov, lineárna separovateľnosť, náčrt dôkazu konvergencie, definícia a príklad.

Binarny perceptron === dáva nám output 1/0 [??]



Učiace pravidlo updatuje perceptrón na základe toho aký výstup nám vypluje a aký bol požadovaný výstup.

$$w_j(t+1) = w_j = \alpha(d-y)x_j$$

kde w_j je váha j-teho vstupu α je rýchlosť učenia, d je očakývaný výstup, y je výstup perceptrónu a x_j je j-ty vstup.

Algoritmus trénovania je nasledovný:

1. Zvoľ vstup x a vypočítaj výstup y.

2. Spočítaj chybovú funkciu $e(t) = 1/2(d-y)^2$ a pripočítaj k celkovej chybe $E := E + e(t)$.

3. uprav všetky váhy na základe učiaceho pravidla (ak e(t) $>$ 0),
4. ak som použil všetky trénovacie vstupy goto 5. inak goto 1.
5. ak $E = 0$ (setky patterny su spravne zaklasifikovane) skonči inak poprehadzuj vstup $E := 0$ a začni od 1.

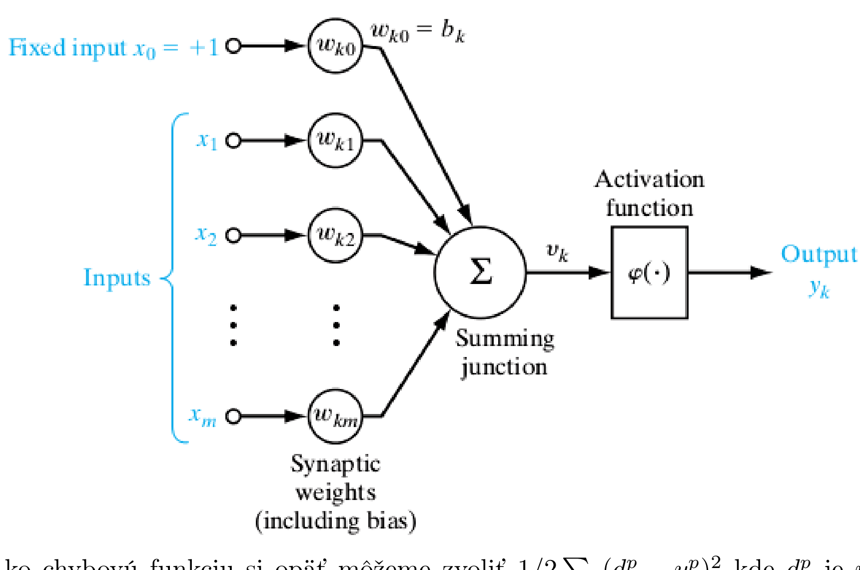
To čo vlastne perceptrón spraví je, že rozdelí(klasifikuje) vstupy do dvoch tried, tie ktoré ho aktivujú a tie ktoré nie. Vo všeobecnosti perceptrón len hľadá nejakú deliacu nadrovinu, ktorú vieme zapísať v tvare $\sum_{i=1}^{n} w_i x_i = \theta$.

V roku 1962 Rosenblatt sformuloval vetu: Nech triedy A a B sú lineárne separoveteľne(existuje nadrovina, ktorá správne oddeli jednu triedu od druhej) potom perceptrón konverguje, tj. nájde deliacu nadrovinu, ktorá rozdelí tieto dáta do dvoch množín.

Dôkaz: Nech $\alpha = 1$ ... TODO

1.3 Spojitý perceptrón: Rôzne aktivačné funkcie perceptrónu, chybová funkcia a spôsob jej minimalizácie, učiace pravidlo, algoritmus trénovania perceptrónu. Súvis s Bayesovským klasifikátorom.

Na rozdiel od prahového perceptrónu už nebudeme mať aktivačnú funkciu signum ale použijemen rôzne spojité napríklad sigmoidu alebo tangens hyperbolický. Sigmoida $\frac{1}{1+e^{-x}}$ nám dáva výstupy v intervale $[0, 1]$ a tanh $[-1, 1]$.



Ako chybovú funkciu si opäť môžeme zvoliť $1/2 \sum_p (d^p - y^p)^2$ kde d^p je p-ty očakávaný výstup. Aby naše výsledky boli čo najpresnejšie chceme chybovú funkciu minimalizovať. Na to používame algoritmus gradient descent, ktorý funguje tak, že nájdeme deriváciu chybovej funkcie a v smere proti gradientu budeme meniť váhy tak, aby sme sa dostali na gradient rovný 0. Gradient descent má viacero spôsobov ako minimalizuje funkciu.

1. Stochastic gradient descent - po každom videnom príklade spravím update parametrov - $w_j(t+1) = w_j(t) + \alpha(d^p - y^p)f'x_j = w_j(t) + \alpha\delta x_j$.

2. Batch gradient descent - prejdeme cez všetky trénovacie príklady spočítame chyby a až potom spravíme update $w_j(t+1) = w_j(t) + \alpha \sum_p \delta x_j$.

Pri stochastickej verzii sise skonvergujeme ale nemusíme sa dostať až do úplného minimia ale budeme niekde okolo neho poskakovať. Pri batch verzii robíme najstrmší krok v chybovom priestore a zmenšujeme chybu ako sa len dá, no platíme za to dlším časom počítania.

Ako dalšie často používané chybové funkcie môžeme spomenúť cross-entropy chybovú funkciu $-\sum_p[d^{(p)}ln(y^{(p)}) + (1-d^{(p)})ln(1-y^{(p)})]$, ktorú keď minimalizujeme dostaneme opäť rovnaké učiace pravidlo ako pri squared error. Táto funkcia nám vlastne povie s akou pravdepodbnosťou príklad patrí do triedy 1 alebo 0.

Tato chybová funkcia je vhodná pri binárnej klasifikácii. Ďaľšou funkciou je softmax $y_i = \frac{exp(net_i)}{\sum_j exp(net_j)}$, ktorá je vhodná napríklad pri klsifikácii do viacerých tried. Potom nám vlastne hovorí s akou pravdepodobnosťou sample patrí do ktorej triedy.

TODO súvis s bayes klasifikatorom.

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

TODO

1.5 Viacvrstvová dopredná NS ako univerzálny aproximátor funkcií (formulácia teorému), trénovacia a testovacia množina, generalizácia, preučenie, skoré zastavenie učenia, selekcia modelu, validácia modelu. Hlboké učenie NS.

TODO

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

Nech máme trénovaciu množinu $A_{train} = \{(x^{(p)}, y^{(p)}), p = 1, \cdots, N\}$ a hľadáme maticu W, ktorá spĺňa

$$y^{(p)} = Wx^{(p)}, \forall p$$

V maticovej notácii

$$Y = WX$$

potom riešenie systému vieme jednoducho nájsť tým, že prenásobíme takýto systém inverznou maticou k matici X zprava a dostaneme teda

$$YX^{-1} = W.$$

Probém je v hľadaní inverznej matice k matici X pretože táto matica nemusí byť regulárna. Z tohto dôvodu sa zaviedol pojem pseudoinverznej(Moore-Penrose) matice, ktorú označujeme X^+. A má nasledovné vlastnosti ($\forall X \exists X^+$)

1. $XX^+X = X$
2. $X^+XX^+ = X^+$
3. X^+X a XX^+ sú symetrické

Vypočítat ich potom môžeme nasledovne

1. $X^+ = X^T(XX^T)^{-1}$ ak n < N a hodnost(X) = n.
2. $X^+ = (X^TX)^{-1}X^T$ ak n > N a hodnost(X) = N.

kde N je počet príkladov a n je dimenzia vstupu.

Chcely by sme natrénovať model X = WX, $N < n$ a chceme aby model vedel rekonštruovať N vstupov. Takýto model voláme lineárny autoasociátor. V prípade, že $N = n$ by sme dostali triviálne riešenie $W = I$, to ale nie je to čo chceme. Keď dostaneme zašumený vstup tak chceme odpovedať zapamätaným vzorom.

Linear manifold $L = \{x \in R | x = a_1x_1 + a_2x_2 + \cdots + a_Nx_N, a_p \neq 0\}, L \subset R^n$
Ortogonálny komplement = $L^\perp = \{x \in R | x \perp L\}$

Každý vektor z X vieme jednoznačne rozložiť $x = x_{obal} + x_{orto}$ kde $x_{obal} \in L, x_{orto} \in L^\perp$.Tréningová množina $X = \{x_1, x_2, \cdots, x_n\}$ bude tvoriť náš lineárny obal L. Teraz keď dostaneme ľubovoľný vstup x predpokladáme, že je zašumený, ale keďže ho vieme rozložiť tak nám stačí spraviť ortogonálnu projekciu $Wx = (XX^+)x = x_p$, čím dostaneme pattern ktorý je najbližšie danému vektoru. V prípade, že by sme spočítali $Wx = (I - XX^+)x = x_p$ potom $x_p \in L^\perp$, toto voláme detektor novosti.

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

TODO

2 Samoorganizácia a RBF sieť. Otázky 8 až 12.

2.1 Samoorganizácia v NS, základné princípy, pojem učenia bez učiteľa, typy úloh použitia, Ojovo pravidlo učenia pre jeden neurón, vysvetlenie konvergencie.

TODO

2.2 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.

TODO

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

TODO

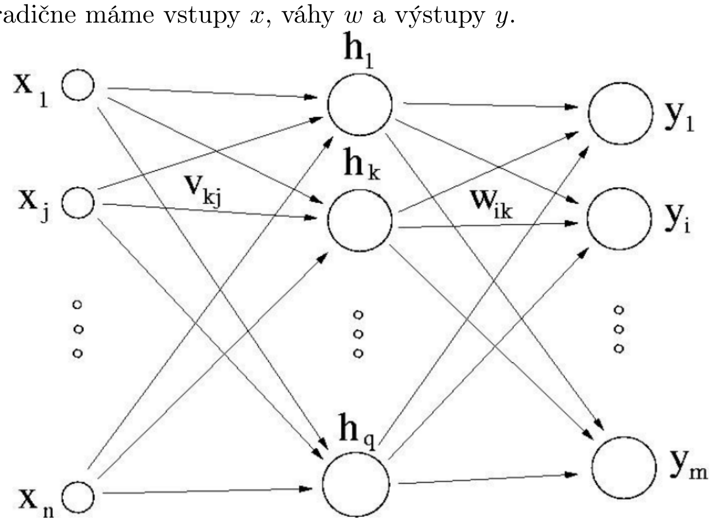
2.4 SOM: vektorová kvantizácia, topografické zobrazenie príznakov, algoritmus SOM, parametre, redukcia dimenzie, magnifikačná vlastnosť, príklad použitia.

TODO

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

Tento model je istou kombináciou učenia s učiteľom a bez učiteľa. Funguje to veľmi dobre pokiaľ pre podobné vstupy očakávame podobné výstupy. Väčšinou vyžadujeme viac neurónov ako pri ostatných modeloch učených iba s učiteľom. Konkrétne sa pozrieme na model Radial Basis function neural network.

Tradične máme vstupy x, váhy w a výstupy y.



Výstupné aktivácie budú teda podla schémy

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

, kde h_k je radiálna aktivačná funkcia, ako napríklad $exp(-\frac{|x-v_k|^2}{\sigma_k^2})$, v_k je centrum k a σ_k je rozsah.

TODO

3 Rekurentné a pamäťové modely. Otázky 13 až 18.

3.1 NS na spracovanie sekvenčných dát: reprezentácia času, typy úloh pre rekurentné NS. Modely s časovým oknom do minulosti, výhody a nedostatky, príklad použitia.

TODO

3.2 Rekurentné NS: princíp trénovania pomocou algoritmu BPTT a RTRL. Príklad použitia.

TODO

3.3 Elmanova sieť: interné reprezentácie pri symbolovej dynamike, Markovovské správanie, architekturálna predispozícia. Model rekurzívnej SOM (RecSOM).

TODO

3.4 Sieť s echo stavmi (ESN): architektúra, inicializácia, trénovanie modelu, vplyv parametrov na vlastnosti rezervoára, echo vlastnosť, pamäťová kapacita.

TODO

3.5 Hopfieldov model NS: deterministická dynamika, energia systému, relaxácia, typy atraktorov, autoasociatívna pamäť – nastavenie váh, princíp výpočtu kapacity pamäte.

Hopfieldov model neurónovej siete vyzerá tak, že má jednu vrstvu, kde každý neurón je prepojený s ostatnými. Každý z neurónov nadobúda stav $S_i = \{-1, 1\}, i = 1, \ldots, n$ a má váhy J_{ij}. Ak $J_{ij} > 0$ nazývame ju excitačná inak inhibičná a definitoricky váha $J_{ii} = 0$.

V prípade, že chceme spočítať update váh najprv musíme spočítať:

1. Postsynapticky potencial - $h_i^int = \sum J_{ij} S_j$.
2. Excitačná hranica(threshold) h_i^{ext}.
3. Efektívny postsynaptický potenciál $h_i = h_i^{int} - h_i^{ext}$.
4. Neuron state update(deterministic) $S_i \leftarrow sgn(h_i) \in \{-1, 1\}$ ak $h_i = 0$ potom $sgn(h_i) = 1$.

Update môže prebiehať buď synchrónne(všetky naraz) alebo asynchróne(randomne po jednom). Je fajn si uvedomiť, že v prípade synchrónneho updatu sa hýbeme po vrcholoch hyperkocky ale v prípade asynchrónneho po jej hranách.

V každom stave siete vieme vypočítať takzvanú energiu siete $E = -\frac{1}{2} \sum_i \sum_j J_{ij} S_i S_j - \sum_i S_i h_i^{ext}$

Dá sa ukázať, že ak používame asynchrónny update s excitačným thresholdom $h_i^{ext} = 0, \forall i$ a symetrickou konektivitou neurónov $J = J^T$ energia vždycky klesá počas relaxácie(updatovania neurónov).

Atraktory delíme na pravé a falošné(lineárna kombinácia zapamätaných vzorov).

V autoasociatívnej pamäti chceme nastaviť váhy tak, aby bodovými atraktormi boli naše zapamätané patterny. Predpokladajme, že máme binárne patterny $x^p = [x_1^{(p)}, \dots, x_n^{(p)}]$ $p = 1, \dots, N$, potom váhy nastavíme podľa nasledovného vzťahu:

$$J_{ij} = \frac{1}{n} \sum_p x_i^{(p)} x_j^{(p)} \text{ for } i \neq j \text{ and } J_{ii} = 0$$

Keď sieť relaxujeme z $S(0) \to \cdots \to x_i^{(r)}$ potom podmienka stability pre vzor $x_i^{(r)}$ je nasledovná

$$x_i^{(r)} \times h_i^{(r)} = x_i^{(r)} \sum_j J_{ij} x_i^{(r)} = \cdots = 1 + C_i^{(r)} > 0$$

, kde $C_i^{(r)}$ voláme crosstalk(šum). Ak sú vzory na seba navzájom kolmé potom šum je $C_i^{(r)} = 0$ a kapacita pamäte je rovná počtu vzorov.

Kapacitu pamäte môžeme merať pomocou toho, že zistíme aká je pravdepodobnosť, že i-ty neurón je nestabilný. $P_{error} = P(C_i^{(r)} > 1)$. Táto pravdepodobnosť priamo závisí od počtu patternov a počtu neurónov. $C_i^{(r)} \sim Bin(0, \sigma^2)$ kde $\sigma^2 = N/n$. Z vety o centrálnej limite potom platí, že $Bin \approx N(0, \sigma^2)$.

3.6 Nelineárne dynamické systémy: stavový portrét, dynamika, typy atraktorov. Hopfieldov model NS: stochastická dynamika, parameter inverznej teploty, princíp odstránenia falošných atraktorov.