Neurónové siete
Inžiniersky prístup
(1. diel)

Peter Sinčák
Katedra kybernetiky a umelej inteligencie
Elektrotechnická fakulta
Technická Univerzita Košice
e-mail: sincak@ccsun.tuke.sk
&
Gabriela Andrejková
Katedra matematickej informatiky
Prírodovedecká fakulta
Univerzita P.J. Šafárika, Košice
e-mail: andrejk@kosice.upjs.sk

22. júla 1996

Predhovor

Predkladáme čitateľom **2–dielnu** publikáciu venovanú inžinierskym aspektom **neurónových sietí**. Cieľom bola snaha podať **základné teoretické poznatky** o neurónových sieťach a ich metódach učenia, ktoré je možné využiť v rôznych aplikačných oblastiach technickej praxe.

Neurónové siete vplyvom rýchleho rozvoja výpočtovej techniky sa využívajú čoraz viac v experimetálnych úlohach ako aj v praxi. Aplikačné úlohy typu **spracovania informácii, klasifikácie vzorov alebo situácií, optimalizačné problémy, predikčné úlohy** nachádzajú uplatnenie v rôznych oblastiach **priemyslu, managementu, finančníctva, telekomunikácie, vojenskej techniky, zdravotníctva a inde**.

Prvý diel je venovaný základným informáciam o **dopredných neurónových sieťach** a **druhý diel** je venovaný **rekurentným a modulárnym neurónovým sieťam**.

Dúfame, že kniha pomôže šíreniu základných poznatkov o neurónových sieťach z pohľadu inžinierskeho chápania ich činnosti a využitia v technickej praxi.

Autori

Poďakovanie

Týmto vyjadrujeme **verejné poďakovanie** študentom 4. a 5. ročníka FEI Katedry kybernetiky a umelej inteligencie TU Košice (šk. roku 1995–1996) a študentom Katedry matematickej informatiky Prirodovedeckej fakulty za **morálnu podporu a inšpirácie** pri príprave tejto publikácie. Nekonečné diskusie a **nadšenie študentov** pre neurónové siete nám pomáhalo pri príprave a dokončovaní tejto knihy.

Touto cestou ďakujeme Ing. Rudolfovi Jakšovi za dokončovacie práce pri príprave knihy, Ing. Marekovi Hatalovi za pomoc pri konečnej úprave a počítačovej sadzbe a Editke Šutakovej za nakreslenie obrázkov. Súčasne ďakujeme Prof. Ing. Jánovi Sarnovskému, CSc., vedúcemu Katedry kybernetiky a umelej inteligencie, ktorý vytvoril podmienky pre rozvoj tejto disciplíny na TU Košice.

Záverom ďakujeme všetkým tým, ktorí **nám držali palce**, aby táto publikácia bola konečne dokončená.

Autori

Venovanie

Túto knihu venujem ľuďom, ktorí mi veľa pomohli a to:

- mojim rodičom, ktorým vďačím za veľa
- mojej rodine, ktorá mi dáva zmysel života
- mojim študentom, od ktorých som sa veľa naučil a ešte veľa naučím

Peter Sinčák

Obsah

1 Úvodné poznámky **1**
1.1 Moderné aspekty umelej inteligencie 1
1.2 Čo je to neurónová sieť (NN) 3
1.3 Stručne o histórii NN 5
1.4 Typy úloh riešiteľných pomocou NN 8
1.5 Pedagogické poznámky 8

2 Základné pojmy **11**
2.1 Základné pojmy teórie učenia 12
2.2 Základné prvky NN 13
2.2.1 Poznámky k jednotlivým častiam neurónu 15
2.2.2 Synaptické spojenia a váhy 17
2.3 Topológia NN a spôsoby šírenia signálu 18
2.4 Perceptrón – najjednoduchšia neurónová sieť 20
2.4.1 Čo je to perceptrón ? 21
2.4.2 Topológia perceptrónu 22
2.4.3 Algoritmus učenia perceptróna 23
2.4.4 Veta o konvergencii perceptrónu 25
2.4.5 XOR-problém, skrytá vrstva NN 27

3 Učenie a jeho paradigmy **31**
3.1 Paradigmy kontrolovaného učenia 32
3.2 Paradigmy nekontrolovaného učenia 34
3.3 Význam inicializácie pri učení NN 37
3.4 Globálna stabilita a konvergencia NN 37
3.4.1 Globálna stabilita NN 37
3.4.2 Konvergencia NN 39

4 Kontrolované učenie na FF NN **41**
4.1 Metóda najstrmšieho zostupu 41
4.1.1 Wienerov filter 41
4.1.2 Metóda najstrmšieho zostupu 43
4.1.3 Metóda nejmenšej kvadratickej chyby 44
4.1.4 Adaline 45
4.2 Repetitórium č. 1 47
4.3 Delta pravidlo 51
4.4 Metóda spätného šírenia chyby 52
4.5 Time-delay na FF NN 56
4.6 Spôsoby urýchlenia konvergencie BP 58
4.6.1 BP-momentum 58
4.6.2 Adaptívne parametre učenia NN 58
4.7 Funkcionálne linky v NN 66
4.8 Dôležité poznámky k návrhu FF NN 69
4.8.1 Návrh topológie NN 69
4.8.2 Problém inicializácie NN 71
4.8.3 Problém stanovenia veľkosti trénovacej vzorky . . 72
4.9 Repetitórium č. 2 73

5 Nekontrolované učenie na FF NN **77**
5.1 Konkurenčné učenie 78
5.1.1 MAXNET 83
5.2 Kohonenove siete 83
5.3 Metóda hlavných komponentov 89
5.3.1 Ojove adaptačné pravidlo zmeny SV 89

6 Hybridné metódy učenia na FF NN **95**
6.1 Nekontrolované učenie metódy BP 95
6.2 Metóda Counterpropagation 96
6.3 Repetitórium č. 3 100

Literatúra **103**

Register **105**

Používané skratky

znak	význam	definované na strane
NN	neurónová sieť (neural network)	3
SV	synaptická váha	13
FF	dopredná (feed–forward)	19
RC	rekurentná (recurent)	19
BP	metóda spätného šírenia chyby (Error Backpropagation)	52
in_i	vstup do neurónu "i"	13
x_i	aktivačná hodnota neurónu "i"	13
ou_i	výstupná hodnota neurónu "i"	13
θ_i	prah neurónu "i"	13
J(t)	chybová funkcia	51
ev_i	očakávaná hodnota aktivácie neurónu "i"	32
γ	učiaci pomer	32
w_{ij}	hodnota SV smerujúcej od neurónu "j" k neurónu "i"	14
e_i	chybový rozdiel $e_i = ev_i - x_i$	32
δ_i	chybový signál pri BP učení	52
α	momentum parameter pri BP momentum učení	58
Λ	funkcia susednosti pri Kohonenových NS	84

1

Úvodné poznámky

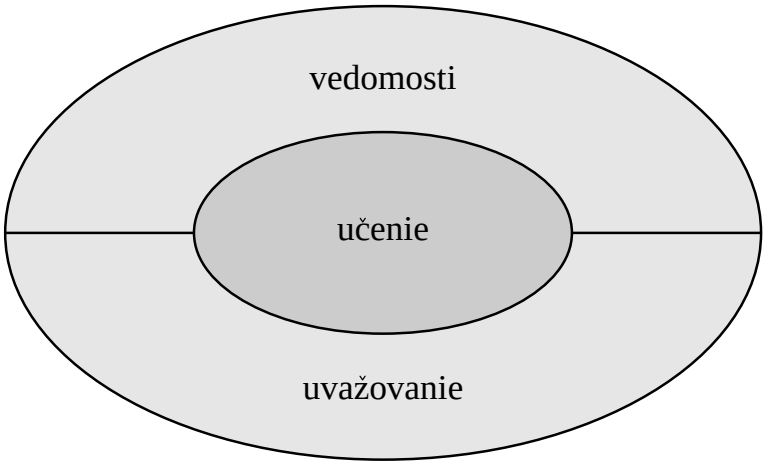
1.1 Moderné aspekty umelej inteligencie

Nový pohľad na paralelné výpočty a na paralelné systémy v súčasnosti prináša najnovší trend, ktorý zaujal mnohých vedcov a odborníkov, a síce **quasiencefalické výpočty**, v ktorých ide o masívny paralelizmus a majú za cieľ modelovať chovanie sa nervovej sústavy u živočíchov a hlavne modelovať chovanie sa ľudského mozgu. Ľudský mozog obsahuje $10^{11} - 10^{14}$ neurónov uložených v šedej kôre mozgovej a synapsie sú realizované v rozsahu asi 10^4 na jeden neurón. Vytvorenie umelého ľudského mozgu so všetkými jeho schopnosťami je vec veľmi ťažko riešiteľná aj z hľadiska kvantity neurónov aj z hľadiska spôsobu ich prepojenia, atď. Je tu však možnosť simulovať aspoň niektoré funkcie ľudského myslenia a implementovať ich. Novovytvárané modely sa o túto situáciu pokúšajú, a preto dostali názov **neurónové siete** (NN - Neural Nets).

V tejto súvislosti vzniká tiež otázka vzťahu neurónových sietí k systémom umelej inteligencie (UI). Pokúsime sa na ňu odpovedať v nasledujúcej úvahe o systémoch umelej inteligencie a ich možných cieľoch.

> **Cieľom systémov umelej inteligencie je vypracovať paradigmy alebo algoritmy, ktoré požadujú od stroja riešiť úlohy, ktoré by vyriešil len človek so znalosťami**

Toto je jedna z mnohých definícií systémov UI [5]. Každý systém



Obr. 1.1: Hlavné komponenty všeobecného systému UI

považovaný za systém UI, musí podľa spomínaného autora splňať 3 nasledujúce požiadavky :

- vedieť uložiť znalosti (knowledge representation)
- aplikovať znalosti pre riešenie problému - uvažovanie (reasoning)
- získavať nové znalosti počas experimentov - učenie (learning)

Všetky 3 požiadavky navzájom súvisia a doplňujú sa. V súčasnej dobe sa systémy UI čoraz väčšmi chápu z dvoch pohľadov :

- **klasické prístupy** v UI (tieto presadzujú symbolickú reprezentáciu vedomostí a ich sekvenčné spracovanie, nachádzajú uplatnenie v problémoch ako je spracovanie prirodzeného jazyka, v problémoch plánovania procesov alebo tzv. explicitného reasoningu). Exemplárnym problémom, kedy klasická UI (symbolická UI) nestačí riešiť všeobecné problémy je Japonský projekt počítačov 5. generácie. Tam logické programovanie a expertné systémy nepriniesli očakávané výsledky.

- **moderné prístupy** v UI (tieto sú založené vo veľkej miere na neurónových sieťach vo väčšine prípadov používajú nesymbolickú reprezentáciu vedomostí a sú vhodne použiteľné v oblastiach ropoznávania vzorov, simulácie pamäte resp. v tzv. nizkoúrovňových percepčných procesoch. Tieto moderné prístupy predstavujú systém UI ako dynamický systém, a preto je nutné ho analyzovať z hľadiska jeho neurodynamiky. V náväznosti na spomínaný Japonský projekt, sa tu ponúka možnosť využitia neurónových sietí aj pre tieto problémy.

Je treba poznamenať, že vo svete je skupina vedcov, ktorí tieto moderné prístupy v UI považujú za nepatriace do UI a na UI pozerajú iba z hľadiska klasického prístupu k UI. Čas však ukazuje že medzinárodná komunita akceptovala tieto moderné prístupy k UI. Cesta k progresu vedie cez využívanie vlastností a výhod **jak klasickej UI tak aj modernej UI**. Explózia záujmu o NN dávno prekročila rozmery niekdajšieho záujmu napr. o expertné systémy. V momentálnej situácii je záujem o NN podporený aj vynikajúcimi možnosťami v oblasti výpočtovej techniky, paralelných systémov a pamäťových možností a širokú odbornú podporu ostatných odborov ako je neurofyzika, neuropsychológia, kognitívna psychológia a pod. . V podstate sa do určitej miery napĺňajú predpovede jedného z významných predstaviteľov rozvoja NN dr.Minského, ktorý v roku 1961 v svojom článku **Step Torward Artificial Intelligence**[1] naznačil mnohé smery, ktoré z nedostatku výpočtových možností nemohli dostať plný rozmer rozvoja. Záverom je treba konštatovať, že tendencie moderných prístupov umelej inteligencie nájdu čoraz väčšie uplatnenie pri riešení praktických problémov technickej a netechnickej praxe.

1.2 Čo je to neurónová sieť (NN)

Neurónová sieť (ďalej NN) je masívne paralelný procesor, ktorý má sklon k uchovávaniu experimentálnych znalostí a ich ďalšieho využívania. Napodobňuje ľudský mozog v dvoch aspektoch:

[1] Krok vpred umelá inteligencia

- **poznatky sú zbierané v NN počas učenia**
- **medzineurónové spojenia (synaptické váhy – SV) sú využívané na ukladanie znalostí**

Toto je jedna z definícii NN, akceptovaná NN komunitou. Je zrejmé, že inšpirácia ku vzniku NN prišla z biologických systémov. Hrubo povedané ide o simuláciu **mozgu**. Na prvý dojem vysoko abstraktná disciplína nachádza množstvo aplikácii v praxi a stáva sa prostriedkom pre riešenie problémov v širokom spektre odborných oblastí. Jednou z veľmi významných vlastností NN je, že svojim spôsobom je tzv. **univerzálnym aproximátorom funkcií**. Môže sa nám stať, že máme systém, ktorého popis je mimoriadne náročný alebo je systém natoľko zložitý, že jeho popis je skoro nemožný. Máme však dáta, ktoré do systému vstupujú, a k nim odpovedajúce výstupy. V takejto situácii, môžeme použiť vhodnú NN a pokúsiť sa ju **naučiť** chovať sa ako sledovaný systém pomocou trénovacích údajov (spomínaných vstupov a výstupov). Toto je veľmi dôležitý moment, ktorý determinuje aj aplikačné uplatnenie NN v praxi.

Pri štúdiu NN môžeme rozlišovať tri oblasti:

- **teória NN** - matematický rozbor činnosti NN, problémy NN ako dynamického systému vo všeobecnosti, teoretické rozbory návrhu topológie NN a pod. Je treba upozorniť, že matematický model pre popis chovania sa NN je dosť náročný.
- **simulácia NN** - ide o simuláciu NN pomocou počítačových systémov. Hlavným problémom simulácie je **naučiť** NN na niečo. Proces učenia je veľmi časovo náročný a vyžaduje veľké (najlepšie paralelné) výpočtové systémy. Vo svete je množstvo simulátorov NN. V súčasnej dobe ako najvhodnejší sa zdá **Stuttgartský simulátor NN**.
- **implementácia NN** - ide o implementáciu naučenej NN do harwarovej formy. Taketo systémy existujú a objavujú sa čoraz častejšie mikročipy NN.

Všetky tri oblasti úzko súvisia, ale v súčasnej situácii možnosti využitia výpočtovej techniky prežívajú plný rozvoj.

Na záver tejto časti je treba zdôrazniť dôležitosť paralelizmu NN systémov. Základným elementom NN je **neurón**. V porovnaní s ľudským neurónom my vieme pomocou počítačov nasimulovať omnoho rýchlejší neurón ako je neurón ľudský. Problémom je však množstvo ľudských neurónov a množstvo spojení v mozgu. Podľa výskumov má ľudský mozog okolo 10^{11} až 10^{14} neurónov a počet prepojení na každý jeden neurón predstavuje počet 10^3 až 10^4. Takúto **masívne paralelnú** NN nemáme ešte veľmi dlho šancu nasimulovať. Takže paradoxne, aj keď vieme dosiahnuť rýchlejší procesný element - neurón, nevieme dosiahnuť taký masívny paralelizmus, ktorý v konečnom dôsledku **určuje silu celej NN**. Teda môžeme konštatovať nasledovné:

Sila ľudského mozgu je vo využití obrovského množstva "pomalých" neurónov, zoskupených do veľmi zložitej masívne paralelnej siete, ktorej veľkosť a topológia je v simulačných procesoch zatiaľ zďaleka nenapodobniteľná.

1.3 Stručne o histórii NN

Vzhľadom za závažnosť ostatných častí publikácie uvádzame túto kapitolu iba ako prehľadovú v odstavcoch.

- 1943 začína éra teórie NN pod vedením amerických vedcov McCullocha a Pittsa. Dr. McCulloch bol psychiater a neuroanatóm, kým dr. Pitts bol matematik a celá aktivita, bola sústredená na Univerzite v Chicagu, kde v spomínanom roku 1943 títo dvaja páni prvýkrát definovali binárny neurón. Je zaujímavé, že John von Neumann, pri konštrukcii svojho prvého počítača **ENIAC** v roku 1946 bol do určitej miery inšpirovaný aj spomínanou prácou.

- 1948 Wiener vo svojej knihe **Kybernetika** naznačuje určité koncepty NN

- 1949 Hebb vo svojej knihe **The Organization of Behavior**[2] prvýkrát explicitne spomína pojem učenia a jeho vzťah k synaptickým váham a ich modifikácii.

[2] Organizácia správania

- 1952 dr. Ashby napísal knihu **Design of a Brain: The Origin of Adaptive Behavior**[3]. Tato publikácia mala zásadný význam pre rozvoj NN .

- 1954 dr. Minsky napísal svoju Ph.D. dizertáciu na tému **Neurónové siete** a pozdejšie v roku 1961 napísal zásadný článok **Step Toward Artificial Intelligence**.

- 1956 páni Rochester, Holland, Habit a Duda sa prvýkrát pokúsili o počítačovú simuláciu NN

- 1958 prichádza dr. Rosenblat s novým prístupom k rozpoznávaniu pomocou tzv. **perceptrónu** a pozdejšie prichádza so svojou konvergenčnou teóriou perceptrónu, ktorá predstavuje počiatky **neurodynamiky**.

- 1960 prichádza s Widrow a Hoff a tzv. **Adaline - Adaptive linear element** a o 2 roky nato prichádza Widrow s tzv. **Madaline - Multiple adaptive element**. Tieto príspevky do značnej miery posunuli teoretickú bázu NN dopredu, hoci nedostatok výpočtovej techniky vytváral ohromné zábrany ďalšiemu rozvoju.

- v roku 1965 mala dôležitý význam publikácia dr. Nilssona **Learning Machines**[4].

- v roku 1967 dr. Cowan predstavuje svoju "sigmoidálnu" aktivačnú funkciu

- v roku 1968 dr. Grossberg predstavuje svoj adaptívny model neurónu a používa nelineárne diferenciálne rovnice na jeho popis so zámerom ich použitia pre tzv. **short term memory**[5].

- v roku 1969 dr. Minsky a dr. Papert popisujú činnosť viacvrstvového perceptrónu.

---

[3] Konštrukcia mozgu : pôvod adaptívneho správania
[4] Učiace sa stroje
[5] krátko trvajúce pamäte

- obdobie 1970-80 nazývame obdobím **útlmu**[6]. Dôvodom, boli nedostatočné výpočtové kapacity včítane pamäťových možností. Určité práce v teoretickej oblasti boli urobené ale nie s takou dynamikou ako predtým

- 1975 dr. Little a Shaw popisujú pravdepodobnostný model neurónu

- v roku 1980 prichádza Grossberg s rozvojom tzv. **Competitive learning**, ktorá po rozpracovaní a modifikácii zakladá novú triedu NN založenej na tzv. **Adaptive resonance theory**.

- v roku 1982 dr. Hopfield použil termín **energie NN** pre pochopenie rekurentných sietí. Postupne rozvojom vzniká aj trieda tzv. Hopfieldových sietí. O rok pozdejšie v podstate formuloval princípy simulácie pamäte resp. uchovania informácie v dynamickom systéme.

- dr. Kirkpatrick a jeho kolegovia popisujú procedúru tzv. **simulovaného ochladzovania**[7]. Toto inšpirovalo v roku 1985 dr. Hintona a kol. k návrhu stochastickej učiacej procedúry pre tzv.**Boltzmanov stroj**.

- v tom istom roku (1983) prišli páni Barto a kol. s tzv.**reinforcement learning** a jeho aplikáciou v oblasti riadenia technologických systémov.

- v roku 1986 prišli dr. Rumelhart a kol. s metódou **učenia spätným šírením chyby**[8]. Táto metóda pre svoju relatívnu jednoduchosť je jednou z najrozšírenejších metód učenia NN.

- v roku 1988 prišli páni Broomhead a Lowe s procedúrou **Radial Basis Functions**, pre dopredné siete, ktorá má korene v teórii potenciálnych funkcií, ktoré využili Duda a Hart v roku 1973 pre rozpoznávanie.

---

[6] Decade of Dormacy
[7] simulated annealing
[8] Backpropagation of Error

Hoveuvedený historický prehľad vývoja aktivít týkajúcich sa NN je stručným náčrtom dejín tohto odboru UI. Je zrejmé, že jeho rozvoj bude naďalej dynamický a v prípade nenájdenia inej alternatívnej cesty v UI, sa NN stanú dominantným prvkom moderného chápania UI pre budúcnosť. Tvorba hybridných systémov symbolických a nesymbolických sa javí perspektívnou cestou rozvoja UI.

1.4 Typy úloh riešiteľných pomocou NN

Vo všeobecnosti môžeme vymenovať nasledovné oblasti využitia NN pre :

- problémy aproximácie funkcií
- klasifikácie do tried, klasifikácia situácií
- riešenie predikčných problémov
- problémy riadenia procesov
- tranformácia signálov
- asociačné problémy, simulácia pamäte

Aplikovateľnosť NN vychádza z niektorých základných vlastností NN. Jednou a najvýznamnejšou je fakt, že NN je **univerzálnym aproximátorom funkcie**. Vzhľadom na skutočnosť, že obrovské množstvo problémov funguje ako **nám neznáme funkcie**, tak použitie NN bude v najbližšej dobe veľmi rozsiahle. Určitou brdzou v týchto aplikáciách sú vysoké nároky na výpočtovú techniku, avšak tieto sa veľmi rýchlo menia s rozvojom vysokovýkonných výpočtových systémov.

1.5 Pedagogické poznámky

Cieľom tejto knihy je podať základné informácie z oblasti neurónových sietí pri inžinierskom prístupe. Ide teda o základy, ktoré môžu inžinierski pracovníci využiť pri aplikácii NN v konkrétnych aplikáciách. Tieto základy sú vhodné aj ako inicializačné informácie pre ďalšie štúdium teórie NN.

Je vhodné si uvedomiť základnú štruktúru knihy, ktorá má 2 základné časti a to :

1. dopredné NN a v rámci nich
   - kontrolované učenie na dopredných NN
   - nekontrolované učenie na dopredných NN
2. rekurentné NN a v rámci nich
   - kontrolované učenie na rekurentných NN
   - nekontrolované učenie na rekurentných NN

Ďalej sa v praci zaoberáme aj hybridnými prístupmi a v závere aj modulárnymi NN.

Kniha sa zámerne venovala výhradne základnej teórii NN, ale je potrebné podotknúť, že k celkovému pochopeniu teórie NN je nutné súčasne realizovať praktické experimenty na vybranom simulátore NN. **Jedine kombináciou teoretických znalostí a skúsenosti zo simulačných príkladov je možné získať komplexné a základné znalosti o NN pre ich aplikačné používanie v praxi**. Kniha vo veľkej miere pokrýva základné učivo z predmetu **Neurónové siete** na **Katedre kybernetiky a umelej inteligencie**. Súčasne čitateľom doporučujeme z knihy získané poznatky doplniť prácou na návrhu NN pomocou simulátota neurónových sieti **SNNS** (**S**tuttgart **N**eural **N**etworks **S**imulator), ktorý pracuje na platforme UNIX. Simulátor je dostupný na anonymnom servri internetu pod IP číslom **129.69.2.211**. Vhodnou pomôckou pre lepšie pochopenie NN bude aj pripravovaná publikácia **Neurónové siete v príkladoch** pripravovaná ešte v roku 1996 vo vydavateľstve Elfa–press.

2

Základné pojmy

V tejto kapitole sú uvedené základné pojmy, ktoré je nutné pochopiť skôr, než čitateľ prejde k ďalším kapitolám. Vo všeobecnosti činnosť neurónových sietí rozdeľujeme do dvoch fáz a to:

- fáza **učenia**, kedy sa znalosti ukladajú do synaptických váh neurónovej siete. Ak si označíme maticu **W** ako maticu všetkých synaptických váh[1] neurónovej siete, tak pod učením budeme chápať stav kedy platí, že

$$\frac{\partial \mathbf{W}}{\partial \mathbf{t}} \neq 0 \tag{2.1}$$

  teda synaptické váhy sa počas učenia **menia**.

  Pojem **učenia** pri NN je synonymom pojmu **adaptácie** NN. Ide teda o zbieranie poznatkov, resp. ich uchovanie. Definíciu učenia možeme interpretovať nasledovne

> **Učenie je proces, v ktorom sa parametre NN (synaptické váhy ďalej SV) menia na základe nejakých pravidiel. Charakter týchto pravidiel , ktoré vyvolávajú zmeny SV NN, determinuje typ učenia NN. Pod učením rozumieme adaptáciu NN, ktorá po ukončení učenia bude nositeľkou znalostí získaných počas učenia.**

[1] Pojem synaptická váha bude vysvetlený v časti 2.2

- fáza **života**, kedy sa získané znalosti využívajú v prospech riešenia nejakého problému (napr.klasifikácia, optimalizácia, zhlukovanie a pod.) Prakticky to teda znamená, že ide o stav, kedy

$$\frac{\partial \mathbf{W}}{\partial \mathbf{t}} = 0, \tag{2.2}$$

  teda synaptické váhy sa **nemenia**.

Niekedy sa neurónové siete nazývajú aj ako bezalgoritmické systémy[2]. Toto tvrdenie je pravdivé, ale vzťahuje sa **iba na fázu života** neurónových sietí. Naopak, vo fáze učenia prebieha v neurónových sieťach **cieľavedomý** proces uchovávania poznatkov do synaptických váh neurónových sietí, ktoré sú nositeľmi znalostí.

2.1 Základné pojmy teórie učenia

Učenie je základnou a podstatnou vlastnosťou neurónových sietí. Toto ich odlišuje od nám doposiaľ známeho používania počítačov, kde bolo potrebné vytvoriť algoritmus, podľa ktorého prebehol výpočet. Teda pri klasických algoritmoch neexistujú fázy učenia a života. Pri neurónových sieťach je navrhnutý všeobecný algoritmus učenia, ktorý sieť používa vo fáze učenia. Vhodnosť tohto algoritmu determinuje kvalitu a rýchlosť učenia na predkladaných reprezentatívnych dátach. Pre presnosť vyjadrovania sa v tejto oblasti sú dôležité nasledujúce pojmy:

- príznak je význačná veličina popisujúca jednu vlastnosť skúmaného objektu; objekt môže byť charakterizovaný viacerými príznakmi. Obyčajne príznak vyjadrujeme pomocou číselných hodnôt, tj. pomocou reálnych, celých alebo binárnych čísel.

- príkladom budeme nazývať popis objektu, ktorý je predmetom nášho záujmu, pomocou číselných hodnôt; teda príklad je vlastne n-rozmerný vektor, kde n vyjadruje počet príznakov daného objektu.

- príkladový priestor Y je množina príkladov

[2] algorithm–less systems

- reprezentívna vzorka **s** je prvkom množiny $Y \times \{0,1\}^m$, kde m je počet príkladov vo vzorke. Preto reprezentatívna vzorka predstavuje množinu usporiadaných dvojíc $s = ((y_1, d_1), (y_2, d_2), ..., (y_m, d_m)$. Samotné $d_i \in \{0,1\}$ a predstavuje adekvátne výstupy k jednotlivým vstupom. V praxi výstupy d_i nemusia byť binárne hodnoty. Je potrebné si uvedomiť, že reprezentívna vzorka nám poskytuje empirické údaje chovania sa systému, ktorý nepoznáme. Pomocou poznania vstupov a výstupov systému sa snažíme poznať **chovanie** systému.

- reprezentívna vzorka je bezosporná, ak žiadne dva príklady vo vzorke nie sú sporné, tj. ak $y_i = y_j$, potom musí platiť $d_i = d_j$.

- reprezentatívnu vzorku zvykneme **náhodne** rozdeliť do dvoch základných typov vzoriek:

  1. **trénovacia vzorka** – je to množina usporiadaných hodnôt, ktorá sa **používa pri fáze učenia**. Dôležitosť reprezentatívnosti týchto dát je mimoriadna, lebo znalosti sa pri učení z týchto dát extrahujú do synaptických váh neurónovej siete. Ak táto množina **nie je vhodne** vybraná, potom aj samotné učenie nebude kvalitné. Tieto dáta **by mali** popisovať komplexné chovanie sa systému, ktorý dáta reprezentujú.
  2. **testovacia vzorka** – je to množina usporiadaných hodnôt, ktorá sa **používa vo fáze života** za účelom otestovania získaných znalostí počas učenia.

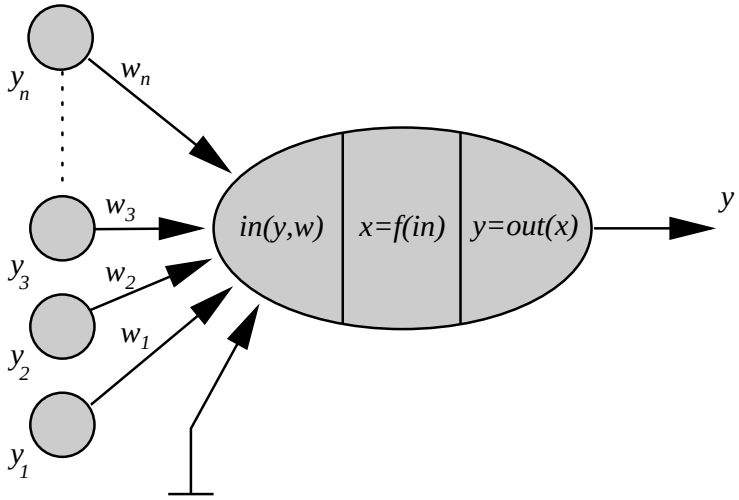
2.2 Základné prvky NN

Základným prvkom a procesnou jednotkou v NN je **neurón**. Štruktúra neurónu je na obrázku 2.1. Neurón pozostáva z nasledujúcich základných častí:

- vstup do neurónu (dendrit)

- prah neurónu - je hodnota θ_i, ktorá vlastne prispieva ku vstupu z externého sveta

- aktivačná funkcia neurónu f, ktorej výsledkom je x_i

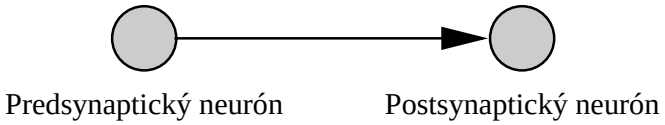


Obr. 2.1: Štruktúra neurónu

- výstupná funkcia neurónu o_i
- synaptické váhy, ktoré sú na synaptických spojeniach (synapsiách), ktoré majú svoj smer a spájajú jednotlivé neuróny do NN

Súčasne podľa toku signálu po synapsii rozoznávane neuróny

- predsynaptické (zdrojové - pred synapsiou)
- postsynaptické (cieľové - po synapsii)



Obr. 2.2: Označenie neurónov

Poznámka: Na označovanie synaptických váh sa používa symbol w_{ij}, kde "i" označuje **postsynaptický** neurón a "j" označuje **predsynaptický** neurón. Teda ide o synapsiu, ktorá vychádza z neuróna "j" a cieli k neurónu "i". Túto konvenciu pri označovaní je vhodné dodržať.

2.2.1 Poznámky k jednotlivým častiam neurónu

- vstup do neurónu - je funkciou jednotlivých vstupov prichádzajúcich od predsynaptických neurónov. Vo väčšine prípadov je to súčet týchto vstupov uvažovaných s určitými váhami, napríklad vstup do $i-$teho neurónu, ktorý má N predsynaptických neurónov, môže byť vyjadrený v tvare

$$in_i = \sum_{j=1}^{N} w_{i,j}\ ou_j + \theta_i \tag{2.3}$$

  kde $w_{i,j}$ sú synaptické váhy, ou_j sú výstupy z neurónov, s ktorými je prepojený, θ_i je prah neurónu i.

  Rovnica (2.3) môže byť prepísaná v tvare

$$in_i = \sum_{j=0}^{N} w_{i,j}\ ou_j \tag{2.4}$$

  kde $w_{i,0} = \theta_i\ a\ ou_0 = 1\ \ alebo\ \ -1$. Prah je vlastne vstup do neurónu z **vonkajšieho sveta**, teda nie z iných neurónov. To znamená, že v prípade, ak nie sú žiadne vstupy do vyšetrovaného neurónu i z ostatných neurónov $j = 1, \ldots, N$, potom vstupom do neurónu je iba prah θ_i. Neuróny, ktoré majú takýto vstup nazývame tiež `sigma` neuróny.

- aktivačná funkcia neurónu

  Už v tomto momente sa musíme začať pozerať na NN ako na dynamický systém, teda systém závislý na čase. Môžeme hovoriť o stave neurónu v čase t resp. v čase $t+1$. Aktivačná funkcia neurónu je funkciou vstupu do neurónu $in_i\ (t)$. Teda stav neurónu i je definovaný premennou x_i v tvare

$$x_i = f(in_i) \tag{2.5}$$

Funkciu f () budeme nazývať **aktivačnou funkciou** neurónu.

Sú známe aktivačné funkcie rôznych tvarov. Uvedieme prehľad tých najdôležitejších. Pôjde o aktivačné funkcie závislé iba na vstupe.

1. **Lineárna** funkcia

$$x_i = f(in_i) = in_i \tag{2.6}$$

2. Funkcia **signum**

$$x_i = f(in_i) = \begin{cases} 1 & ak\ in_i \geq 0 \\ 0 & ak\ in_i < 0 \end{cases} \tag{2.7}$$

Neuróny s takouto aktivačnou funkciu sa nazývajú tiež `McCulloch-Pittsove` neuróny. Tvar tejto funkcie je zobrazený na obr. 2.3

3. **Po častiach lineárna funkcia**
Táto funkcia má tvar

$$x_i = f(in_i) = \begin{cases} 1 & ak\ in_i \geq \frac{1}{2} \\ in_i & ak\ in_i\ je \in (\frac{-1}{2}, \frac{1}{2}) \\ 0 & ak\ in_i \leq \frac{-1}{2} \end{cases} \tag{2.8}$$

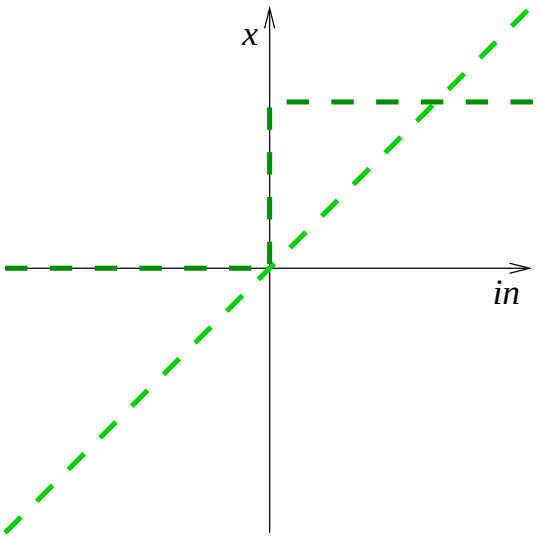
4. **Sigmoidálna funkcia**, ktorá má tvar

$$x_i = f(in_i) = \frac{1}{1 + e^{-\alpha in_i}} \tag{2.9}$$

kde α je parameter strmosti sigmoidy. Táto funkcia je dosť bežná a pri štúdiu sa s ňou často stretneme. Existuje ešte množstvo ďalších aktivačných funkcií.

- **výstupná funkcia** neurónu je taktiež dôležitou súčasťou neurónu ako procesnej jednotky. Vo všeobecnosti teda má tvar

$$ou_i = f(x_i) \tag{2.10}$$



Obr. 2.3: Príklady aktivačnej funkcie neurónu

Veľmi často je funkcia f `identickou` funkciou, tj. $ou_i = x_i$, pre všetky i. Z toho vyplýva, že

$$ou_i = O(x_i) = x_i = f(in_i) \tag{2.11}$$

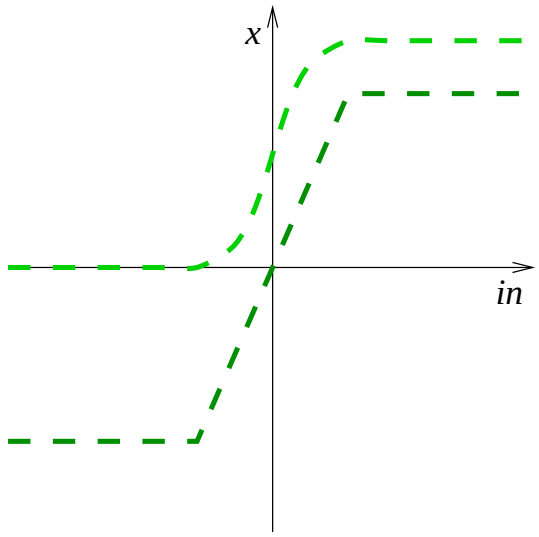
Napriek tomu je nutné rozlíšovať funkciu $f(in_i)$ od funkcie $O(x_i)$, a pri štúdiu NN počítať aj s možnosťou neindentickej výstupnej funkcie.

2.2.2 Synaptické spojenia a váhy

.

Prepojenia medzi neurónmi patria medzi dôležité časti NN. Na týchto **orienovaných** prepojeniach uvažujeme aj tzv. **synaptické váhy**. Váhy ovplyvňujú celú sieť tým, že ovplyvňujú vstupy do neurónov a tým aj ich stavy. Vo všeobecnosti ich rozdeľujeme na :

- kladné, teda **excitačné**
- záporné, teda **inhibičné**



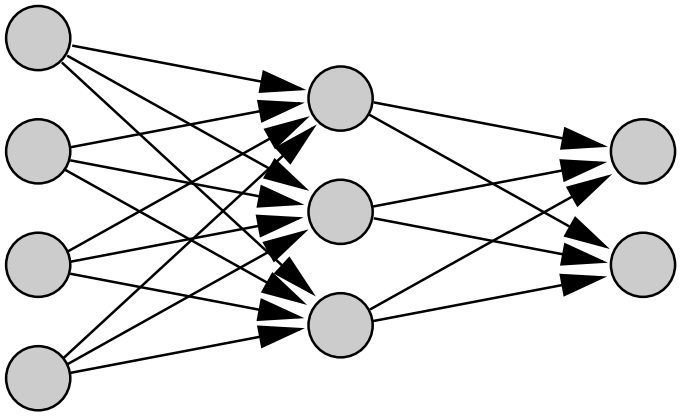
Obr. 2.4: Aktivačné funkcie neurónu

Synaptické váhy medzi neurónmi i, j označujeme $w_{i,j}$. Najdôležitejším momentom pri činnosti NN je práve **zmena váh** $\Delta w_{i,j}$.

2.3 Topológia NN a spôsoby šírenia signálu

Vo všeobecnosti by neurónová sieť mohla mať štruktúru popísateľnú ľubovoľným orientovaným grafom pomocou vrcholov (neuróny) a orientovaných hrán (prepojenia). Vlastnosti takýchto všeobecných sietí sa ťažko analyzujú, preto sú študované a analyzované najprv siete s nejakými pravidelnými štruktúrami. Jednou z pravidelných a pomerne dosť preskúmaných štruktúr je viacvrstvová štruktúra, ktorá je na obr. 2.5. V takých NN sú vrstvy pomenované. Rozlišujeme nasledujúce vrstvy NN

- vstupnú vrstvu, v ktorej neuróny dostávajú vstup len z vonkajšieho sveta a výstup obvykle pokračuje k ďalším neurónom NN
- skrytú vrstvu (hidden layer), v ktorej neuróny dostávajú vstup z ostatných neurónov alebo aj z externého sveta cez prahové prepojenia a ich výstupy pokračujú ďalej do NN



Obr. 2.5: Štruktúra doprednej NN

- výstupná je podobná ako skrytá vrstva, akurát je obvyklé, že výstup z tejto vrstvy vyusťuje do externého sveta.

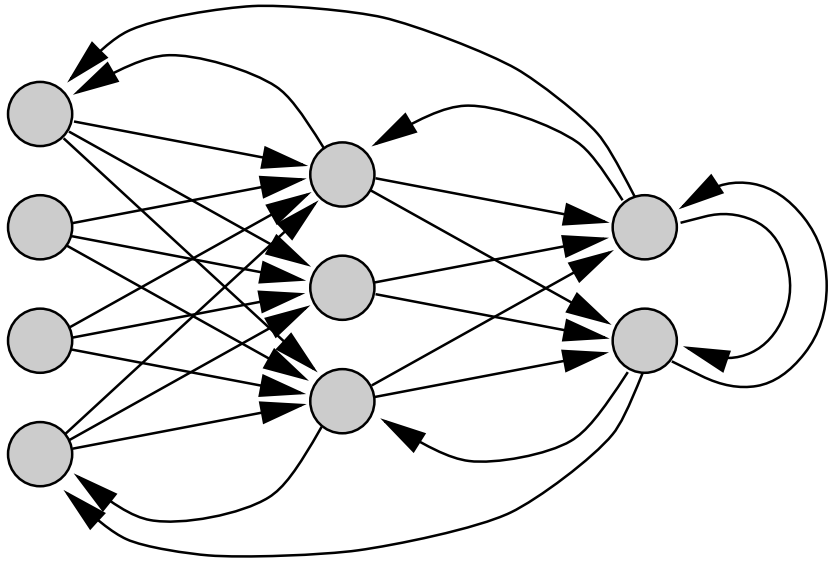
V náväznosti na túto situáciu rozpoznávame aj samotné neuróny ako `vstupné, skryté, a výstupné`.

Pri návrhu NN vo všeobecnosti rozdeľujeme topológiu NN dvoch základných skupín:

- dopredné NN (feed-forward **FF NN**) - pri týchto sa signál šíri po orientovaných synaptických prepojeniach len jedným smerom a to dopredu - viď obr. 2.5.

- rekurentné NN (recurrent **RC NN**) - pri rekurentných sieťach je dosť ťažké rozdelenie vrstiev a neurónov na `vstupné, resp. výstupné`. Niekedy neuróny v rekurentných sieťach predstavujú vstupné ale aj výstupné typy neurónov a tým aj vrstiev (viď obrázok 2.6.). Špecialným prípadom sú tzv. čiastočne rekurentné NN, v ktorých je stanovená určitá požiadavka na štruktúru a na prepojenia. Napr. vrstvové čiastočne rekurentné siete pripušťajú šírenie signálu oboma smermi.

Šírenie signálu v rámci NN môže byť veľmi rozmanité. Uvedieme niektoré z nich:



Obr. 2.6: Štruktúra RC NN

- synchrónne šírenie signálu - všetky neuróny menia svoj stav do taktu (prostredníctvom synchronizačných hodín)
- sekvenčné - neuróny menia svoj stav postupne pri šírení signálu
- blok-sekvenčné - aktivizujú sa len skupiny neurónov, podľa vopred určenej stratégie
- asynchrónne - neuróny menia svoje stavy asynchrónne, teda úplne nezávisle jeden od druhého

Pri pozornom premýšlaní o tom, čo sa môže diať v týchto NN a pri rôzmanitých spôsoboch šírenia signálov nutne musíme priznať zložitosť problematiky chovania sa NN.

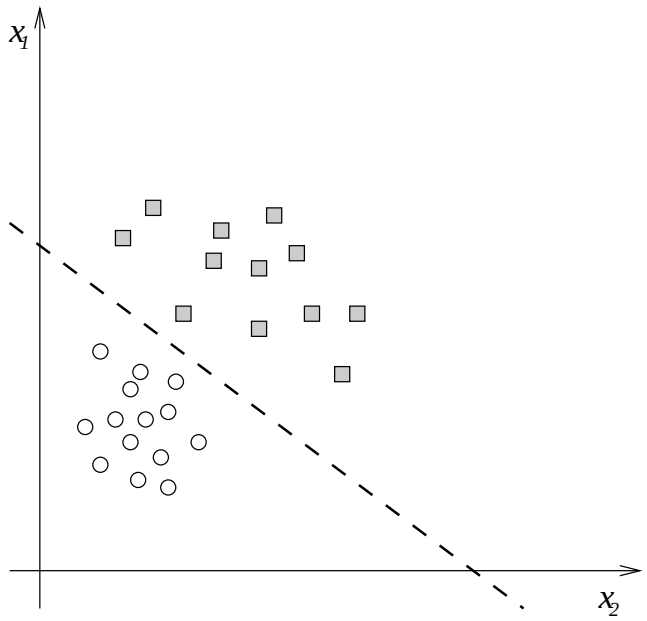
2.4 Perceptrón – najjednoduchšia neurónová sieť

V tejto časti budeme prezentovať najjednoduchšiu neuronovú sieť, ktorá predstavuje základ pre komplikovanejšie dopredné siete.

2.4.1 Čo je to perceptrón ?

Perceptrón bol navrhnutý Rosenblattom [16] a je určený na **dichotomickú** klasifikáciu, tj. rozdelenie do dvoch tried, pri ktorých sa predpokladá, že triedy sú **lineárne separovateľné** v príkladovom priestore.

Nech je daná množina vektorov $\mathbf{X} = \{\mathbf{x^j}\}, j = 1, \ldots, k, \ldots, \infty$ v n-rozmernom priestore. O každom z týchto vektorov vieme, že určite patrí do triedy **CL1** alebo **CL2**. Pod lineárnou separovateľnosťou dvoch tried rozumieme situáciu, keď existuje možnosť oddeliť objekty v príkladovom priestore pomocou nadroviny napr: priamka v 2-rozmernom alebo rovina v 3-rozmernom priestore. Príklad lineárnej separovateľnosti v rovine je na obr. 2.7

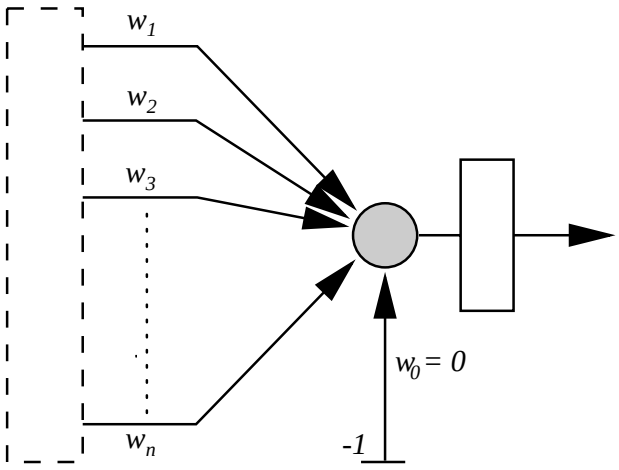


Obr. 2.7: Lineárna separovateľnosť tried v príkladovom priestore

2.4.2 Topológia perceptrónu

Označme perceptrón, ktorý bol navrhnutý Rosenblattom ako jednoduchý perceptrón so svojím učiacim algoritmom (ďalej JPR). Má tri vrstvy a to :

- senzorová vrstva
- asociatívna vrstva
- výstupný neurón



Obr. 2.8: Topológia perceptrónu navrhnutého Rosenblattom

Asociatívne neuróny boli určené na stanovenie príznakov vhodných na klasifikáciu a v literatúre sa niekedy nazývajú aj ϕ funkcie. Podľa charakteru tejto funkcie v podstate môžeme zatriediť JPR do rôznych skupín NN. Spojenie medzi senzorovou vrstvou a asociatívnou má pevné váhy, teda sa na procese učenia nezúčastňuje. Spojenie medzi asociatívnou vrstvou a výstupným neurónom je prepojené synapsiami s premenlivými SV. Teda vstup do výstupného neurónu bol daný rovnicou

$$in(t) \;=\; \sum_{j=1}^{n} w_j(t)x_j(t) \;-\; \theta \tag{2.12}$$

kde n je počet neurónov v asociatívnej vrstve, $w_j(t)$ sú váhy medzi asociatívnou vrstvou a výstupným neurónom, kde neurón "j" je v asociatívnej vrstve a na vstupe je "k-ty" prvok množiny $\mathbf{X}$, $\mathbf{x_j(t)}$ je stav "j"-teho neurónu a θ je prah. Výstup prechádza prahovaním v zmysle konečného výstupu

$$ou(t) \;=\; \begin{cases} 1 & ak\ in(t) \;\geq\; 0 \\ 0 & ak\ in(t) \;<\; 0 \end{cases} \tag{2.13}$$

Je nutné poznamenať opätovne fakt, že JPR bol navrhnutý za cieľom klasifikácie do dvoch tried **CL1 a CL2**. V prípade elementárneho PR je separujúca nadrovina daná rovnicou

$$\sum_{j=1}^{n} w_j(t)x_j(t) \;-\; \theta \;=\; 0 \tag{2.14}$$

Z praktického hľadiska označme vektor $\mathbf{w}(t) \;=\; (w_0(t), w_1(t), w_2(t), \ldots, w_n(t))$, vektor $\mathbf{x}(t) \;=\; (x_0(t), x_1(t), x_2(t), \ldots, x_n(t))$ kde n je rozmer vektorov, teda počet neurónov v asociatívnej vrstve. Je potrebné poznamenať, že stále je

$$w_0(t) \;=\; \theta$$

a

$$x_0(t) \;=\; -1.$$

Taktiež v matematických operáciách budeme pre prehľadnosť vynechávať označenie **transponovania**. Teda na vstup do JPR ponúkame vstupy $\mathbf{x}(t)$ a k nemu korešpondujúce výstupy na výstup JPR $\mathbf{ev}(t)$. To všetko realizujeme opakovane po určitý konečný počet krokov.

2.4.3 Algoritmus učenia perceptróna

Najprv ukážeme algoritmus **procesu učenia** - teda hľadania vhodných synaptických váh, a potom pre tento algoritmus dokážeme vetu o konvergencii perceptrónu, tj. dokážeme, že po konečnom počte krokov JPR sa dostane do stavu, v ktorom obidve triedy vie separovať.

Postup učenia JPR bude teda nasledovný:
Prepokladajme, že $(\mathbf{x}(1), d^1), (\mathbf{x}(2), d^2, ..., (\mathbf{x}(m), d^m)$ je trénujúca vzorka vektorov, na ktorej sa bude perceptrón učiť, $\mathbf{x}(t) \in R^n$, $d^t = 1$, ak $\mathbf{x}(t) \in$ **CL1**, $d^t = -1$, ak $\mathbf{x}(t) \in$ **CL2**.

- inicializácia váh
- ak vstupný vektor $\mathbf{x}(t)$ je **správne** klasifikovaný pomocou $\mathbf{w}(t)$ potom **nenastáva zmena** váh pre ďalší vstup $\mathbf{w}(t+1)$, teda
  1. ak $\mathbf{w}(t)\mathbf{x}(t) \geq 0$ a $\mathbf{x}(t)$ skutočne patrí do **CL1**, tj. $d^t = 1$ (podľa $ev(t)$), potom
  $$\mathbf{w}(t+1) = \mathbf{w}(t) \tag{2.15}$$
  2. ak $\mathbf{w}(t)\mathbf{x}(t) < 0$ a $\mathbf{x}(t)$ skutočne patrí do **CL2, tj.** $d^t = -1$ (podľa $\mathbf{ev}(t)$), tak taká istá rovnica platí ako (2.15)
- ak vstupný vektor $\mathbf{x}(t)$ je **nesprávne** klasifikovaný pomocou $\mathbf{w}(t)$ potom **nastáva zmena** váh $\mathbf{w}(t+1)$, teda
  1. ak $\mathbf{w}(t)\mathbf{x}(t) \geq 0$ a $\mathbf{x}(t)$ skutočne patrí do **CL2** (podľa $\mathbf{ev}(t)$) potom
  $$\mathbf{w}(t+1) = \mathbf{w}(t) - \gamma\, \mathbf{x}(t) \tag{2.16}$$
  2. ak $\mathbf{w}(t)\mathbf{x}(t) < 0$ a $\mathbf{x}(t)$ skutočne patrí do **CL1** (podľa $\mathbf{ev}(t)$), tak potom
  $$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma\, \mathbf{x}(t) \tag{2.17}$$

  kde γ je učiaci parameter a môže predstavovať ľubovoľnú kladnú premennú, ktorá je po dobu učenia konštantná alebo sa môže aj meniť teda $\gamma \to \gamma(t)$. Vyššie uvedená rovnica môže byť zapísaná v tvare
  $$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma\, (d^t - ou(t))\mathbf{x}(t)/2,\ t = 1, 2, ... \tag{2.18}$$

Ak položíme $\gamma = 0.5$ postup učenia môže byť stručne zapísaný nasledovne:

$$\mathbf{w}(0) = 0 \tag{2.19}$$

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{z}(t), \qquad \text{ak } \mathbf{z}(\mathrm{t})\mathbf{w}(\mathrm{t}) \leq 0 \tag{2.20}$$

$$\mathbf{w}(t+1) = \mathbf{w}(t), \qquad \text{inak} \tag{2.21}$$

kde $\mathbf{z}(t) = +\mathbf{x}(t)$, ak $d^t = +1$, $\mathbf{z}(t) = -\mathbf{x}(t)$, ak $d^t = -1$. Teda úprava váh sa vykonáva len vtedy, keď $\mathbf{z}(t)\mathbf{w}(t) \leq 0$, čo **reprezentuje oba prípady výskytu chyby**.

2.4.4 Veta o konvergencii perceptrónu

Predpokladajme, že separujúca nadrovina existuje, tj. triedy sú separovateľné. Cieľom tejto časti je dokázať, že JPR po konečnom počte krokov, tj. ak dostane "k" rôznych vstupov[3] z množiny $\mathbf{X^n}$, $k < S$, kde S je konečné prirodzené číslo, dosiahne stabilný stav, tj. už bude vedieť klasifikovať hocijaký vstup z množiny $\mathbf{X}$ správne, t.j. váhy JPR sa nebudú meniť. Výsledkom dôkazu bude tiež horný odhad pre počet krokov k. Pre jednoduchosť celý dôkaz realizujeme vo vektorovom tvare.

Predpokladajme, že rovnica hľadanej nadroviny bola nájdená, tj. bol nájdený vektor váh $\mathbf{v}$, ktorý je riešením. To znamená, že už nedôjde k zmene váh, tj. platí $\mathbf{vx}(t) > 0$ pre t = 1, 2, ... Predpokladajme, že nastáva situácia, že v t-tom kroku je nesprávna klasifikácia teda prípad (2.20), tj.

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{z}(t) \tag{2.22}$$

čo je možné upraviť odčítaním výrazu $\phi\mathbf{v}$ od obidvoch strán rovnice, teda

$$\mathbf{w}(t+1) - \phi\mathbf{v} = \mathbf{w}(t) - \phi\mathbf{v} + \mathbf{z}(t) \tag{2.23}$$

kde ϕ je kladná konštanta. Pre výpočet normy vektora na pravej strane platí

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 = \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 + 2\mathbf{z}(t)(\mathbf{w}(t) - \phi\mathbf{v}) + \|\mathbf{z}(t)\|^2 \tag{2.24}$$

[3] k - omylov

Pretože $\mathbf{z}$(t) je klasifikované nesprávne, platí $\mathbf{z}(t)\mathbf{w}(t) \leq 0$, a teda

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - 2\phi\mathbf{v}\mathbf{z}(t) + \|\mathbf{z}(t)\|^2 \qquad (2.25)$$

Zaveďme nasledujúce označenie

$$\alpha = min_{j=0,\ldots,k}(\mathbf{v}\mathbf{z}(j)) \qquad (2.26)$$

α je zrejme kladné číslo, pretože $\mathbf{z}(t)\mathbf{v} > 0$. Nech

$$\beta = max_i\|\mathbf{z}(i)\| \qquad (2.27)$$

Nerovnosť (2.25) je možné upraviť nasledujúcim spôsobom

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - 2\phi\alpha + \beta^2 \qquad (2.28)$$

Ak vyberieme ϕ dostatočne veľké, napríklad $\phi = \beta^2/\alpha$, dostaneme

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - \beta^2 \qquad (2.29)$$

Teda druhá mocnina vzdialenosti medzi $\mathbf{w}$(t) a $\phi\mathbf{v}$ je redukovaná aspoň o β^2 pri každej úprave. Po k úpravách potom dostávame nerovnosti

$$0 \leq \|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(0) - \phi\mathbf{v}\|^2 - k\beta^2 \qquad (2.30)$$

Z toho vyplýva, že postupnosť úprav musí skončiť najviac po konečnom počte k_0 krokov, kde

$$k_0 = \|\mathbf{w}(0) - \phi\mathbf{v}\|^2/\beta^2 \qquad (2.31)$$

Veľkosť k_0 je závislá od α a β, pri vypočte ktorých sa vyskytne znovu hodnota k_0. Ak sú však známe odhady α a β, potom k_0 je možné určiť hneď. Teda ak riešenie existuje, je dosiahnuteľné po konečnom počte krokov. Predpokladajme znovu, že počiatočné nastavenie váh je nulové. Tým predchádzajúcu rovnicu zjednodušíme

$$k_0 = \phi^2\|\mathbf{v}\|^2/\beta^2 \qquad (2.32)$$

Na základe (2.32) sme dokázali, že k existuje a je to konečné číslo, a tým môžeme vysloviť nasledovné tvrdenie :

Majme trénovaciu množinu vektorov X, ktoré môžu patriť len do dvoch rôznych tried CL1 a CL2, ktoré sú lineárne separovateľné. Perceptrón po realizovaní "k_0" omylov sa určite dostane do stavu, keď nebude meniť svoje SV, kedy konverguje . To znamená, že bude spoľahlivo klasifikovať vektory do príslušných tried.

x_1	x_2	výstup
0	0	0
0	1	1
1	0	1
1	1	0

Tabuľka 2.1: XOR binárna funkcia

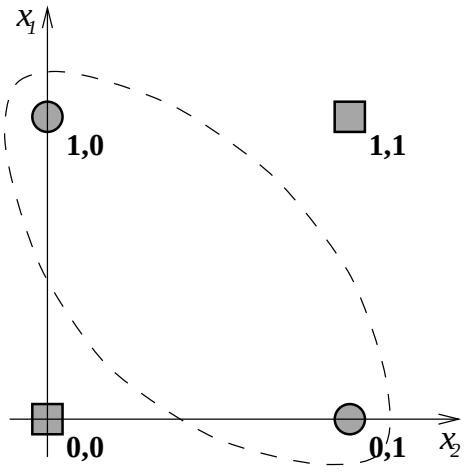
2.4.5 XOR-problém, skrytá vrstva NN

V prípade JPR sme predpokladali, že na vstup budú vstupovať príznaky objektov, ktoré sú v príznakovom priestore lineárne separovateľné. V prípade, že táto podmienka nie je splnená, JPR je nie schopný spoľahlivo realizovať svoju dichotomickú klasifikáciu. Príkladom **nelineárne separovateľnej** funkcie je funkcia XOR. Klasickým JPR nevieme zabezpečiť separáciu výsledkov tejto funkcie v žiadnom prípade, čo je zrejme z obr. 2.9. Je potrebné poznamenať, že stále uvažujeme o **jednoduchej aktivačnej funkcii** napr. typu signum. V prípade iných aktivačných funkcii by sme vedeli XOR problém riešiť. Cieľom je však to, aby sme vedeli aj pri jednoduchších funkciách riešiť klasifikačné úlohy.

Východiskom je práve zavedenie **skrytej vrstvy**, ktorá pomôže tento problém riešiť. Zapojenie tejto skrytej vrstvy znamená použitie viacnásobnej lineárnej separácie (na výstup prvej vrstvy je znovu použitá lineárna separácia) , kde je možné už tieto objekty lineárne odseparovať. Teda objekty, ktoré v 2-rozmernom priestore nie sú lineárne separovateľné sa dajú separovať zavedením ďalšej vrstvy neurónov. Na obrázku sú znázornené 2 rôzne topológie NN, ktoré vedia riešiť XOR-problém. Je potrebné si uvedomiť, že tu ide o aproximáciu binárnej funkcie uvedenej v tabuľke a grafe.

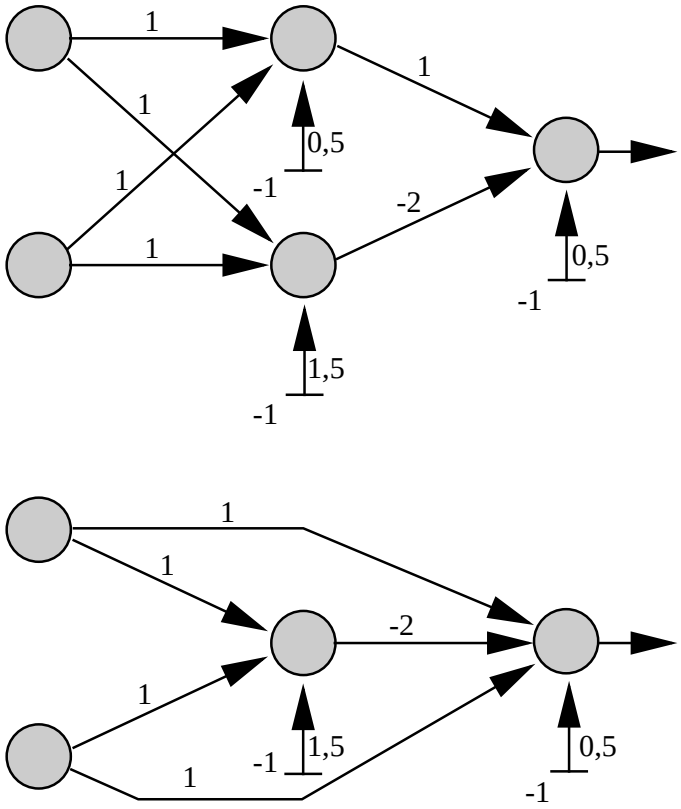
Poznámka:

Záverom tejto celej kapitoly je potrebné upozorniť na terminologickú zameniteľnosť medzi metódou učenia JPR a topológiou typu perceptrón. Siete s rovnakou topológiou môžu mať rôzne algoritmy učenia.



Obr. 2.9: Grafické znazornenie funkcie XOR

Napr. Adaline a JPR majú rovnakú topológiu, ale odlišnú metódu učenia ako aj celkové zameranie NN. Pod topológiou **perceptrón** rozumieme FF NN resp. pod **viacvrstvovým perceptrónom** viacvrstvové FF NN. Teda na topológii typu perceptrón môže byť relizované principiálne kontrolované učenie rôzneho druhu.



Obr. 2.10: Dve možné riešenia funkcie XOR pomocou skrytej vrtsvy

3

Učenie a jeho paradigmy

Vo všeobecnosti rozdeľujeme prístupy k učeniu do dvoch veľkých skupín:

- **kontrolované učenie** - supervised learning (učenie s učiteľom), ktoré sa ďalej rozdeľuje do dvoch podskupín a to do
  - štrukturálne učenie - tu rozoznávame dve skupiny metód a to :
    - autoasociačné - na prvý pohľad je nelogické, aby sme na vstup do NN dávali vzorku α a na výstup tú istú vzorku α. Teda NN sa musí tak adaptovať, aby to čo je na vstupe, bolo aj na výstupe. Takéto NN majú význam napr. pri simulácii pamäte apod.
    - heteroasociačné - tieto vlastne učia NN, že ku vstupu α patrí výstup β. Teda NN sa naučí **rozpoznávať** vstupy a zatrieďovať ich, ako jej to povedal **učiteľ**.
  - temporálne učenie - je v podstate heteroasociatívne učenie, ale na vstup musí prísť za čas $t > 0$ sekvencia vstupov a až tejto sekvencii ako **celku** je priradený jeden výstup. Napr. iba sekvencia ťahov šachu v čase znamená výhru, alebo iba určitý časový vývoj ekonomických a iných parametrov môže znamenať vzrast ceny akcií na burze a pod.
- **nekontrolované učenie** - unsupervised learning (učenie bez učiteľa).

3.1 Paradigmy kontrolovaného učenia

Filozofia kotrolovaného učenia (teda spôsob zmeny SV) je ovplyvnený prítomnosťou **učiteľa** v celom procese učenia. **Prakticky to znamená, že NN musíme ponúknuť v procese učenia**

- **vstup do NN**
- **k vstupu prislúchajúci výstup NN.**

Teda prístupy ku zmene SV (učeniu) môžeme v prípade kontrolovaného učenia koncepčne rozdeliť do 3 skupín:

- učenie za základe opravy chyby **(error correction learning)**
- stochastické učenie **(stochastic learning)**
- učenie na základe hodnotenia činnosti[1] **(reinforcement learning)**

Poznámky k jednotlivým typom :

1. **Učenie na základe korekcie chyby**
   Tento prístup predpokladá zmenu SV ako funciu premenej e_i, kde e_i predstavuje rozdiel medzi očakávaným stavom neurónu "i" (ev_i) a vypočítaným stavom neurónu "i" (x_i) v procese učenia. Teda
   $$e_i \;=\; ev_i \;-\; x_i \tag{3.1}$$
   Potom môžeme napísať všeobecný vzorec pre výpočet zmeny SV pre spojenie medzi výstupným neurónom "i" a do neho vstupujúcim neurónom "j" v tvare
   $$\Delta w_{ij} \;=\; \gamma\, x_j\, e_i \tag{3.2}$$
   kde γ je parameter učenia, zvyčajne v intervale hodnôt $(0, 1>$, x_j je stav neurónu "j". V prípade viacvrstvovej siete sa pri výpočte jednotlivých zmien váh použije rekurentný vzorec pre zmenu váh, ktorý sa odvíja od výstupu NN a smeruje späť do NN (viď BP-algoritmus).

[1] veľmi voľný preklad

2. **Stochastické učenie**
   Pri stochastickom type učenia ide o zmeny SV založené na stochastických prístupoch. Globálna stratégia je založená na nasledovných krokoch:

   - navrhne sa stochastická zmena SV a vypočíta sa **energia** NN.
   - ak zmena priniesla **zníženie** energie NN, **návrh zmeny sa príjme**
   - ak zmena nepriniesla spomínaný efekt **návrh sa zamietne**

   Príkladom takýchto NN je Boltzmanov stroj a jeho modifikácie .

3. **Učenie na základe hodnotenia činnosti**
   Tento charakter učenia je podobný ako v prípade učenia podľa korekcie chyby,ale základným rozdielom je, že sa zhodnocuje stav výstupu celej výstupnej vrstvy pomocou nejakej skalárnej veličiny. Potom môžeme napísať všeobecný tvar rovnice učenia:

$$\Delta w_{ij} \;=\; \gamma\,(r \;-\theta_i)\; e_{ij} \tag{3.3}$$

   kde r je skalárna hodnota úspešnosti celej NN odvodená z výstupnej vrstvy NN, θ_i je prahový koeficient úpravy pre neurón "i" a e_{ij} je koeficient rozhodnutia a predstavuje zmenu pravdepodobnosti minimálnej chyby podľa synaptickej váhy, ktorý sa vo všeobecnosti vypočíta

$$e_{ij} \;=\; \frac{\partial ln\; g_i}{\partial w_{ij}} \tag{3.4}$$

   kde teda g_i je pravdepodobnosť, že očakávaný výstup sa bude rovnať vypočítavanému výstupu ev_{ij}(teda minimálnej chybe), teda

$$g_i \;=\; P\;(x_i \;=\; ev_i|W_i\;,\;\Lambda) \tag{3.5}$$

   kde ev_i je očakávaná hodnota výstupného neurónu "i", x_i je vypočítaná hodnota neurónu, W_i je vektor SV, ktoré vstupujú do

neurónu "i" a Λ je vektor hodnôt aktivačných stavov neurónov, ktorých SV vstupujú do neurónu "i".

Záverom ku kontrolovanému učeniu je nutné počiarknuť, že existujú metódy učenia, ktoré predstavujú hybridný prístup zostavený z horeuvedených prístupov.

3.2 Paradigmy nekontrolovaného učenia

Pri tomto type učenia ide o spracovanie vstupu do NN na základe určitých zákonitosti. **Prakticky to znamená, že NN môžeme počas učenia ponúknuť (iba) vstup do NN**. NN potom sama spracuje a určuje výstup.Z toho dôvodu nazývame siete, ktoré používajú takúto metódu učenia tzv. samo-organizujúce sa siete (**self-organising NN**). Otázka ukončenia učenia je založená na nájdení GS NN. Teda NN sa prestane učiť, ak zmena SV v čase t a v čase $t+1$ budú dostatočne malé, t.j. pre matice váh W platí

$$|\Delta W(t) \; - \; \Delta W(t+1)| \; \leq \; \epsilon \tag{3.6}$$

kde ϵ je dostatočne malá hodnota. NN s nekontrolovaným učením ukončuje svoju činnosť, ak sa dostane do GS. Tu môžeme hovoriť o GS a nie o konvergencii NN, lebo nevieme, aký bude výstup z NN.

Vo všeobecnosti rozdeľujeme metódy nekontrolovaného učenia do dvoch základných typov:

- Hebbovo učenie **(Hebbian learning)**
- Kooperačné a konkurenčné učenie **(Cooperative and competitive learning)**

Poznámky k jednotlivým typom učenia :

1. **Hebbovo učenie** predstavuje prístup, ktorý prvýkrát spomenul dr.Hebb vo svoje knihe "Organizácia správania". Tieto myšlienky boli v ďalšom rozpracované do podoby dvoch nasledovných zásad:
   - ak 2 neuróny na opačných stranách synapsie sú aktivované naraz (synchrónne), potom SV synapsie sa zvýši.

- ak 2 neuróny na opačných stranách synapsie sa aktivizujú v rôzných časoch (asynchrónne), potom SV synapsie sa zníži alebo $\rightarrow$ 0.

Takéto synapsie NN, ktoré splňujú spomenuté zásady nazývame **Hebbove synapsie**. Globálne potom môžeme Hebbove synapsie charakterizovať 3 nasledovnými mechanizmami:

- mechanizmus závislý na čase - stav SV je závislý na stave pred a postsynaptických neurónov, ktoré sú závislé na čase,
- mechanizmus lokálnej definície - stav synapsie má priestorovo-časový charakter. Synapsie a SV sú nositeľmi informácie a zdrojom ďalšej lokálnej zmeny. Preto NN s Hebbovými synapsiami sú schopné nekontrolovaného učenia,
- mechanizmus interakcie a korelácie - spôsob zmeny SV zaručuje interakciu medzi pred- a postsynaptickými neurónmi. Tak isto je zaručený korelačný vzťah medzi spomínanými neurónmi, práve zaručením vlastností Hebbových synapsii.

Matematicky môžeme vyjadriť zmenu SV Hebbových synapsii pre dvojicu neurónov "k" a "i" ("k" vstupuje do "i") nasledovne :

$$\Delta w_{ki}(t) \;=\; funkcia(x_k(t), x_i(t)) \tag{3.7}$$

resp.

$$\Delta w_{ki}(t) \;=\; \gamma \; x_k(t) \; x_i(t) \tag{3.8}$$

Pri takýchto prípadoch je možná saturácia, ale tieto problémy sa riešia modifikáciou uvedených prístupov. K záveru informatívneho popisu Hebbovho učenia je treba naznačiť, že v závislosti s horeuvedenými zásadami rozdeľujeme synapsie do troch skupín a to

- Hebbove synapsie
- nie-Hebbove synapsie

- inverzné Hebbove synapsie

2. **Kooperačné a konkurečné učenie**

   Majme NN s M-neurónmi, ktoré majú stavy x_i ,$i = 1, \ldots, M$. Ak sledujeme NN z hľadiska dynamiky potom označme

$$\frac{d\ x_i}{d\ t} = V_i(X) \tag{3.9}$$

   kde $X = (x_1, x_2, \ldots, x_M)$ je vektor aktivačných hodnôt neurónov v celej NN. V_i teda predstavuje zmenu aktivácie neurónu za jednotku času. Ako k tejto zmene prispel iný neurón, môžeme vyšetriť, ak uvažujeme

$$\frac{\partial V_i}{\partial x_j} = \begin{cases} \leq 0 \ \ \forall\ j \neq i & konkurenčné\ \ učenie \\ \geq 0 \ \ \forall\ j \neq i & kooperačné\ \ učenie \end{cases} \tag{3.10}$$

   Pre konkurenčné učenie platí tzv. zákon **"kto vyhrá berie všetko"**[2] a môžeme ho zhrnúť do nasledovných krokov:

   - vstup prichádza do NN
   - signály prechádzajú do nasledujúcej vrstvy
   - neurón s najvyššou hodnotou sa stáva **víťazom** a je nastavený na 1 a ostatné neuróny sú nastavené na 0. Následne sa upravia iba tie SV, ktoré smerujú k "víťaznému" neurónu "i" napr :

$$\Delta w_{ki} = w_{ki} + (-w_{ki} + ou_h) \tag{3.11}$$

     kde $ou_{k,i}$ sú stavy príslušných neurónov pri existencii nelineárnych aktivačných funkcií (resp. ich výstupy).

   Konkurenčné metódy učenia sa vhodne využívajú pre zámery zhlukovania vstupných dát.

[2] winner takes all

3.3 Význam inicializácie pri učení NN

Vo veľkom množstve metód učenia je prvým krokom inicializácia SV NN na náhodné hodnoty. Tomuto kroku je potrebné klásť príslušnú **dôležitosť** vzľadom na veľký vplyv na samotný proces učenia. Je potrebné si uvedomiť, že v priebehu učenia sa spočítavajú zmeny SV NN a tie sa potom využijú, a to pri **prvom** výpočte novej hodnoty v iterácii (1):

$$w(1) \;=\; w(0) \;+\; \Delta w(0) \tag{3.12}$$

kde $w(t)$ je SV, ktorá vznikla inicializáciou NN. Vo väčšine prípadov sa inicializuje náhodnými číslami z intervalu napr. $(-1, 1)$.

3.4 Globálna stabilita a konvergencia NN

V predchádzajúcich častiach už bola NN pripomenutá ako dynamický systém, ktorý adekvátne k tomu je potrebné aj spravovať. Vyšetrovanie globálnej stability a konvergencia NN sú dve rôzne veci pre vyšetrovaný dynamický systém. Je treba poznamenať, že ambície vyšetrovania NN sú určené pre ľudí, ktorí navrhujú nové NN a nové metódy učenia. Preto bude táto oblasť spomenutá iba okrajovo. V teórii NN sa jej venuje časť zvaná **Neurodynamika**. V nej sa používajú prostriedky stavového, resp. fázového priestoru pre vyšetrovanie stability systému.

3.4.1 Globálna stabilita NN

V rámci globálnej stability (ďalej GS) NN, ako nelinárneho dynamického systému uvažujeme dva prípady:

- GS NN počas procesu učenia[3]
- GS NN mimo procesu učenia

Všeobecne pod GS NN rozumieme stav, kedy neuróny NN zotrvávajú v nejakom stave[4]. Ak si predstavíme NN,ktorá má M neurónov, ktoré majú stavy $x_i \quad i = 1, \ldots, M$, potom pod globálnou stabilitou rozumieme vo všeobecnosti stav

[3] proces učenia je charakteristický zmenou váh NN v čase

[4] nemusí to byť taký stav, aký my chceme

$$\frac{dx_i}{dt} \rightarrow 0 \quad pre \quad i = 1, \ldots, M \tag{3.13}$$

Na vyšetrovanie GS NN by bolo vhodné mať funkciu, ktorá stavy týchto M neurónov vyjadrí integrálne nejakou skalárnou formou. Vo všeobecnosti na popis GS je dobrá funkcia, ktorá tranformuje M-rozmerný priestor stavou do jedného skalárneho čísla. Teda

$$R^M \rightarrow R^1 \tag{3.14}$$

V teórii NN sa popisuje základná tzv. **priama metóda Ljapunova** popisujúca GS NN. Majme teda NN s M neurónmi, ktoré majú stavy $\mathbf{x}(t) = (x_1(t), \ldots, x_M(t))$. Nech sú splnené nasledovné požiadavky:

1. ak
$$X = \frac{\dot{\mathbf{x}}(t)}{\dot{t}} = G(\mathbf{x}(t))$$
a platí, že $X = 0$, ak všetky $x_i = 0$

2. X je holomorfná, t.j. existujú prvé defivácie podľa x_i

3. $\sum_{i=1,\ldots,M} x_i(t) \leq H$ pre $t \geq t_0$, teda po nejakom čase $(\geq t_0)$, bude suma stavov $x_i(t)$ menšia ako konštanta H.

4. ak vieme zostrojiť funkciu $L(X)$ pre ktorú platí, že
$$\sum_{i=1,\ldots,M} \frac{\partial L}{\partial x_i} \leq 0 \tag{3.15}$$
pre všetky x_i

potom funckia L sa nazýva **Ljapunovova funkcia energie NN**, popisuje GS NN a pre stabilné NN $L \rightarrow 0$. Táto funkcia skutočne realizuje tranformáciu stavov neurónov NN pomocou skalárnych hodnôt. Uvedená metóda vyšetrovania stability je pre popis NN základnou a od nej sa odvádzajú ďalšie metódy popisujúce jednotlivé typy NN (Cohen-Grossbergova resp. Cohen-Grossberg-Koskova teoréma).

Pre FF NN má význam hovoriť o GS NN len v procese učenia. V procese mimo učenia[5] tam principiálne nemôže dôsť k nestabilite NN. Pri RC NN však má význam hovoriť o stabilite pri učení ako aj mimo neho, vzhľadom na existenciu rekurentných prepojení. Teda, ak je RC NN počas učenia stabilná, teoreticky mimo učenia môže na vstup NN prísť nejaký vstup, ktorý NN urobí nestabilnou[6].

Záverom tejto časti je treba podčiarknuť dôležitosť neurodynamiky hlavne v prípade vývoja nových učiacich metód. Už navrhnuté metódy učenia NN sa vyznačujú GS. Tabuľka prezentuje význam vyšetrovania GS pri rôznych topológiach NN (kvôli prehľadnosti).

	FF–NN	RC–NN
fáza učenia	**áno**	**áno**
fáza života	**nie**	**áno**

3.4.2 Konvergencia NN

Pojem **konvergencia NN** je spojený iba s fázou učenia NN. Naviac o konvergencii NN môžeme hovoriť iba pri kontrolovanom učení, teda vtedy, keď vieme presne aký výstup z NN očakávame. Ide teda o GS NN ale do nami definovaného stavu. Z toho vyplýva,že konvergenciu môžeme definovať ako existenciu takého čísla n_0, kedy

$$|x_i^n - x| \leq \epsilon \tag{3.16}$$

pre všetky $n \geq n_0$, kde x_i^n je n-tý stav neurónu i, x je očakávaný stav tohoto neurónu a ϵ je tolerančná odchýlka. Vo väčšine prípadov nás zaujíma konvergencia výstupných neurónov NN.

[5] váhy NN sa nemenia

[6] mimo učenia, cez rekurentné synaptické prepojenia

4

Kontrolované učenie na FF NN

4.1 Metóda najstrmšieho zostupu

V tejto časti popíšeme základný matematický prístup, ktorý sa v teórii NN považuje za klasický. Ide o prístup k výpočtu zmeny SV v NN cez chybové funkcie **metódou najstrmšieho zostupu**.

4.1.1 Wienerov filter

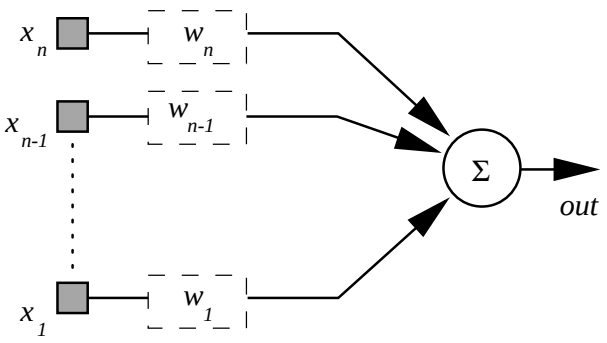
Predstavme si n-senzorov umiestnených na rôznych miestach.(viď obr. 4.1) Tieto senzory produkujú signály $in_1, \ldots, in_n$. Signály sa potom s určitými váhami integrujú do výstupného neurónu. My chceme mať na výstupe nejaký výsledok, ktorý v procese učenia dobre poznáme ev. Teda ak ou je výstup z výstupného neurónu, tak

$$ou = \sum_{k=1}^{n} w_k in_k \tag{4.1}$$

Nech je známy chybový rozdiel

$$e = ev - ou \tag{4.2}$$

Už v úvode je potrebné poukázať na principálnu rozdielnosť voči perceptrónu tým, že kým pri perceptróne na vstup vstupovali vstupy iba z dvoch rôznych tried, tu môžu byť usporiadané dvojice (in, ev) patriace



Obr. 4.1: Schéma Wienerovho filtra

do mnohých skupín a v podstate sa tu nejedná o dichotómiu. Topologický sú z pohľadu NN Wienerov filter a perceptrón veľmi podobné. Teda môžeme definovať všeobecnú chybovú funkciu v tvare

$$J \;=\; 0.5\, E(e^2) \tag{4.3}$$

kde $E(e^2)$ je stredná hodnota kvadrátov všetkých rozdielov. Základným problémom je teda nájsť také $w_1, \ldots, w_n$, pri ktorých $J \to 0$. Takýto filter v oblasti spracovania signálov nazývame **Wienerov filter**. Do určitej miery je pojem **filter** trocha mätúci, ale ide o filtráciu vstupov. Je potrebné to chápať ako vhodné priradenie výstupov k jednotlivým vstupom.

Ak do rovnice (4.3) dosadíme (4.2) resp. (4.1), dostaneme nasledovný výraz:

$$J \;=\; 0.5\, E(ev^2) - E(\sum_{j=1}^{n} w_i in_i ev) + 0.5E(\sum_{j=1}^{n}\sum_{k=1}^{n} w_j w_k in_j in_k) \tag{4.4}$$

Stredná hodnota je lineárna operácia preto môžeme urobiť nasledovné upravy:

$$J \;=\; 0.5\, E(ev^2) - \sum_{j=1}^{n} w_i E(in_i ev) + 0.5\sum_{j=1}^{n}\sum_{k=1}^{n} w_j w_k E(in_j in_k) \tag{4.5}$$

teraz si skúsme popísať jednotlivé funcie v rovnici (4.5) nasledovne

- $E(ev^2) = r_{ev}$ označme ako **strednú hodnotu** očakávaného výsledku ev.
- $E(in_j ev) = r_{in,ev}(j)$ je **kroskorelačná funkcia** medzi vstupmi in a očakávanými výstupmi ev.
- $E(in_j in_k) = r_{in,in}(jk)$ je **autokorelačná funkcia** medzi samotnými vstupmi navzájom

Potom môžeme rovnicu (4.5) prepísať do tvaru

$$J = 0.5 r_{ev} - \sum_{k=1}^{n} w_k r_{in,ev}(k) + 0.5 \sum_{j=1}^{n} \sum_{k=1}^{n} w_j w_k r_{in,in}(jk) \tag{4.6}$$

Teda pri hľadaní váh, ktoré by minimalizovali chybovú funkciu môžeme napísať že

$$\frac{\partial J}{\partial w_l} = -r_{in,ev}(l) + \sum_{j=1}^{n} w_j r_{in,in}(jl) \tag{4.7}$$

potom logicky pri hľadaní váh počítame s podmienkou $\frac{\partial J}{\partial w_l} = 0$ a dostávame z (4.7)

$$\sum_{j=1}^{n} w_j r_{in,in}(jl) = r_{in,ev}(l) \tag{4.8}$$

Potom systém l-rovníc (4.8) o n-neznámych $w_j, l = 1, \ldots, n$ nazývame Wienerovým systémom rovníc a filter s vypočítanými váhami voláme v teórii signálov **Wienerovým filtrom**.

4.1.2 Metóda najstrmšieho zostupu

K tomu, aby sme vedeli vyriešiť rovnice (4.8), by sme potrebovali vypočítať maticu (n x n) a jej inverziu , čo je dosť náročný výpočet[1]. Existuje aj iná forma výpočtu hľadaných SV a to metódou najstrmšieho zostupu (**steepest descent**). Tento vypočet budeme realizovať

[1]Niekedy je výpočet váh cez matice vyhodný, hlavne v prípade, ak počítač, na ktorom realizujeme výpočty, má vektorový procesor a maticové operácie vie realizovať veľmi rýchlo

iteračným spôsobom v t-iteráciách a budeme vypočítavať zmenu SV, ktorú môžeme vyjadriť nasledovne pre synapsiu "i"

$$\Delta w_i(t) \;=\; -\gamma \frac{\partial J(t)}{\partial w_i(t)} \tag{4.9}$$

kde γ je učiaci pomer, potom hľadanú váhu v iterácii "t+1" vypočítame

$$w_k(t+1) \;=\; w_i(t) \;+\; \Delta w_i(t) \tag{4.10}$$

potom z rovníc (4.7) a (4.9) dostaneme z rovnice (4.10) upravený tvar

$$w_i(t+1) \;=\; w_i(t) \;+\gamma(r_{in,ev}(k,t) - \sum_{j=1}^{n} w_j(t) r_{in,in}(jk,t)) \tag{4.11}$$

kde $i = 1, \ldots, n$, n -je počet senzorov. Teda v konečnom dôsledku nájdeme príslušné SV po určitom počte iterácii aj takým spôsobom, avšak výpočtová náročnosť vzorca (4.11) je dosť veľká vzhľadom na funkcie $r_{ev,in}$ a $r_{in,in}$. V prípade, že si graficky zobrazíme chybovú funkciu $J(t)$ v závislosti od jednotlivých váh $w_1, \ldots, w_n$, dostali by sme hyperplochu, ktorá sa nazýva **povrch chybovej funkcie** (**error surface**). Táto zvlnitá hyperplocha má svoje **globálne minimum**, ktoré zodpovedá nejakým $w_1, \ldots, w_n$ a to sú **práve** SV, ktoré pre filter hľadáme. Pod filtrom budeme rozumieť množinu synaptických váh, ktoré hľadáme.

4.1.3 Metóda nejmenšej kvadratickej chyby

Metóda nejmenšieho stredného kvadrátu (**least-mean-square** ďalej LMS) je založená na okamžitých odhadoch funkcii $r_{in,in}$ a $r_{in,ev}$ a to aproximáciou odhadov nasledovnými výrazmi v iterácii "t":

- $$\hat{r}_{in,in}(ji,t) \;=\; in_j(t) in_i(t) \tag{4.12}$$

- $$\hat{r}_{in,ev}(i,t) \;=\; in_i(t) ev(t) \tag{4.13}$$

Tieto odhady $\hat{r}_{in,in;ev,in}$ sa spočítavajú v každej iterácii. Ak toto dosadíme do rovnice (4.11) a budeme uvažovať odhady jednotlivých SV, potom

$$\hat{w}_i(t+1) \;=\; \hat{w}_i(t) \;+\gamma(in_i(t)ev(t) - \sum_{j=1}^{n} \hat{w}_j in_l(t) in_i(t) \tag{4.14}$$

čo môžeme upraviť do tvaru

$$\hat{w}_i(t+1) \;=\; \hat{w}_i(t) \;+\gamma(ev(t) - \underbrace{\sum_{j=1}^{n} \hat{w}_j in_l(t))}_{ou(t)} in_i(t) \tag{4.15}$$

Teda konečný tvar rovnice (4.15) je

$$\hat{w}_i(t+1) \;=\; \hat{w}_i(t) \;+\gamma \underbrace{(ev(t) - ou(t))}_{e(t)} in_i(t) \tag{4.16}$$

čo je pravidlo výpočtu nových hodnôt SV podľa LMS. Teda zhrňujúc môžeme LMS popísať v nasledovných krokoch:

- **inicializácia** $\forall j \;= 1, \ldots, n; w_j(t=0) \;=\; 0$
- **filtrácia** pre $t \;= 1, \ldots$ vypočítavame
  1. $y(t) \;=\; \sum_{j=1}^{n} \hat{w}_j(t) in_j(t)$
  2. $e(t) \;=\; ev(t) \;-\; y(t)$
  3. $\hat{w}_i(t+1) \;=\; \hat{w}_i(t) + \gamma e(t) in_i(t)$ pre $k = 1, \ldots, n$
  4. zvýšime (t+1) a návrat do bodu 1

4.1.4 Adaline

Adaline[2] bol popísaný Widrowom a Hoffom v roku 1960 (viď obr. 4.2). Predstavuje najednoduchšiu NN, topológiou zhodnú s JPR. Principiálny rozdiel je v učiacej funkcii, kým v prípade adaline ide o učenie typu

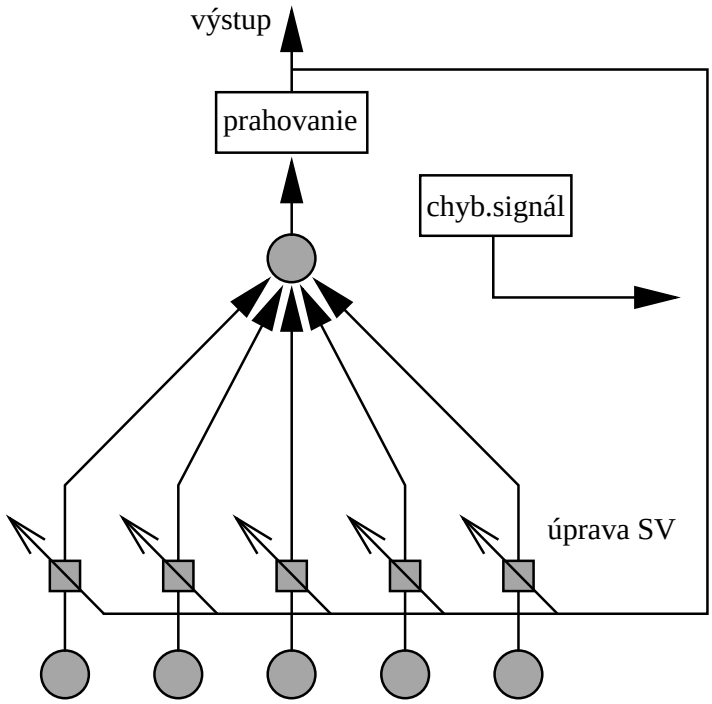
[2] **Ada**ptive **lin**erar **e**lement

LMS a v prípade JPR ide o iné učenie. Vstup do výstupného neurónu má tvar

$$in(t) \;=\; \sum_{j=1}^{n} w_j(t) in_j(t) \tag{4.17}$$

a potom výstup

$$ou(t) \;=\; \begin{cases} 1 & ak\ in(t) \;\geq \theta \\ -1 & ak\ in(t) \;<\; \theta \end{cases} \tag{4.18}$$



Obr. 4.2: Topológia adaline s naznačenou zmenou SV

Podľa metódy LMS potom e(t) (viď vzorec 4.2) môže nadobúdať hodnoty {-2,0,2} čo je zrejmé z možnosti výstupov a možnými rozdielmi medzi nimi.

4.2 Repetitórium č. 1

Bolo by veľmi vhodné, keby ste zodpovedali nasledovné tematické okruhy na základe doposiaľ prečítaných častí. V jednotlivých tutoriáloch sú zoskupené problémy, na ktoré by ste mali vedieť reagovať. V prípade, že neviete komentovať nasledovné tematické okruhy, doporučujeme Vám vrátiť sa k prebraným témam ešte raz.

- **1. tutoriál**

  1. Aké vlastnosti má splňať systém UI?
  2. Aké sú dva základné prístupy pri riešení problémov UI? Popíšte ich. Aká je predpokladaná perspektíva?
  3. Čo je to NN?
  4. Akú významnú vlastnosť majú NN ? Čo dokážu ? Aké sú základné aplikačné oblasti ?
  5. Aké základné okruhy problémov pri štúdiu NN existujú?
  6. Aký je rozdiel medzi NN a ľudským mozgom ? Je ľudský mozog napodobniteľný ?
  7. Ktoré sú základné historické medzníky vo vývoji teórie NN?
  8. Charakterizujte základnú procesnú jednotku **neurón**
  9. Čo je to učenie? Aký je rozdiel medzi činnosťou NN počas a mimo učenia ?
  10. Aké sú základné paradigmy učenia? Aký je rozdiel medzi kontrolovaným a nekontrolovaným učením?
  11. Aké sú základné druhy kontrolovaného učenia, nekontrolovaného učenia a učenia na základe stavu systému?

- **2. tutoriál**

  1. Prečo je nutné hovoriť o stabilite NN a kedy ? Aká je kritetálna funkcia stability?
  2. Aký je rozdiel medzi konvergenciou NN a stabilitou NN?
  3. Aké sú typy úloh riešených pomocou NN?

4. Aká je topológia perceptrónu? Aká je úloha základného perceptrónu a jeho činnosť?
5. Čo vlastne chceme dokázať konvergenciou perceptrónu ?
6. Aký je rozdiel medzi lineárnou a nelinárnou separabilitou ? Čo je to to XOR problém?
7. Môžte komentovať terminologický problém perceptrónu ?
8. Aká je logická podstata Wienerovho filtra ?
9. Je metóda najstrmšieho zostupu cestou k hľadaniu riešení Wienerovho systému rovníc ?
10. Aký je rozdiel medzi metódou najstrmšieho zostupu a metódy najmenšej strednej kvadratickej chyby?
11. Aký je rozdiel medzi adaline a perceptrónom ?

Poznámky

Poznámky

4.3 Delta pravidlo

Delta pravidlo (ďalej DP) predstavuje veľmi dôležitý postup pri výpočte zmeny SV ($\Delta w_{ij}(t)$) a je typu **LMS**. DP rozširuje LMS na prípad ne-McCullochových neurónov[3]. Teda pre jednoduchú jednovrstvovú sieť s M vstupnými neurónmi a jedným výstupným neurónom "i" dostaneme

$$x_i \;=\; f(in_i) \;=\; in_i \;=\; \sum_{j=1}^{M} x_j w_{ij}(t) \;+\; \theta_i \tag{4.19}$$

z rovnice je zrejmé, že ide o lineárnu aktivačnú funkciu v neuróne "i". Potom **chybovú funkciu**, ktorá má charakter LMS cez všetkých N_0 výstupov do NN vyjadríme v prípade, ak výstupná funkcia je identická teda $x_i \;=\; ou_i$

$$J(t) \;=\; \sum_{i=1}^{N_0} J^i(t) \;=\; 0.5 \sum_{i=1}^{N_0} (ev_i(t) \;-\; x_i(t))^2 \tag{4.20}$$

kde $ev(t)$ je očakávaná a $x_i(t)$ vypočítaná hodnota na $i-tom$ vystupe z NN. Metóda LMS hľadá hodnoty zmeny SV pri **minimalizácii** tejto chybovej funkcie. Myšlienka zmeny SV v závislosti od negatívnej parciálnej derivácie chybovej funkcie podľa váhy teda má tvar

$$\Delta w_{ij}(t) \;=\; -\gamma \frac{\partial J(t)}{\partial w_{ij}(t)} \tag{4.21}$$

kde γ je učiaci pomer[4]. Pravú stranu v (4.21) je možné upraviť

$$\frac{\partial J(t)}{\partial w_{ij}(t)} \;=\; \frac{\partial J(t)}{\partial x_i(t)} \frac{\partial x_i(t)}{\partial w_{ij}(t)} \tag{4.22}$$

.

Prvý člen pravej strany sa ďalej rovná

$$\frac{\partial J(t)}{\partial x_i(t)} \;=\; -(ev_i(t) \;-\; x_i(t)) \tag{4.23}$$

[3] niekedy ich nazývame kontinuálne neuróny

[4] alebo koeficient proporcionality

a druhý člen na základe (4.19) sa rovná

$$\frac{\partial x_i(t)}{\partial w_{ij}(t)} \;=\; x_j(t) \tag{4.24}$$

Ak označíme $\delta(t) \;=\; -(ev_i(t) \;-\; x_i(t))$, tak výsledný vzorec pre výpočet zmeny váhy pri ľubovoľnom vstupe má tvar

$$\Delta w_i(t) \;=\; \gamma\delta(t)x_j(t) \tag{4.25}$$

Takto vypočítaná zmena SV podľa **delta pravidla - ďalej DP** dala základ ďalším modifikáciam delta pravidla, čo prispelo k jeho rozšíreniu a aplikácii pri učení NN. Signál $\delta(t)$ nazývame chybovým signálom.

4.4 Metóda spätného šírenia chyby

V predošlej časti matematicky odvodené DP vlastne predstavuje základ **učenia so spätným šírením chyby**[5] a umožňuje použitie v podstate ľubovoľnej aktivačnej funkcie f aj nelineárneho typu, ktorá splňuje podmienku diferencovateľnosti, t.j. platí

$$x \;=\; f(in) \;\neq\; in \tag{4.26}$$

Ide teda znova o určovanie zmeny SV pre NN s nelineárnymi neurónmi. Postup bude analogický ako pri základnom DP, avšak o funkcii f predpokládame, že **nie je lineárna** a je diferencovateľná. Teda opäť stav neurónu "i" pri ľubovoľnom vstupe do NN má tvar

$$x_i(t) \;=\; f(in_i(t)), \tag{4.27}$$

kde

$$in_i(t) \;=\; \sum_{j=1}^{M} w_{ij}(t)x_j(t) \;+\; \theta_i. \tag{4.28}$$

Z predchádzajúceho DP vieme, že

$$\Delta w_{ij}(t) \;=\; -\gamma\frac{\partial J(t)}{\partial w_{ij}(t)}. \tag{4.29}$$

[5]Back-propagation of Error

$J(t)$ má tvar

$$J(t) \;=\; 0.5 \sum_{j=1}^{N_0} (ev_i(t) - x_i(t))^2, \tag{4.30}$$

kde N_0 je počet neurónov vo **výstupnej vrstve NN**. Samotný výpočet parciálnej derivácie chybovej funkcie podľa príslušnej SV má tvar

$$\frac{\partial J(t)}{\partial w_{ij}(t)} \;=\; \frac{\partial J(t)}{\partial in_i(t)} \frac{\partial in_i(t)}{\partial w_{ij}(t)} \tag{4.31}$$

Označme

$$\frac{\partial J(t)}{\partial in_i(t)} \;=\; -\delta_i(t) \tag{4.32}$$

a

$$\frac{\partial in_i(t)}{\partial w_{ij}(t)} \;=\; x_j(t), \tag{4.33}$$

potom dostaneme obvykly zápis výpočtu zmeny SV v tvare

$$\Delta w_{ij}(t) \;=\; \gamma \delta_i(t) x_j(t) \tag{4.34}$$

Základným problémom je teraz stanovenie príslušného δ_i pre **každý neurón** NN. Vedie to k jednoduchému rekurzívnemu vzťahu pre výpočet jednotlivých δ_i, ktoré predstavujú spätné šírenie chyby smerom **od výstupu** NN.

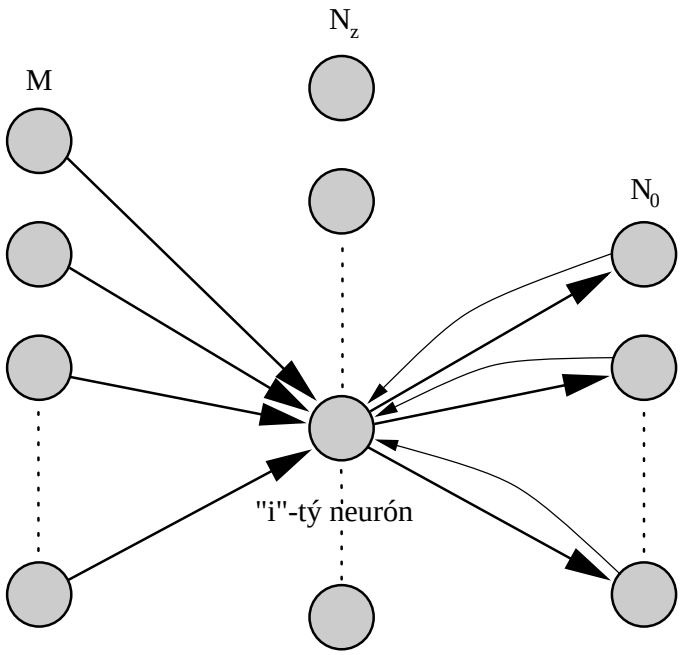
Pre príslušné $\delta_i(t)$ môžeme ďalej písať na základe (4.32)

$$\delta_i(t) \;=\; -\frac{\partial J(t)}{\partial in_i(t)} \;=\; -\frac{\partial J(k)}{\partial x_i(t)} \frac{\partial x_i(t)}{\partial in_i(t)} \tag{4.35}$$

Najprv vyriešme druhý člen pravej strany (4.35). Vzhľadom na nelineárny neurón je zrejmé, že môžeme napísať pomocou (4.27),že

$$\frac{\partial x_i(t)}{\partial in_i(t)} \;=\; f^{'}(in_i(t)) \tag{4.36}$$

Pre výpočet prvého člena z rovnice musíme uvažovať dva rôzne prípady :



Obr. 4.3: Zobrazenie toku chybového signálu z výstupu pre "i"-ty neurón

- ak neurón "i" je **výstupným neurónom** - vtedy je to pomerne jednoduché, lebo hľadaná parciálna derivácia má tvar

$$\frac{\partial J(t)}{\partial x_i(t)} \;=\; -(ev_i(t) - x_i(t)) \tag{4.37}$$

  a tým máme výpočet $\delta_i(t)$ pre tento prípad vyriešený pomocou (4.36) a (4.37) v tvare

$$\delta_i(t) \;=\; (ev_i(t) - x_i(t)) f^{'}(in_i(t)) \tag{4.38}$$

- ak neurón "i" **nie je výstupným neurónom** - výpočet je trocha zložitejší a postupuje sa takto

$$\frac{\partial J(t)}{\partial x_i(t)} \;=\; \sum_{h=1}^{N_0} \frac{\partial J(t)}{\partial in_h(t)} \frac{\partial in_h(t)}{\partial x_i(t)} \tag{4.39}$$

kde N_0 je počet neurónov vo výstupnej vrstve, resp. napravo od "i", čo je znázornené na Obr. 4.3. Z matematického hľadiska pri výpočte derivácie chybovej funkcie J, ktorá popisuje celkovú chybu na **výstupnej vrstve**, podľa $x_i(t)$ [6], je nutné vyjadriť J ako funkciu $x_i(t)$. Preto rovnica (4.39) má takýto tvar. Súčasne prvý člen pravej strany je jasný z rovnice (4.32) a teda platí, že

$$\frac{\partial J(t)}{\partial x_h(t)} = \frac{\partial x_h(t)}{\partial in_h(t)} = -\delta_h(t) \tag{4.40}$$

Tu je potrebné poznamenať, že $x_h(t)$ v rovnici (4.40) je z inej vrstvy ako $x_i(t)$ v rovnici (4.39). Čo sa týka druhého člena rovnice (4.39), tam samotný člen $in_h(t)$ predstavuje vstup do výstupného neurónu "h" a môžeme ho nahradiť nasledovne

$$in_h(t) = \sum_{l=1}^{N_z} w_{hl}(t)x_l(t) \tag{4.41}$$

avšak parciálna derivácia $in_h(t)$ podľa $x_i(t)$ znamená, že jedno z $l = i$ a tým

$$\frac{\partial \sum_{l=1}^{N_z} w_{hl}(t)x_l(t)}{\partial x_i(t)} = w_{hi}(t) \tag{4.42}$$

teda v konečnom dôsledku

$$\frac{\partial J(t)}{\partial x_i(t)} = -\sum_{h=1}^{N_0} \delta_h(t)w_{hi}(t) \tag{4.43}$$

a konečne hľadaný koeficient $\delta_i(t)$ bude mať tvar na základe (4.36) a (4.43)

$$\delta_i(t) = f^{'}(in_i(t)) \sum_{h=1}^{N_0} \delta_h(t)w_{hi}(t) \tag{4.44}$$

Je potrebné dobre si všimnúť **rekurzívnosť** tohoto vzťahu. Ide o výpočet koeficientu $\delta_i(t)$ neurónu "i", ktorý **nie** je výstupným

[6] to je stav neurónu vo vrstve, ktorá nie je výstupná

neurónom. Vypočítame ho za pomoci $\delta_h(t)$, ktoré prichádzajú z vrstvy **napravo** od neurónu "i" a ich počet je N_0 (všimnite si obr. 4.3).

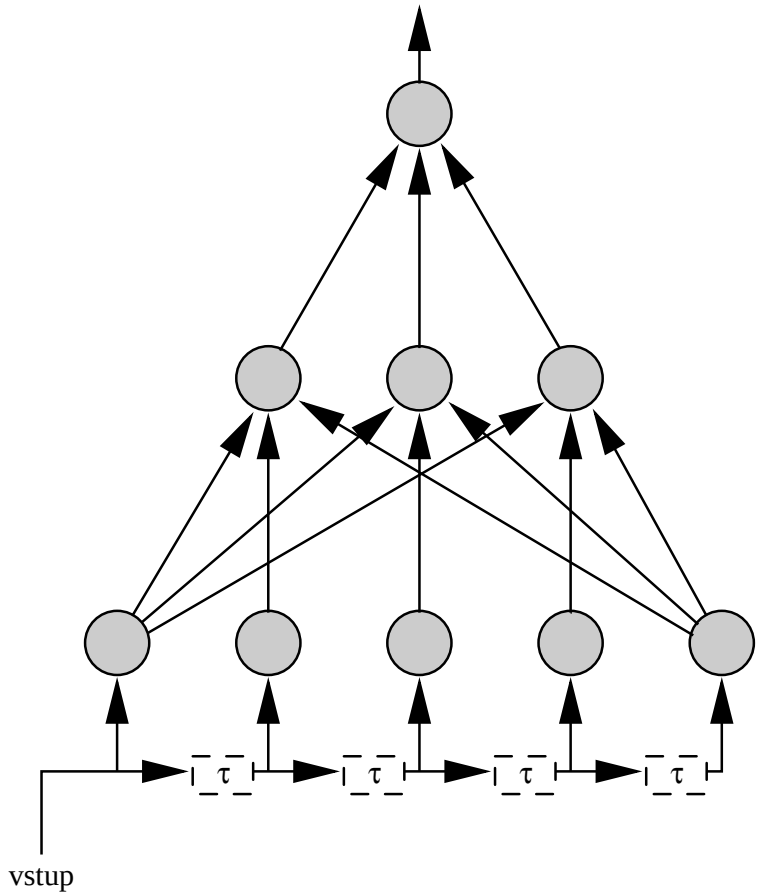
Existuje modifikácia vzťahu (4.44) a to v tvare

$$\delta_i(t) = (f' + c)(in_i(t)) \sum_{h=1}^{N_0} \delta_h(t) w_{hi}(t) \qquad (4.45)$$

kde parameter c je tzv.parameter rovinnosti, ktorý rieši prípad, ak chyba sa nachádza na rovinnej časti chybovej plochy.

4.5 Time-delay na FF NN

Tento prístup predstavuje klasickú metódu BP s upraveným vstupom. Topológia TD FF NN je znázornená na obrázku 4.4, kde (z^{-1}) predstavujú časové oneskorovacie členy. Táto topológia je vhodná pre vstup určitej série meraní, ktoré budú do NN vstupovať ako celok. V podstate ide o vytvorenie klzavého okna nad vstupmi, ktorého šírka predstavuje samotné celkové časové oneskorenie. Matematicky je TD BP málo odlišné od klasickej formy BP. Aplikácie tohoto prístupu sú veľmi časté v rôznych oblastiach napr. v ekonomike, riadení, spracovaní dynamických obrazov a pod.



Obr. 4.4: NN s vstupom časového signálu

4.6 Spôsoby urýchlenia konvergencie BP

Vzhľadom na charakter metódy BP sa hľadali jej modifikácie, ktoré by priniesli kvalitatívne lepšie výsledky a zároveň by urýchlili proces samotného učenia. Uvedieme dva prístupy modifikácie BP, a síce

- použitie BP s momentom
- použitie adaptívnych parametrov učenia pri BP

4.6.1 BP-momentum

Použitie BP s momentom vychádza z poznatkov o chovaní sa siete. Ak vo vzťahu (4.25) pre SV, ktoré prepájajú neuróny "j" a "i" (smeruje do "i"), tj.

$$\Delta w_{ij}(t) \;=\; \gamma\delta_i(t)x_j(t) \tag{4.46}$$

zvolená hodnota γ ja veľmi veľká, dochádza k oscilácii NN. Zase veľmi malá hodnota γ vedie k neúmerne pomalému učeniu. Tlmeniu spomínaných oscilácii môžeme napomôcť, ak pre výpočet zmeny SV v čase (t) nejako započítame aj zmenu SV v čase $(t-1)$ napr. v tvare

$$\Delta w_{ij}(t) \;=\; \gamma\delta_i(t)x_i(t) \;+\; \alpha\Delta w_{ij}(t-1) \tag{4.47}$$

Novopridanú časť $\alpha\Delta w_{ij}(t-1)$ nazývame momentový výraz **(momentum term)** a teda dochádza k zmene vzorca (4.46). Kde α je koeficientom, ktorý volíme. Teda touto úpravou je možné do určitej miery zrýchliť proces učenia BP. Existujú však ešte zložitejšie úpravy samotného BP, ktoré sú efektívnejšie a prinášajú kvalitnejšie výsledky.

4.6.2 Adaptívne parametre učenia NN

Použitie adaptívnych parametrov učenia pri BP predstavuje veľmi progresívny prístup k dosiahnutiu kvalitných výsledkov v skrátenom čase. Tieto parametre môžeme meniť podľa zvolenej stratégie. V ďalšom sú popísané dva prístupy umožňujúce adaptáciu parametrov učenia NN:

- Delta–bar–Delta pravidlo
- Adaptácia parametrov učenia NN pomocou fuzzy logiky

Delta–bar–Delta pravidlo

Toto pravidlo bolo navrhnuté so zámerom urýchlenia a skvalitnenia učebného procesu a je založené na 4 základných **heuristických úvahách**:

- (HU1) každý parameter chybovej funkcie by mal mať vlastný parameter učenia
- (HU2) každý učiaci parameter by mal mať možnosť sa meniť v každej iterácii učenia
- (HU3) v prípade, ak parciálna derivácia chybovej funkcie podľa konkrétnej SV má **rovnaké** znamienko po viacerých iteráciach, potom učiaci parameter pre konkrétnu SV by mal byť **zvýšený**
- (HU4) v prípade, ak parciálna derivácia chybovej funkcie podľa konkrétnej SV má **alternujúce** znamienko po viacerých iteráciach, potom učiaci parameter pre konkrétnu SV by mal byť **znížený**

Tu je nutné si uvedomiť, že už nemáme jeden parameter γ, ale máme γ_{ij} pre každú konkrétnu synapsiu. Teda je tu snaha meniť učiaci parameter pri každej iterácii[7] t. Takýto algoritmus sa nazýva **delta-delta pravidlo**, resp. jeho ďalšia modifikácia **delta-bar-delta pravidlo**. Kým predtým sme hovorili, že LSM využíva tzv. prístup postupného zostupu, tu LSM využíva hodnotenie momentálnej situácie na chybovom priestore v záujme urýchlenia konvergencie NN a zmenšenia chybovej funkcie siete.

Odvodenie modifikácie BP učenia urobíme v 2 krokoch:

1. odvodenie **delta-delta pravidla** - Tu opäť základom je chybová funkcia, ale z prísne matematického hľadiska by sme mali rozlišovať medzi $J(t)$, kde predpokladáme konštantné γ a funkciou $\mathcal{J}(t)$, kde predpokladáme premenlivé $\gamma(t)$ resp. $\mathcal{J}(t) = fcia(w, \gamma)$, kým pôvodná $J(t) = fcia(w)$. V ďalšom budeme $\mathcal{J}(t)$ vyjadrovať pomocou $J(t)$. Matematické vyjadrenie $\mathcal{J}(t)$ a $J(t)$ je rovnaké, pri identickej výstupnej funkcii, teda

[7] pojem iterácia môžeme chápať tiež ako čas resp. čas zmeny

$$\mathcal{J}(t) \;=\; 0.5 \sum_{j=1}^{N_0}(ev_j(t) \;-\; x_j(t))^2 \;=\; 0.5 \sum_{j=1}^{N_0} e_j^2(t) \qquad (4.48)$$

kde $ev_j(t)$ a $x_j(t)$ sú očakávaná a vypočítaná hodnota výstupného neurónu v $t-$tej iterácii. Je nutné si pripomenúť, že

$$\Delta w_{ij}(t-1) \;=\; -\gamma \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \qquad (4.49)$$

Túto zmenu SV použijeme vo vzťahu

$$w_{ij}(t) \;=\; w_{ij}(t-1) \;+\; \Delta w_{ij}(t-1) \qquad (4.50)$$

vzhľadom na to že γ sa bude viazať ku konkrétnej SV, ale bude sa tiež meniť pre každú iteráciu. Teda ak počítame $\Delta w_{ij}(t)$, musíme uvažovať o $\gamma_{ij}(t)$, a tým dostaneme vzorec pre výpočet SV "ij" v iterácii "t" v tvare

$$w_{ij}(t) \;=\; w_{ij}(t-1) \;-\gamma_{ij}(t) \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \qquad (4.51)$$

Súčasne platí, že

$$\gamma_{ij}(t) \;=\; \gamma_{ij}(t-1) \;+\; \Delta\gamma_{ij}(t) \qquad (4.52)$$

Je potrebné si dobre uvedomiť indexy, ak porovnáme rovnice (4.50) a (4.52). Súčasne môžeme vstup do neurónu "i" vyjadriť

$$in_i(t) \;=\; \sum_{j=0}^{M} w_{ij}(t) x_j(t) \;+\; \theta_i \qquad (4.53)$$

Na základe horeuvedených rovníc sme zistili, že

- $\mathcal{J}(t)$ je funkciou $x_i(t)$
- $x_i(t)$ je samozrejme funkciou vstupu $in_i(t)$
- samotný vstup $in_i(t)$ je funkčne závislý cez SV na $\gamma_{ij}(t)$ cez $w_{ij}(t)$.

Tieto poznámky nám pomôžu pochopiť nasledujúcu rovnicu. Naším globálnym cieľom je nájsť vzťah zmeny γ_{ij} v nejakej iterácii "t" v závislosti od minimalizácie chybovej funkcie $\mathcal{J}(t)$ teda môžeme písať :

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \underbrace{\frac{\partial \mathcal{J}(t)}{\partial x_i(t)}}_{A} \underbrace{\frac{\partial x_i(t)}{\partial in_i(t)}}_{B} \underbrace{\frac{\partial in_i(t)}{\partial \gamma_{ij}(t)}}_{C} \tag{4.54}$$

Jednotlivé členy A, B, C vo vzťahu (4.54) môžeme vypočítať nasledovne

- **člen A** je zrejmý z rovnice (4.48) pre chybovú funkciu

$$\frac{\partial \mathcal{J}(t)}{\partial x_i(t)} = -(ev_i - x_i(t)) = - e_i(t) \tag{4.55}$$

- **člen B** je zrejmý z definície aktivačnej funkcie teda

$$\frac{\partial x_i(t)}{\partial in_i(t)} = f^{'}(in_i(t)) \tag{4.56}$$

- **člen C** ak do rovnice (4.53) dosadíme zrejmú rovnicu

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t-1) \tag{4.57}$$

dostaneme výraz pre pre $in_i(t)$

$$in_i(t) = \sum_{j=1}^{M} \underbrace{(w_{ij}(t-1) - \gamma_{ij}(t)\frac{\partial J(t-1)}{\partial w_{ij}(t-1)})}_{w_{ij}(t)} x_j(t) + \theta_i \tag{4.58}$$

keď predošlú rovnicu zderivujeme podľa $\gamma_{ij}(t)$, dostaneme

$$\frac{\partial in_i(t)}{\partial \gamma_{ij}(t)} = -x_j(t)\frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \tag{4.59}$$

teda v konečnom dôsledku dostaneme

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \underbrace{-e_i(t)f^{'}(in_i(t))}_{A+B} \underbrace{-x_j(t)\frac{\partial J(t-1)}{\partial w_{ij}(t-1)}}_{C} \tag{4.60}$$

Teraz pomocou (4.34) môžeme vyjadriť

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = \delta_i(t)x_j(t) \tag{4.61}$$

potom za $\delta_i(t)$ môžeme dosadiť z (4.37) a skutočne dostaneme za $\frac{\partial J(t)}{\partial w_{ij}(t)}$ výrazy $A+B$ a teda

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = -\frac{\partial J(t)}{\partial w_{ij}(t)}\frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \tag{4.62}$$

a v konečnom dôsledku môžeme napísať pre zmenu $\gamma_{ij}(t)$ výraz

$$\Delta\gamma_{ij}(t) = -\varphi\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \varphi\frac{\partial J(t)}{\partial w_{ij}(t)}\frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \tag{4.63}$$

kde φ predstavuje kladnú konštantu tzv. kontrolný parameter zmeny učiaceho parametra v jednotlivých iteráciach. Horeuvedený vzťah reprezentuje tzv. **delta-delta D-DP** pravidlo učenia NN. Po úvahe prichádzame k záveru, že D-DP splňuje HU3 a HU4. Problémy však nastávajú s presným určením hodnoty φ, čo predstavuje veľmi citlivý koeficient. Preto sa prikročilo k ďalšej modifikácii D-DP.

2. ďalšia modifikácia D-DP vychádza z hore uvedeného a bola navrhnutá v [7] v tvare pre zmenu $\gamma_{ij}(t)$ v tvare

$$\Delta\gamma_{ij}(t) = \begin{cases} \kappa & ak\ S_{ij}(t-1)D_{ij}(t) > 0 \\ -\beta\gamma_{ij}(t) & ak\ S_{ij}(t-1)D_{ij}(t) < 0 \\ 0 & inak \end{cases} \tag{4.64}$$

kde S a D definujeme ako

$$S_{ij}(t-1) = (1-\zeta)D_{ij}(t-1) + \zeta S_{ij}(t-2) \tag{4.65}$$

a

$$D_{ij}(t) \;=\; \frac{\partial J(t)}{\partial w_{ij}(t)} \tag{4.66}$$

kde v rekurzívnom vzťahu pre S uvažujeme $S_{ij}(0) \;=\; 0$. Horeuvedené vzťahy považujeme za **delta-bar-delta pravidlo** učenia. Podobnosť D-bar-DP s D-DP je vyjadrené vo vzťahu (4.64), kde vlastne ovplyvňujeme výpočet $\Delta\gamma_{ij}(t)$ iteráciou t a $t-1$. Súčasne sa naplňujú požiadavky HU1 a HU2, pretože každý prípad vo vzťahu (4.64) má svoj vlastný koeficient κ a β. Je zrejmé, že celý proces adaptácie samotného γ začne v $t = 2$.

Vzhľadom na dosť komplikovaný výpočet je nutné uvažovať o zmene iteračného prístupu. Kým v obyčajnom BP za iteráciu považujeme každý vstup a stále po ňom urobíme zmeny Δw_{ij} v NN, teraz pod iteráciou budeme rozumieť epochu $B, B \;>\; 0$-vstupov do NN a **až potom** urobíme zmenu SV. Tomuto prístupu hovoríme dávkové učenie (**Batch** learning). Potom

$$\frac{\partial J(t)}{\partial w_{ij}(t)} \;=\; -\sum_{k=1}^{B} \delta_i^k(t) x_j^k(t) \;=\; D_{ij}(t) \tag{4.67}$$

a teda zmena SV potom bude mať tvar

$$\Delta w_{ij}(t) \;=\; \alpha \Delta w_{ij}(t-1) \;+\; \gamma_{ij}(t) \sum_{k=1}^{B} \delta_i^k(t) x_j^k(t) \tag{4.68}$$

a nová hodnota SV bude vypočítana podľa obvyklého vzťahu

$$w_{ij}(t+1) \;=\; w_{ij}(t) \;+\; \Delta w_{ij}(t) \tag{4.69}$$

Na záver pripomíname, že pre praktický výpočet musíme pre každú $w_{ij}(t)$ najprv vypočítať príslušný parameter $\gamma_{ij}(t)$. **Je nutné poznamenať, že jednotlivé derivácie na konci vzťahu (4.63) sú v každom prípade spočítavané aj pri klasickom BP**. Teda tu ide iba o výpočet novej hodnoty γ_{ij} z jednotlivých derivácií, ktoré tak či tak by sme v klasickom prístupe BP vypočítali. Z toho vyplýva, že z výpočtového hľadiska nejde o výpočtové úkony naviac. Negatívom však je, že pre modifikovaný BP-prístup založený na D-bar-DP máme až 5 parametrov a to

- $\gamma_{ij}(0)$ - štartovacie γ
- koeficient α pre výpočet $\Delta w_{ij}(t)$
- koeficienty κ, β, ζ pre výpočet $\Delta\gamma_{ij}(t)$

Tento fakt sťažuje použitie tejto metódy, hoci na druhej strane predstavuje systém, ktorý je možné voľnejšie riadiť.

Adaptácia parametrov učenia NN pomocou fuzzy logiky

Fuzzy logika sa interaguje s NN v troch základných oblastiach:

- použitie NN na generovanie fuzzy systému
- priame prepojenie fuzzy systému s NN
- využitie fuzzy logiky na riadenie procesu učenia NN
- simulovanie činnosti fuzzy regulátora pomocou NN

Z doteraz uvedeného by malo byť zrejmé, že proces učenia NN je najdôležitejším problémom v činnosti NN. Zvýšenie jeho efektivity pre dosiahnutie GS NN je cieľom mnohých metód na dosiahnutie konvergencie NN. V tejto časti je popísaný prístup [17], ktorý bol experimentálne overený. Fuzy logika dnes už prestavuje samostatný vedný odbor a poskytla vhodné matematické prostriedky na prácu s lingvistickými premennými. Veľmi vhodnou publikáciou popisujúcou fuzzy množiny je [12]. Majme množinu U,A a množinu L.Nech $\mathcal{A} \in \mathcal{U}$. Potom definujme funkciu μ a to nasledovne

$$\mu \ : \ \mathcal{A} \in \mathcal{U} \ \rightarrow \ \mathcal{L}$$

kde L je možina definovaná na celom intervale $< 0, 1 >$. Potom funkciu μ nazývame **funkciou príslušnosti fuzzy množiny A** , ktorá každému prvku z množiny A priradí hodnotu z množiny L. Ak $\mu_{\mathcal{A}}(x) = 0$ tak $x \notin A$ a naopak ak $\mu_{\mathcal{A}}(x) = 1$ tak určite platí,že $x \in A$. K tomu, aby sme zvládli túto časť, pripomeňme si základné operácie na fuzzy množinách, ktoré budeme využívať a to **zjednotenie a prienik** dvoch fuzzy množín A,B. Tieto operácie definujeme nasledovne:

- zjednotenie A,B je definované ako

$$\mu_{\mathcal{A}\cup\mathcal{B}}(x) \;=\; max(\mu_{\mathcal{A}}(x), \mu_{\mathcal{B}}(x))$$

- prienik A,B je definovaný ako

$$\mu_{\mathcal{A}\cap\mathcal{B}}(x) \;=\; min(\mu_{\mathcal{A}}(x), \mu_{\mathcal{B}}(x))$$

Teraz, keď sa vrátime k našej chybovej funkcii $\mathcal{J}(t)$, chceme na základe nej modifikovať γ. Teda ide o niečo podobné ako v prípade D-Bar-DP, len tu to realizujeme pomocou fuzzy logiky. Celý postup bol overený v [17] pre riadenie metódy BP. Pre riadenie celého procesu boli zadefinované nasledovné koeficienty:

- zmena chybovej funkcie v "po sebe nasledujúcich" iteráciách $CE(t)$

$$CE(t) \;=\; \mathcal{J}(t) \;-\; \mathcal{J}(t-1)$$

- pomer hodnôt chybovej funkcie v "po sebe nasledujúcich" iteráciách $QCE(t)$

$$QCE(t) \;=\; \frac{\mathcal{J}(t)}{\mathcal{J}(t-1)}$$

- a taktiež výpočet koeficienta ξ, pomocou ktorého môžeme vypočítať novú hodnotu γ v tvare

$$\gamma(t+1) \;=\; \xi(t)\gamma(t)\ .$$

Spomínaný koeficient $\xi(t)$ bude vlastne výsledkom snaženia celého procesu adaptácie v kroku t. Úvahy, ako vlastne použiť spomínané koeficienty, sú podporené nasledovnými tvrdeniami:

- ak je $\mathcal{J}(t)$ veľká, znamená to, že sme **ďaleko** od globálneho minima na povrchu chybovej funkcie $\mathcal{J}(t)$. To znamená, že parameter $\gamma(t)$ môže byť veľký.

- zmena chyby $CE(t)$ reprezentuje dôležitý príznak, ako sa vyvíja situácia pri hľadaní minima. V prípade, že $CE(t)$ je veľké, $\gamma(t)$ môžeme zvyšovať. V prípade, že $CE(t) \;<\; 0$ znamená, že minimum na povrchu bolo prekročené.

- spomínané minutie minima nie je veľmi vhodné, a preto je dobré riadiť zmenu γ pomocou dvoch koeficientov $CE(t)$ a $QCE(t)$

K spomínanému riadeniu adaptácie $\gamma(t)$ je nutné, aby boli známe funkcie príslušnosti $CE(t)$ a $QCE(t)$ pre určenie hľadaného $\xi(t)$, ktorý použijeme pre výpočet novej hodnoty $\gamma(t+1)$. (viď grafy pre $CE(t)$, $QCE(t)$ a $\xi(t)$) Tieto koeficienty mali definované nasledovné lingvistické premenné:

- **CE**(**t**) : veľmi malé, malé, stredné, veľké, veľmi veľké ;
- **QCE**(**t**): malé, stredné, veľké, veľmi veľké ;
- ξ(**t**): malé, stredné, veľké ;

Samotná rozhodovacie tabuľka vyzerá nasledovne:

	QCE			
CE	malé	stredné	veľké	veľmi veľké
veľmi malé	malé	malé	stredné	veľké
malé	malé	stredné	stredné	veľké
stredné	malé	stredné	stredné	veľké
veľké	malé	stredné	veľké	veľké
veľmi veľké	malé	stredné	veľké	veľké

Výsledné hodnoty $\xi(t)$ sú v tabuľke, t.j. ak $CE(t) = stredné$ a $QCE(t) = veľmi\ veľké$, potom hľadané $\xi(t)$ bude **veľké**. Potom podľa grafu určíme presnú hodnotu $\xi(t)$.

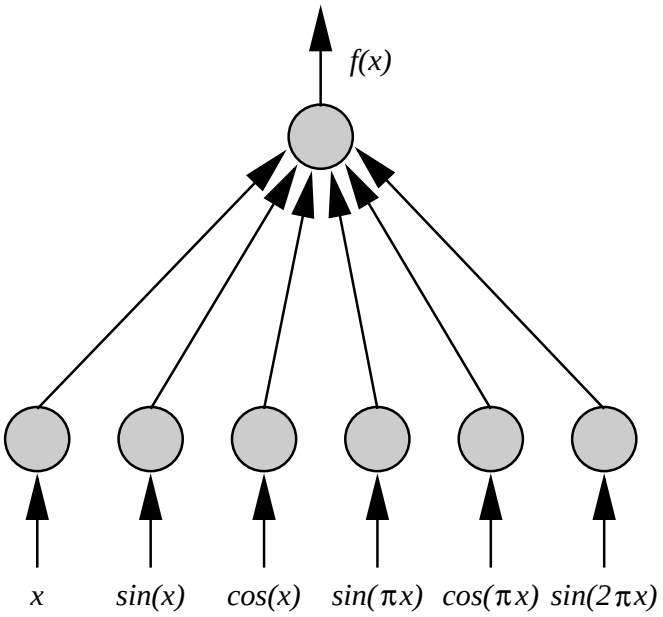
4.7 Funkcionálne linky v NN

V [13] Pao zaviedol tzv.**funkcionálne linky**. Ide o ľubovolnú transformáciu vstupu **in** na niekoľko vstupov napr.:

$$in^2,\ sin(in),\ cos(in),\ sin(\pi in),\ cos(\pi in)$$

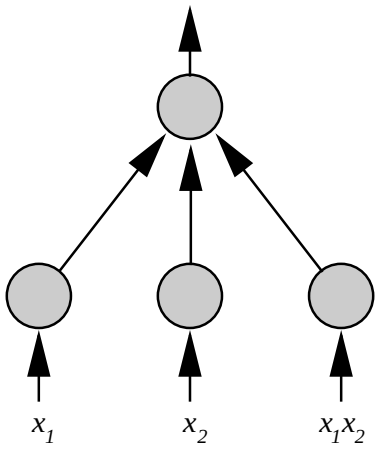
a podobne viď obrázok 4.5. NN s funkcionálnymi linkami sa taktiež nazývajú NN **vyšších rádov**. V prípade použitia takéhoto prvku je

možné do určitej miery zjednodušiť NN. Teda napr. známy XOR problém sa dá pomocou funkcionálnej linky vyriešiť bez skrytej vrstvy (viď obr. 4.6).



Obr. 4.5: Príklad NN vyššieho rádu

Problematika výberu funkcií namiesto pôvodného vstupu nie je deterministicky určená. V [13] je uvedená vhodnosť použitia týchto sietí oproti klasickým NN. Je známe použitie týchto NN v energetike [18] a v spracovaní obrazov .



Obr. 4.6: XOR problém riešený pomocou funkcionálnej linky

4.8 Dôležité poznámky k návrhu FF NN

V tejto časti sú uvedené základné poznámky, ktoré môžu ovplyvniť efektívnosť hľadania optimálnej topológie NN a parametrov učiaceho procesu. Je treba podotknúť, že tieto pravidlá platia pri **aproximáciach funkcií**, teda pri **kontrolovanej forme učenia a FF NN**. Z doposiaľ uvedeného môžeme špecifikovať nasledovné okruhy problémov:

1. návrh topológie NN
2. problém inicializácie NN
3. problém veľkosti trénovacej množiny pre aproximačné ciele NN

4.8.1 Návrh topológie NN

V prípade aproximácie funkcie celý proces v konečnom dôsledku musí viesť k aproximácii s nejakou presnosťou. Je to vlastne niečo podobné ako aproximácia funkcie nejakým radom. Ak si predstavíme jednoduchú NN s jedným vstupom a jedným výstupom pri predpoklade, že aktivačné funkcie v skrytej vrtsve sú nelineárne a vo vstupe a výstupe je aktivačnou funkciou funkcia identity. Teda na výstupe zo vstupných neurónov dostaneme

$$ou(t) \;=\; \sum_{j=1}^{M} w_j(t)x_j(t) - \theta \tag{4.70}$$

a po prechode skrytou vrstvou na jej výstupe dostaneme

$$ou(t) \;=\; \sum_{j=1}^{N} w_j(t) f(\sum_{l=1}^{N} w_l(t)x_l(t) - \theta_{in}) - \theta_h \tag{4.71}$$

a tak ďalej. Teda je položená otázka, ako ďaleko je potrebné pokračovať v rozvoji aproximácie funkcie v zmysle (4.71). Odpoveď na túto otázku dala matematická analýza celého problému v práci [3], ktorá znela **NN s jednou, resp. max dvoma skrytými vrstvami je schopná aproximácie ľubovoľnej funkcie** s dostatočnou presnosťou. Je zrejme, že viac ako 2-vrstvové NN to vedia tiež, ale za cenu vyššej zložitosti NN. Existuje tzv. **Univerzálna aproximačná**

teoréma popísaná tiež v [5], ktorá tvrdí, že stačí jedna skrytá vrstva na aproximáciu ľubovoľnej funkcie. Z praktického hľadiska, však nehovorí koľko neurónov má mať táto jediná skrytá vrstva. Ak budeme predpokladať, že pripustíme aj podľa [3] dve skryté vrstvy, môžeme naše úvahy zotriediť do dvoch bodov:

- každá "rozumná" funkcia môže byť vyjadrená ako lineárna kombinácia lokálnych **zvlnení**
- každé takéto zvlnenie môžeme s dostatočnou presnosťou aproximovať **dvoma** skrytými vrstvami

Otázka, akú úlohu má v tom prípade 1. resp. 2. vrstva môže byť vyjadrená nasledovne podľa [5] :

- **prvá skrytá** vrstva nachádza **lokálne príznaky** aproximovanej funkcie
- **druhá skrytá** vrstva spracováva výstup z prvej skrytej vrstvy a jej úlohou je určovať **globálne príznaky** aproximovanej funkcie

Z praktického hľadiska návrhu topológie NN je vhodné sa pri kontrolovanom učení a FF NN obmedziť na NN, ktoré obsahujú **najviac dve skryté vrstvy**. Tým sa hľadanie optimálnej topológie zefektívni a priestor možností rôznych topológii FF NN sa zmenší. Otázka ale stále ostáva, ako aj pri tomto obmedzení máme pristupovať k návrhu topológie FF NN, aby sme čo v najkratšom čase dostali uspokojivé výsledky. V literatúre sú známe dva principiálne prístupy a to :

1. konštrukčný algoritmus ([6])
2. eliminačný algoritmus (tzv. pruning [5][8]

V prvom prípade ide o postupné budovanie NN, nahrádzanie jedného neurónu dvoma až po nahradenie jednej skrytej vrstvy dvoma vrstvami. Je to algoritmus postupného budovania až po nájdenie optimálnej topológie NN. Samozrejme v každej skúmanej topológii musíme nastavovať parametre učenia, takže ide o dosť náročný proces.

[8]Niektoré prístupy "pruningu" sú súčasťou SNNSv3.3

Eliminačný prístup naproti tomu začína s tzv. maximálnou verziou topológie NN a postupnými elimináciami neurónov alebo SV dochádza k optimálnej topológii NN. K tejto forme hľadania optimálnej topológie NN existujú špeciálne prístupy napr. Optimal Brain Damage, Optimal Brain Surgeon[9] a pod.

4.8.2 Problém inicializácie NN

V prípade BP sa niekedy stáva, že nastane **saturácia NN**, t.j. neuróny sa zahltia a NN produkuje konštatný výsledok. K tomu, aby sme zabránili podobným situáciam je možné uvažovať o nasledovných doporučeniach:

- inicializácia NN by mala byť realizovaná generátorom rovnomerného náhodného rozdelenia z dostatočne malého intervalu
- ak počet neurónov v skrytej vrstve je veľmi veľký, saturácia je pravdepodobnejšia ako v prípade malého počtu neurónov
- pri lineárných neurónoch dochádza k saturácii zriedka.

V ideálnom prípade, by sme mali realizovať inicializáciu v závislosti od počtu vstupov do "i"-teho neurónu, lebo je rozdiel, keď inicializujeme SV pre neurón "i", ktorý má napr. 50 vstupov a neurón "j", ktorý má napr. 2 vstupy. Teda ideálna inicializácia by bola, keby SV vstupujúce do každého neurónu boli inicializované z rôznych intervalov podľa počtu vstupov do neurónu. V [5] je uvedený empirický interval pre inicializačné hodnoty v tvare

$$(\frac{-2.4}{NI_i}, \frac{2.4}{NI_i}) \tag{4.72}$$

kde NI_i počet vstupov do neurónu "i".

Vo všeobecnosti inicializácia nemá priamy vplyv na konečný výsledok učenia, ale do veľmi veľkej miery vie ovplyvniť efektívnosť a samotný priebeh učenia. Ide o stanovenie počiatku na chybovom povrchu, od ktorého sa hľadá minimum korešpondujúce s konvergenčným stavom NN.

[9] tieto nájdete v SNNSv3.3

4.8.3 Problém stanovenia veľkosti trénovacej vzorky

Na to, aby sme s dostatočnou presnosťou aproximovali zadanú funkciu, je veľmi vhodné mať dostatočne veľkú trénovaciu vzorku. Samozrejme, ide o prípady, keď si môžeme dovoliť mať veľkosť vzorky meraní takú, ako chceme, čo v praxi nie je vždy možné. Podľa [5], ak chceme dosiahnúť $\frac{\epsilon}{2}$-vú presnosť, je potrebné uvažovať o veľkosti trénovacej množiny

$$n \geq \frac{32W}{\epsilon} ln(\frac{32m}{\epsilon}) \tag{4.73}$$

kde W je počet SV v NN a m je celkový počet nerónov v NN. V podstate po zjednodučení predchádzajúceho vzťahu môžeme uvažovať o počte

$$n > \frac{W}{\epsilon} \tag{4.74}$$

V prípade, že je možné získať taký počet prvkov trénovacej vzorky, potom je vhodné v záujme efektívnosti činnosti NN a presnosti aproximácie skúsiť sa priblížiť týmto počtom trénovacej vzorky pre učebný proces.

4.9 Repetitórium č. 2

- **3. tutoriál**

1. Aká je logika a cieľ Delta pravidla ?
2. Aký je rozdiel medzi Delta pravidlom a zovšeobecneným Delta pravidlom - ZDP (metódou spätného šírenia chyby) ?
3. Je odvodenie zmeny SV rovnaké vo všetkých častiach NN?
4. Odvoďte ZDP pre vybranú aktivačnú funkciu !
5. Vysvetlite prístupy k urýchleniu konvergencie BP-učenie; Prečo chceme vlastne urýchlovať učenie NN? Čo sú heuristické pravidlá ?
6. Aký je rozdiel medzi funkciami $\mathcal{J}$ a J v odvádzaní Delta-bar-delta pravidla ?
7. Ako je možné použiť fuzzy logiku na urýchlenie BP učenia?
8. Kde sa dajú využiť time-delay NN?
9. Aké sú vaše komentáre na nasledovné problémy pri návrhu a činnosti NN?
   (a) Akou topológiou začať?
   (b) Ako hladať optimálnu topológiu ? Koľko je potrebných skrytých vrstiev NN?
   (c) Čo znamená univerzálna aproximačná teória?
   (d) Aká by bola ideálna forma inicializácie?
   (e) Má veľkosť trénovacej množiny význam pri kontrolovanom učení?

Poznámky

Poznámky

5

Nekontrolované učenie na FF NN

V prípade nekontrolovaného učenia my vlastne poskytujeme NN **len vstupy** a očakávame, že samotná NN s tými vstupmi niečo urobí. V podstate nevieme výsledok spracovania, resp. nevieme ho presne. Aj v tomto prípade pojem **učenia** je spojený so zmenou SV v NN. Samotné učenie sa skončí, ak sa NN dostane do stavu **GS** (nie konvergencie[1]) a zmena SV $\rightarrow$ 0. Tento moment je spojený s neurodynamikou učebného procesu, ktorý by mal mať vlastnosti procesu končiaceho v stave GS. V tomto prípade je neurodynamická stránka problému veľmi dôležitá, lebo môže dôjsť k situácii nekonečného učenia. Existujú však prípady, kedy NN osciluje v nejakých pevných bodoch, potom aj toto chovanie je považované za GS systému a prejav NN ako oscilátora a generátora "rozumných" výsledkov.

Základné skupiny úloh, ktoré sa môžu riešiť pomocou FF NN s nekotrolovaným učením sú:

- zhluková analýza
- kvantovanie vektorov, zhusťovanie dát
- zníženie dimenzie[2] v príznakovom priestore

Pripomeňme si niektoré princípy nekontrolovaného učenia:

[1] o konvergencii nemá význam hovoriť pri nekontrolovanom type učenia
[2] metóda hlavných komponent

1. **Modifikácia SV vedie ku seba-zosilneniu**. Tu je vhodné si pripomenúť princípy tzv. rozšíreného Hebbovho učenia spomenuté v 2. kapitole vo vzťahu (3.7). tento princíp sa vo veľkej miere používa aj pri tomto type učenia.

2. Vzhľadom na cieľ zosilňovania SV **nie je možné**, aby všetky SV boli zosilnené, a preto dochádza ku **konkurenčnému chovaniu** neurónov. Tento prístup sa plne zobrazí v konkurenčnom učení, ked sa upravujú **iba tie** SV,ktoré smerujú k **víťaznému neurónu**.

3. Pri konkurenčnom učení sa po určitom čase prejavujú prvky **kooperačného správania sa**. Tento fakt je založený na poznaní, že ak nejaká SV je výrazne silná, tak potom postupne sa zosilňuje aj jej okolie v zmysle Hebbovho adaptačného prístupu.

5.1 Konkurenčné učenie

Konkurenčné učenie predstavuje jednoduchý prístup nekontrolovaného učenia k spracovaniu dát za účelom zhlukovej analýzy. Topológia pre konkurenčné učenie je veľmi jednoduchá (vid obrázok 5.1). Teda na vstupnej vrstve je M-neurónov a na výstupe N_c neurónov. Pri konkurenčnom učení môžeme zhrnúť úvahy do nasledujúcich bodov:

1. výstupné neuróny majú aktivačnú funkciu - funkciu identity. Potom môžeme napísať, že

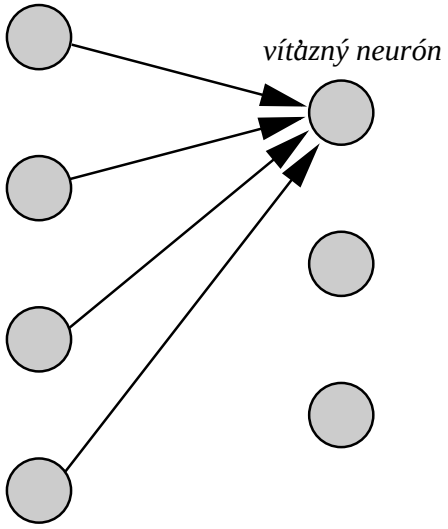
$$ou_i(t) \;=\; \sum_{j}^{M} w_{ij}(t)x_j(t) \tag{5.1}$$

   Ku vzťahu (5.1) sa vrátime v ďalšom **dôležitou** poznámkou.

2. vyberieme **maximálnu hodnotu** $ou_i(t)$ z $i = 1, \ldots, N_c$ a urobíme nasledovnú operáciu

$$ou_i(t) \;=\; \begin{cases} 1 & ak\ ou_i \;=\; max\{ou_i, \forall\ i = 1, \ldots, N_c\} \\ 0 & inak \end{cases} \tag{5.2}$$

   teda neurón, ktorý má najvyššiu hodnotu bude nastavený na **1** a ostatné budú nastavené na **0**. Túto procedúru nazývame **víťaz berie všetko (winner takes all - v ďalšom WTA)**.



Obr. 5.1: Zmena váh, ktoré smerujú k **víťaznému** neurónu

3. Ďalším krokom bude zmena SV,**ale iba tých**, ktoré vstupujú do víťazného neurónu. Odvodenie adaptačného pravidla pre zmeny SV je analogické ako pri kontrolovanom učení a LMS prístupe. Kľúčovým momentom je stanovenie chybovej funkcie. V **konkurenčnom učení** ide o zámer, aby sa jednotlivé vstupy **x premietli** do hodnôt SV, ktoré smerujú k **víťaznému** neurónu, ktorý prísluší vstupu **x**. Teda v konečnom dôsledku budú mať rôzne vstupy **x** rôzne víťazné neuróny a stredy rôznych zhlukov by sa mali premietnuť do hodnôt SV, ktoré smerujú k jednotlivým víťazným neurónom. V podstate budeme mať N_c chybových funkcií typu[3] :

$$J(t) \;=\; 0.5 \sum_{j=1}^{M} \left(w_{ij}(t) - x_j(t)\right)^2 \tag{5.3}$$

Teda ide o rozdiel SV, ktoré smerujú k $i-$temu víťaznému neurónu a hodnoty vstupného neurónu. Teda minimalizáciou tohoto

[3] $\mathbf{x}_j(t)$ je vstup !!!

rozdielu, dochádza k premietnutiu vstupov do hodnôt SV, ktoré smerujú k jednotlivým víťazným neurónom.

Samotné odvodenie zmeny SV bude jednoduché, teda :

$$\Delta w_{ij}(t) \;=\; -\gamma \frac{\partial J(t)}{\partial w_{ij}(t)} \tag{5.4}$$

kde γ je opäť učiaci pomer. Potom vypočítame spomínanú deriváciu nasledovne

$$\frac{\partial J(t)}{\partial w_{ij}(t)} \;=\; \begin{cases} (w_{ij}(t) - x_j(t)) & ak\ "i"\ je\ víťazný\ neurón \\ 0 & inak \end{cases} \tag{5.5}$$

Na základe (5.5) dostaneme zmenu SV v tvare

$$\Delta w_{ij}(t) \;=\; -\gamma (w_{ij}(t) - x_j(t)) \tag{5.6}$$

a z toho dostaneme vzťah pre novú hodnotu SV v tvare

$$w_{ij}(t+1) \;=\; w_{ij}(t) + \gamma (x_j^k(t) - w_{ij}(t)) \tag{5.7}$$

v prípade ak "i"-ty neurón je víťazný.

4. problémom ostáva **inicializácia** váh. Vo všeobecnosti sa to realizuje tým spôsobom, že sa náhodne vyberie N_c vektorov zo vstupnej množiny a tieto sa použijú ako štartovacie hodnoty SV.

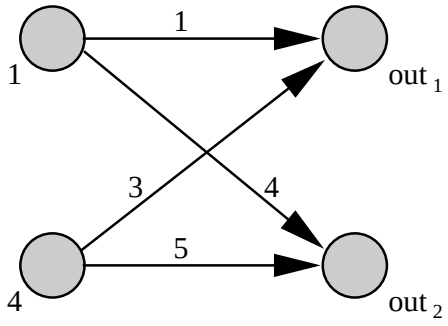
5. ak sa vrátime ku vzorcu (5.1), tak v podstate ide o skalárny súčin dvoch vektorov $\mathbf{w}_i(t)$ a $\mathbf{x}(t)$, ktorý má tvar :

$$\mathbf{w}(t)\mathbf{x}(t) \;=\; |\mathbf{w}\ |\ |\mathbf{x}\ |\ cos(\phi) \tag{5.8}$$

kde ϕ je uhol medzi vektormi $\mathbf{w}(t)$ a $\mathbf{x}(t)$. V prípade, že sú $|\mathbf{w}(t)|$ a $|\mathbf{x}(t)|$ rovné "**1**", potom samotný $cos(\phi)$ sa rovná[4]

$$cos(\phi) \;=\; \sum_{j}^{M} w_{ij}(t) x_j(t) \tag{5.9}$$

Ak sú $|\mathbf{x}(t)|$ rovné "**1**", sú normalizované. Potrebu normalizácie si ozrejmime na nasledujúcom prípade. Vzhľadom na zhlukovací



Obr. 5.2: Príklad jednoduchého zhlukovania do 2 zhlukov v dvojrozmernom príznakovom priestore

charakter celej procedúry je potrebné dať pozor na prípady, keď sú vektory nesprávne zaradené, resp. vyhráva **nesprávny** neurón. Majme situáciu na obr. 5.2.

Ak na vstup dáme vektor **(1,4)** a SV majú také hodnoty, ako je naznačené na obrázku, potom na základe vzťahu (5.1) dostaneme pre **ou1** $=$ **13** a pre **ou2** $=$ **24**, teda podľa stratégie **WTA** vyhráva neurón **2**. Ak si pripomenieme, že vlastne naším cieľom je premapovať vstupy do hodnôt SV, dostali sme výsledok ktorý hovorí, že vektor **(1,4)** má bližšie k vektoru **(4,5)** ako k vektoru **(1,3)** - čo je logicky nesprávne. Túto situáciu riešime pomocou **normalizácie** vstupných vektorov a aj hodnôt SV. A potom skutočne môžeme použiť vzorec (5.5) a dostaneme lepšie výsledky. Tu treba dať pozor aký normalizačný prístup zvolíme. Napríklad ak zvolíme :

$$\mathbf{x_{nor}} = \frac{\mathbf{x}}{\sqrt{\sum_{i=1}^{M} x_i^2}} \tag{5.10}$$

a to isté urobíme pre SV. V uvedenom príklade dostaneme vektory $(\frac{1}{\sqrt{17}}, \frac{4}{\sqrt{17}})$, $(\frac{1}{\sqrt{10}}, \frac{3}{\sqrt{10}})$ a $(\frac{4}{\sqrt{41}}, \frac{5}{\sqrt{41}})$. Potom[5] pre $ou1 = 0.997$ a pre $ou2 = 0.7$. Teda pomocou príslušnej normalizácie vstup-

[4]Tomuto výrazu hovoríme **dot produkt**

[5]**Pozor -** po normalizácii musí byť výraz $\sqrt{\sum_{i=1}^{M} x_{norm-i}^2} = 1$!!!

ných vektorov a samotných SV sme dosiahli **správne správanie** sa procedúry zhlukovania. Vyriešením tohoto problému sme však dospeli k problému pre určité typy vektorov. Napr. v prípade, že použijeme normalizáciu, tak nebudeme schopní rozlíšiť vektory, ktoré majú rovnaký smer, ale rôzne veľkosti, lebo my v podstate spočítavame uhol medzi týmito vektormi. Riešením môže byť doplnenie vstupného vektrora a SV o ďalší príznak[6], teda budeme pracovať nie v n-rozmernom priestore ale v (n+1)-rozmernom priestore. Po normalizácii už tieto vektory nebudú mať rovnaký smer. Dokonca v prípadoch, keď tieto smery sú blízke, sa procedúra nechová korektne. Ak chceme predísť týmto problémom, musíme zmeniť prístup k výpočtu hodnoty pre **víťazný** neurón teda vzťah (5.1).

Vzhľadom na to, že vlastne chceme hľadať rozdiel medzi vstupným vektorom a SV, tak môžeme novú hodnotu neurónu vypočítať ako **Euklidovu vzdialenosť** medzi jednotlivými vektormi, teda pre neurón x_i to bude hodnota

$$ou_i \;=\; \sum_{l=1}^{M} (w_l - x_l)^2 \tag{5.11}$$

kde x_l sú hodnoty vstupu a w_l sú hodnoty SV, ktoré smerujú k neurónu "i". Tento výpočet ale mení vzťah (5.2) do tvaru

$$ou_i(t) \;=\; \begin{cases} 1 & ak\ ou_i \;=\; \mathbf{min}\{ou_i, \forall\ i = 1, \ldots, N_c\} \\ 0 & inak \end{cases} \tag{5.12}$$

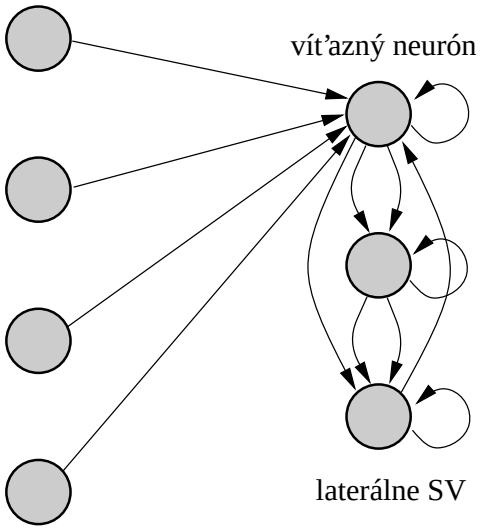
Tento prístup má výhodu voči predošlému v tom, že **nie je** požadovaná normalizácia a odpadávajú problémy s vektormi rovnakých smerov. Ak sa vrátime k predošlému prípadu, tak dostaneme výsledky $ou1 \;=\; 1$ a $ou2 \;=\; \sqrt{10}$, čo znamená, že vyhráva neurón **1**.

Výsledok samotného procesu zhlukovania je **skrytý** v hodnotách jednotlivých SV, ktoré reprezentujú **centrá zhlukov**, ktoré systém vedel nájsť.

[6] napr. súčin prvkov vektora – funkcionálne linky

5.1.1 MAXNET

V práci [9] je konkurenčné učenie typu WTA doplnené tzv. **laterálnymi** prepojeniami medzi neurónmi tej istej vrstvy viď obr. 5.3. Teda



Obr. 5.3: Štruktúra NN typu MAXNET (laterálne SV)

existujú SV aj v rámci jednej vrstvy a ich hodnoty sú definované ako

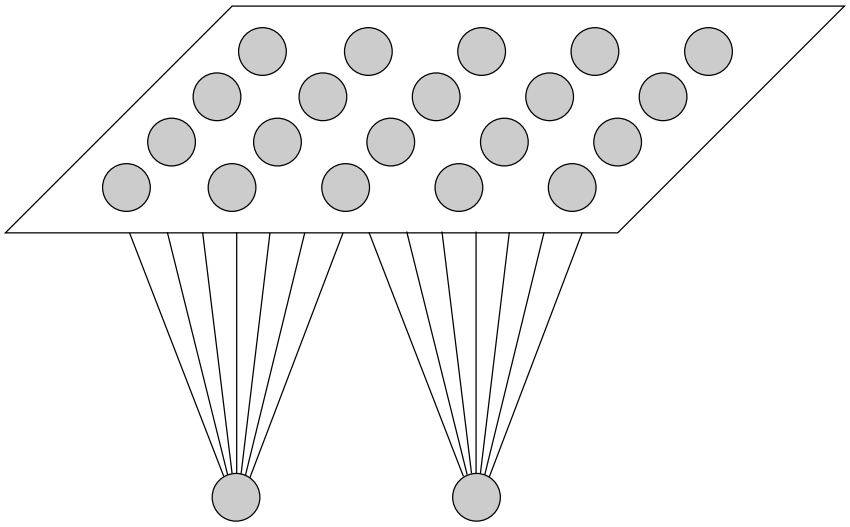
$$w_{ik} = \begin{cases} -\epsilon & ak\ i \neq k \\ 0 & inak \end{cases} \tag{5.13}$$

Toto prepojenie pomôže zvýrazneniu rozdielu medzi víťazným neurónom a ostatnými neurónmi. Vzťah (5.13) je vhodný pre situáciu, keď vypočítavame novú hodnotu podľa (5.1). Ak použijeme na výpočet vzorec (5.11), tak $-\epsilon$ sa musí zmeniť na $+\epsilon$.

5.2 Kohonenove siete

Kohonenove siete predstavujú veľmi dôležité rozšírenie konkurenčného učenia. Biologické systémy majú podobný tvar hlavne vo svojej percepčnej časti - oku. Rozšírenie konkurenčného učenia spočíva v tom,

že sa pripúšťa princíp **viacerých víťazov - multiply WTA**. Ďalej výstup z takejto NN je geometricky usporiadaný do nejakého útvaru napr. vedľa seba, obdlžníka, a teda existuje možnosť určenia **suseda**. Túto vrstvu nazývame **Kohonenovou vrstvou**. Súčasne sa samotné



Obr. 5.4: Typická topologia Kohonenovej NN

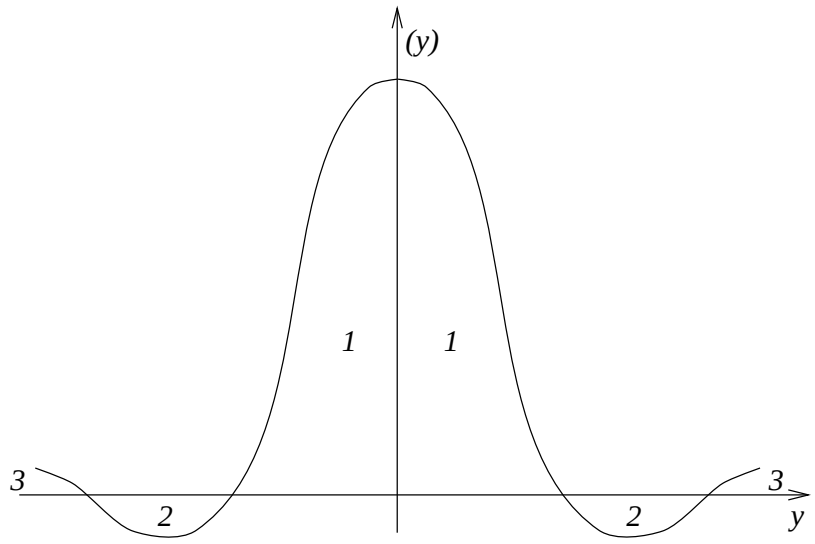
zhlukovanie organizuje takým spôsobom, že susedné neuróny resp. SV, ktoré k nim smerujú, majú podobné hodnoty a tie, ktoré sú ďalej od seba, majú viac rozdielne hodnoty. Príčinou toho celého je zavedenie tzv.**susednej funkcie Λ_{ij}**. Vo väčšine prípadov ide o funkciu v tvare :

$$\Lambda_{ij} \;=\; h(t).e^{\frac{(d_j)}{r_i(t)}^2} \tag{5.14}$$

kde $h(t)$ je adaptačná výška, d_j je vzdialenosť medzi neurónmi v Kohonenovej vrstve a samotné $r(t)$ predstavuje polomer priestorového susedstva v iterácii t. Takáto funkcia môže byť vyjadrená v tvare **mexického klobúka**, ako je uvedené na obr. 5.5.

Potom nastane modifikácia vzorca (5.6) do tvaru :

$$\Delta w_{ij}(t) \;=\; -\gamma\Lambda_{ij}(w_{ij}(t) - x_j^k(t)) \tag{5.15}$$



Obr. 5.5: Možný tvar funkcie susednosti Λ_{ij}

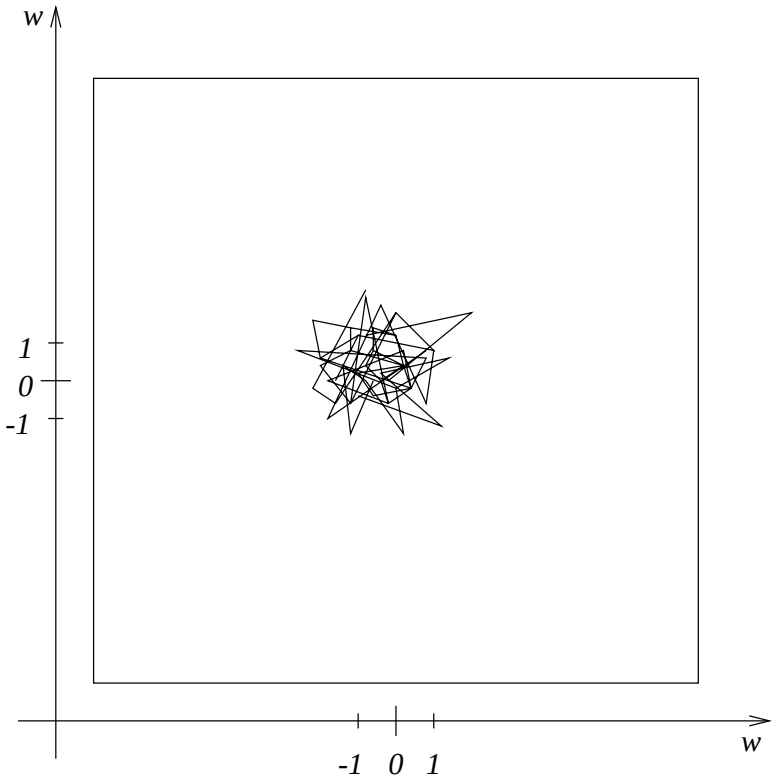
a v konečnom dôsledku bude výpočet novej hodnoty SV mať tvar[7] :

$$w_{ij}(t+1) \;=\; w_{ij}(t) + \gamma\Lambda_{ij}(x_j^k(t) - w_{ij}(t)) \qquad (5.16)$$

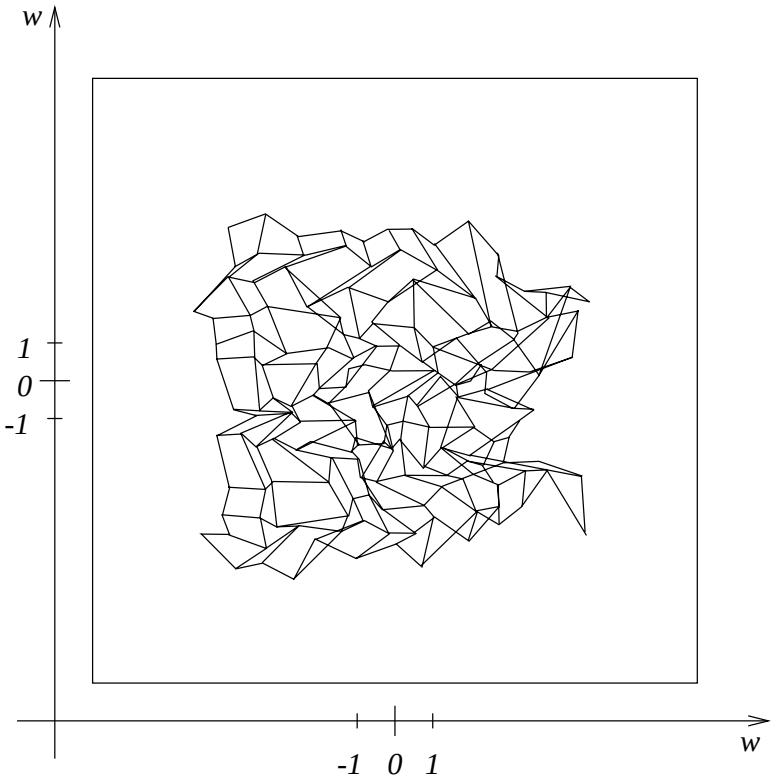
Cez túto funkciu susednosti sú teda spojené jednotlivé neuróny v geometricky usporiadenej výstupnej vrstve. Potom graf, ktorý bude obsahovať jednotlivé rozdelenia váh bude mať v nasledovné možné tvary uvedené na obr. 5.6, 5.7 a 5.8.

Je nutné si uvedomiť, že uvedené grafy zobrazujú situáciu SV len z dvoch vstupov. V prípade, ak je vstupov viac je potrebné pre potreby vizualizácie graficky zvýrazniť dvojice . Zobrazenie v trojrozmernom zobrazení je dosť zriedkavé.

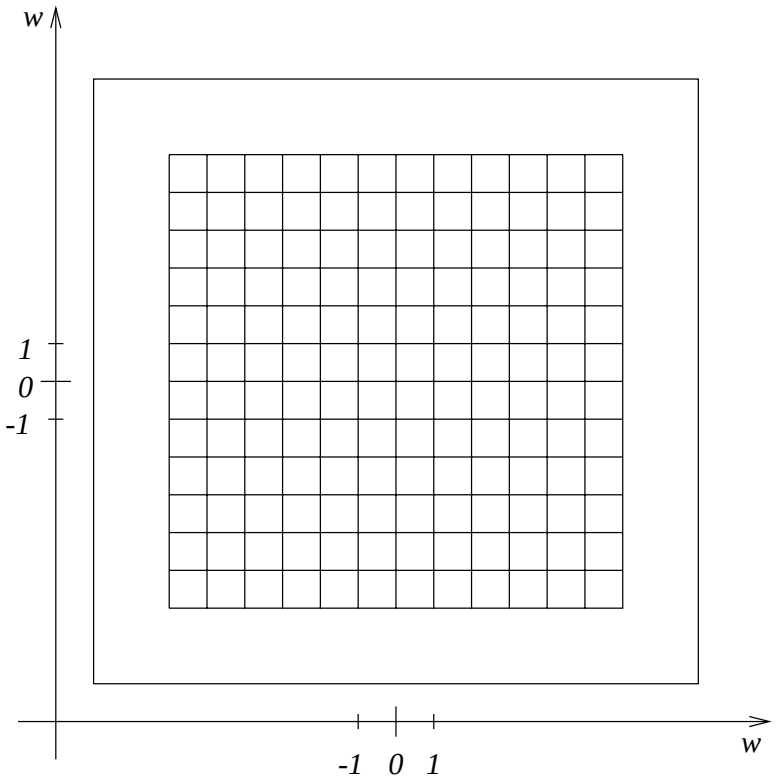
[7] voči vzorcu (5.15) sa vymenili argumenty v zátvorke



Obr. 5.6: Stav SV pri iniacializácii z intervalu $< -1, 1 >$



Obr. 5.7: Stav SV po určitej dobe učenia t



Obr. 5.8: Stav SV na konci učenia,v stave GS, keď vstupné vzorky boli rovnomerne rozdelené

5.3 Metóda hlavných komponentov

Majme dáta z M-rozmerného príznakového priestoru. Tieto dáta majú svoju informačnú hodnotu. Predpokladajme, že **chceme prejsť** z M-rozmerného priestoru do menšieho N-rozmerného priestoru, ale s podmienkou čo **najmenšej informačnej straty**. Takýto prechod je transformáciou príkladového priestoru popísaného pôvodnými príznakmi do príkladového priestoru, ktorého súradnice sú tvorené z **hlavných komponentov**. V klasickej matematike existuje rutinný prístup k hľadaniu hlavných komponentov napr. v [10]. Cieľom tejto časti je ukázať, že pomocou NN je tiež možné túto tranformáciu zvládnuť. Konečná výhoda NN implementácie je vo využití výpočtovej výhodnosti, hlavne pri **paralelných** systémoch. Táto procedúra predstavuje **zhustenie** informácie pri zachovaní čo najväčšej informačnej hodnoty dát. Dôležité je uvedomiť si, že tento prístup predstavuje zhustenie, kde **úplná** rekonštrukcia dát je problematická.

5.3.1 Ojove adaptačné pravidlo zmeny SV

Majme jednoduchú NN s **n** vstupmi a jedným výstupným neurónom (viď obr. 5.9) Nech výstupný neurón je **lineárneho typu**, teda

$$ou(t) \;=\; \sum_{i=1}^{n} w_i(t)x_i(t) \;=\; \mathbf{w}(t)\mathbf{x}(t) \tag{5.17}$$

Predpokladajme **Hebbove synapsie** medzi neurónmi, teda je zrejmé, že adaptačné pravidlo bude mať tvar[8]

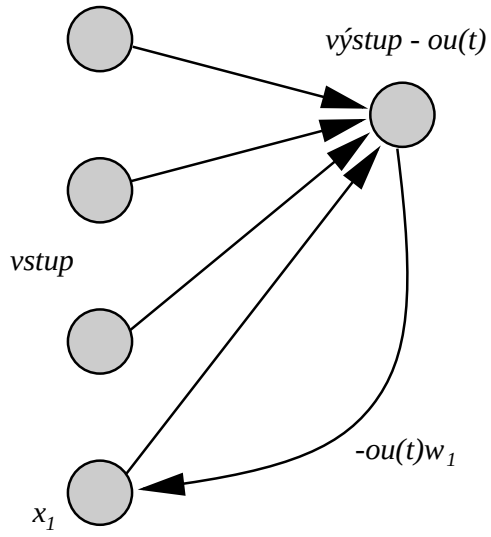
$$\Delta\mathbf{w}(t) \;=\; ou(t)\mathbf{x}(t) \tag{5.18}$$

čo v konečnom dôsledku ovplyvňuje novú hodnotu SV v $(t+1)$ iterácii

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma ou(t)\mathbf{x}(t) \tag{5.19}$$

pre $\forall i = 1, \ldots, n$. Problém vzorca (5.19) spočíva v tom, že pre $t \to \infty$ w_i neúmerne rastie, čo pri reálnych systémoch spôsobuje problémy. Predísť tejto saturácii môžeme určitou formou normalizácie výrazu

[8] vo vektorovom zápise pre prehľadnosť označenie tranponovania vynechávame



Obr. 5.9: Priklad NN pre hľadanie prvého hlavného komponentu

(5.19). Oja navrhol nasledovnú formu (výrazy sú vo vektorovom tvare):

$$\mathbf{w}(t+1) = \frac{\mathbf{w}(t) + \gamma \mathbf{\Delta w}(t)}{L(\gamma)} \tag{5.20}$$

kde $L(\gamma)$ je $|\mathbf{w}(t)+\gamma \mathbf{\Delta w}(t)|$ a po dosadení zo vzorca (5.18) má tvar

$$L(\gamma) = \sqrt{|\mathbf{w}(t)|^2 + 2\gamma ou(t)|\mathbf{w}(t)|\mathbf{x}(t) + \gamma^2 ou(t)^2 |\mathbf{x}(t)|^2} \tag{5.21}$$

Funkciu $L(\gamma)$ rozvinieme do Taylorovho radu a členy γ^2 a vyššie mocniny γ pri predpoklade malého γ zanedbajme.[9] Potom pre druhý člen rozvoja dostaneme (predpokladáme $|\mathbf{w}(t)| = 1$) :

$$2\frac{\partial L}{\partial \gamma}_{\gamma=0} = 2ou(t) \underbrace{\mathbf{w}(t)\mathbf{x}(t)}_{ou(t)} + 0 \tag{5.22}$$

[9] Taylorov rad rozvoj funkcie napr $f(\gamma) = 1 + \gamma \frac{\partial f(\gamma)}{\partial \gamma}_{\gamma=0} + O(\ldots)$, kde $O(\ldots)$ sú ostatné členy rozvoja

čo po úprave znamená, že

$$\frac{\partial L}{\partial \gamma}_{\gamma=0} \;=\; ou^2(t) \tag{5.23}$$

teda samotný Taylorov rozvoj $L(\gamma)$ má konečný tvar pri zanedbaní členov s vyššími mocninami γ

$$L(\gamma) \;=\; 1 \;+\; \gamma ou^2(t) \tag{5.24}$$

ale v podstate my máme v zmysle situácie v (5.20) tvar

$$\frac{1}{1 \;+\; \gamma ou^2(t)} \tag{5.25}$$

ktorý keď rozšírime výrazom $1 \;-\; \gamma ou^2(t)$ dostaneme

$$\frac{1 - \gamma ou^2(t)}{1 \;-\; \gamma^2 ou^4(t)}. \tag{5.26}$$

Pretože γ môže byť zvolené dostatočne malé, $\gamma^2 ou^4(t) \to 0$, teda menovateľ sa blíži k 1 a môžeme písať

$$\frac{1}{1 \;+\; \gamma ou^2(t)} \;\doteq\; 1 \;-\; \gamma ou^2(t) \tag{5.27}$$

ak to dosadíme späť do vzorca (5.20) a znova **zanedbáme** γ^2 dostaneme

$$\mathbf{w}(t+1) \;=\; (\mathbf{w}(t) \;+\; \gamma ou(t)\mathbf{x}(t))(1 \;-\; \gamma ou^2(t)) \tag{5.28}$$

čo v konečnom dôsledku znamená

$$\mathbf{w}(t+1) \;=\; (\mathbf{w}(t) \;+\; \gamma ou(t) \underbrace{(\mathbf{x}(t) \;-\; ou(t)\mathbf{w}(t))}_{\mathbf{x}'(t)} \tag{5.29}$$

kde je zrejmé, že

$$\Delta\mathbf{w}(t) \;=\; \gamma ou(t)(\mathbf{x}(t) \;-\; ou(t)\mathbf{w}(t)) \tag{5.30}$$

teda nová hodnota SV sa vypočíta ako

$$\mathbf{w}(t+1) \;=\; (\mathbf{w}(t) \;+\; \gamma ou(t)\mathbf{x}'(t)) \tag{5.31}$$

kde $\mathbf{x}'(t)$ predstavuje tzv. **efektívny** vstup do výstupného neurónu. Teda doteraz uvedený proces môžeme zhrnúť do nasledovných bodov:

- nárast SV na základe vstupu $\mathbf{x}$(t)
- **pomyselná** spätná vazba $-ou(t)\mathbf{w}(t)$, ktorej úloha je kontrolovať nárast SV a tak stabilizovať činnosť NN. Táto pomyselná spätná väzba sa nazýva tiež **zabúdací faktor**.

Vzorec (5.29) sa tiež nazýva **Ojove adaptačné** pravidlo zmeny SV. Po nájdení GS NN dostaneme v hodnotách **w** vektora **prvý hlavný** komponent. To, že NN s takýmto adaptačným pravidlom určite nájde svoju GS, nájdeme popísané v [5].

Rozšírme teraz náš zámer o hľadanie ďalších hlavných komponentov zhusteného priestoru komponentov. Teda majme jednoduchú NN s M vstupmi a N výstupmi viď obr. 5.10 súčasne platí, že $N < M$ a výstupné neuróny sú lineárneho typu. Z výhodných dôvodov si označme jednotlivé neuróny vo vstupnej vrstve nasledovne vstupné

$$\mathbf{j} = \mathbf{0}, \ldots, \mathbf{M-1}$$

a výstupné

$$\mathbf{i} = \mathbf{0}, \ldots, \mathbf{N-1}$$

. Potom výstup v neurónu "i" vypočítame ako (v skalárnom vyjadrení)

$$ou_i(t) = \sum_{l=0}^{M-1} w_{ij}x_j \tag{5.32}$$

potom dostaneme vo vzorci (5.33) tzv. zovšeobecnený tvar Hebbovho adaptačného pravidla v tvare[10]

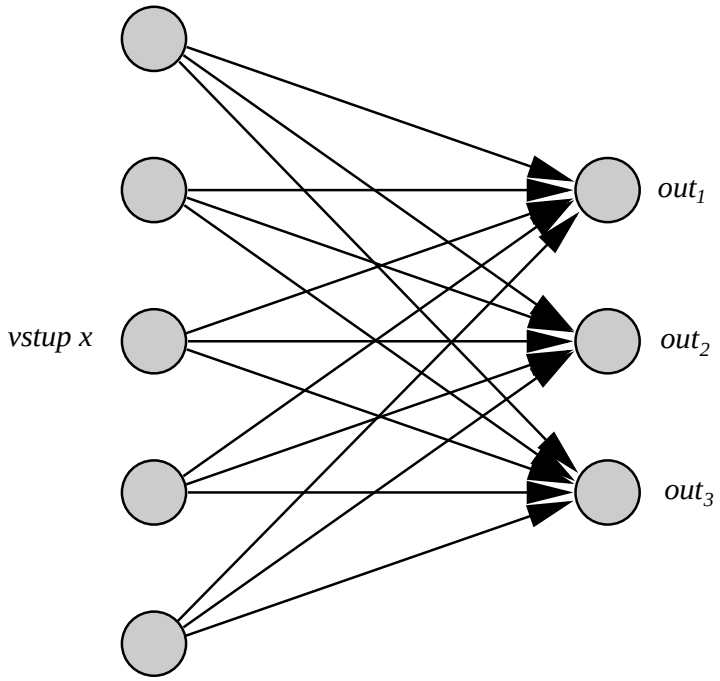
$$\Delta w_{ij}(t) = \gamma ou_i(t)\left(x_j(t) - \sum_{l=0}^{i} w_{lj}(t)ou_l(t)\right) \tag{5.33}$$

Ak do vzorca (5.33) dosadíme $i = 0$, tak dostaneme vzorec pre adaptačné pravidlo, ktoré sme už odvodili vo vzťahu (5.30).

Teraz opäť pre následnú výhodnosť si vzorec (5.33) do tvaru

$$\Delta w_{ij}(t) = \gamma ou_i(t)\left(\underbrace{x_j(t) - \sum_{l=0}^{\mathbf{i-1}} w_{lj}(t)ou_l(t)}_{x'_j(t)} - w_{ij}(t)ou_i(t)\right) \tag{5.34}$$

[10]berte do úvahy označenie neurónov



Obr. 5.10: NN pre vypočet 3 hlavných komponentov

Teraz v podstate dostávame situáciu, keď máme jedno adaptačné pravidlo, kde sa efektívny vstup do jednotlivých výstupných neurónov mení. V rámci jednej učebnej procedúry sa nám podľa (5.33) budú meniť SV rôzne, podľa toho, ku ktorému z výstupných neurónov smerujú (index "i"). Ak chceme nájsť všetkých N hlavných komponentov, potom vlastne hľadáme

1. **prvý hlavný komponent** potom $x_{'}(t)$ zo vzorca (5.34) má tvar (**i = 0**)

$$x_j^{'}(t) = x_j(t) \tag{5.35}$$

2. **druhý hlavný komponent** (**i = 1**)

$$x_j^{'}(t) = x_j(t) - w_{0j}(t) o u_0(t) \tag{5.36}$$

3. **tretí hlavný komponent** (**i = 2**)

$$x_j^{'}(t) = x_j(t) - w_{0j}(t) o u_0(t) - w_{1j}(t) o u_1(t) \qquad (5.37)$$

4. atď

Takýmto spôsobom je možné postupne vypočítať jednotlivé hlavné komponenty dát a realizovať ich zhustenie. Existuje ešte verzia učiaceho algoritmu, ktorá uvažuje o laterálnych prepojeniach. Tak isto aj pre prípad nelineárnych neurónov bol vyvinutý Ojo-om podobný postup.

6

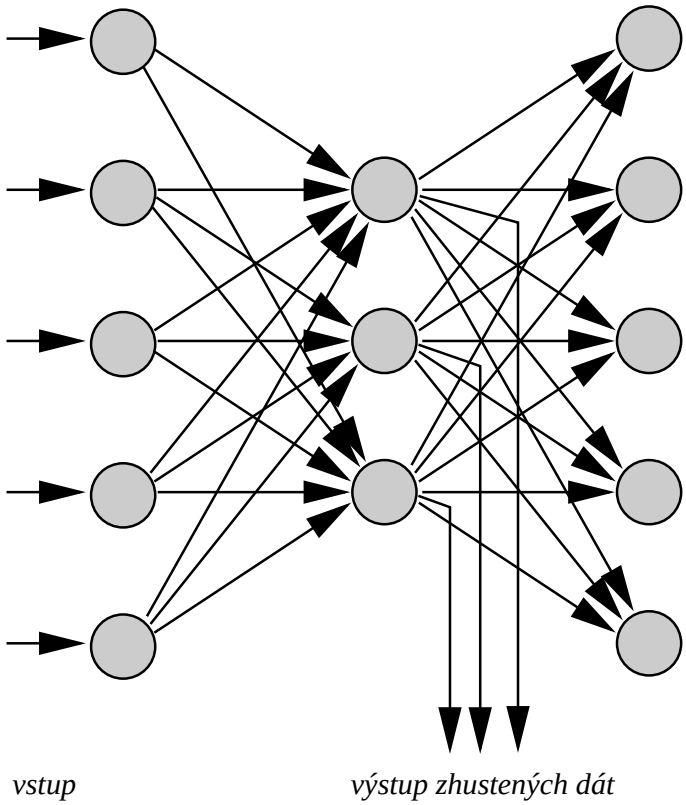
Hybridné metódy učenia na FF NN

Po uvedených prístupoch kontrolovaného a nekontrolovaného učenia na FF NN je potrebné pripomenúť, že existujú aj prístupy, ktoré využívajú obidva tieto učiace mechanizmy. Jedným z nich je využitie konkurenčného učenia a metódy BP v jednom organickom celku.

6.1 Nekontrolované učenie metódy BP

V podstate ide o pseudo-nekontrolované učenie. Spočíva v tom, že FF NN sú predkladané na vstup a výstup tie isté vzorky. Uvažujme o topológii FF NN v tvare M-N-M, kde $N < M$ a väčšinou $N = log_2 M$. Príklad takejto topológie je na obrazku 6.1. Takáto NN sa dá vhodne využiť ma kompresiu dát. Je použitý klasický BP prístup, ale s tým, že na vstup a výstup dodávame tie isté dáta. Zhustené dáta potom môžeme získať na skrytej vrstve o veľkosti N. Tento prístup nám šikovne zabezpečuje aj možnosť rekonštrukcie dát do pôvodnej formy. Teda postup je triviálny :

- naučíme NN podľa BP s tým, že ponúkame na vstup trénovacie data (predstavujú množinu všetkých možnosti), ktoré môžu v dátach nastať a také isté dáta očakávame na výstupe. Snažíme sa dostať do stavu, keď NN vie spoľahlivo realizovať celý proces a SV sa už nemenia, teda GS.



Obr. 6.1: Zhusťovanie dát z M na N

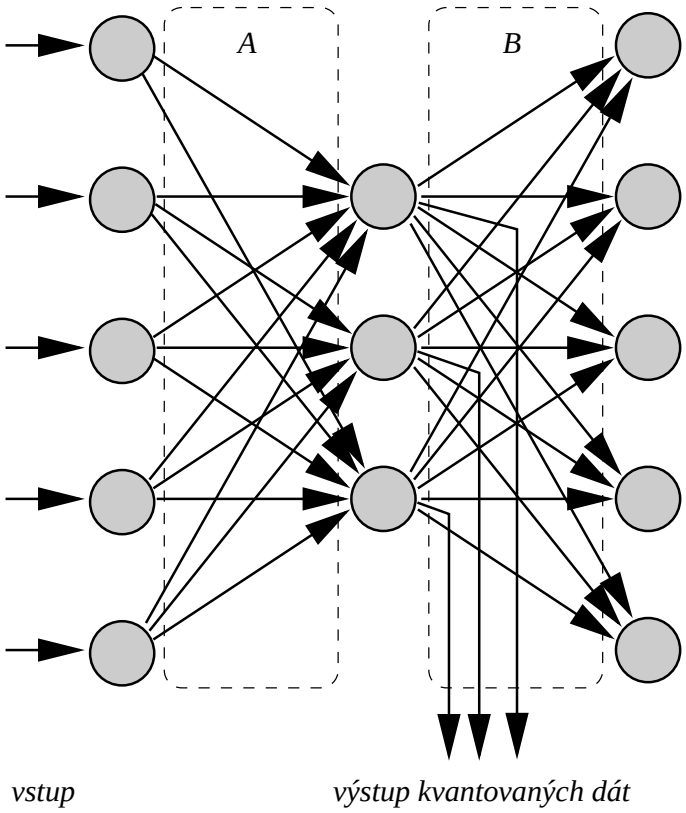
- potom sme v stave, keď môžeme spoľahlivo realizovať kompresiu dát a získať zhustené dáta zo skrytej vrstvy
- tým máme zabezpečenú aj spätnú rekonštrukciu dát.

Tento jednoduchý prístup sa zdá byť veľmi efektívny pre spomínanú kompresiu dát.

6.2 Metóda Counterpropagation

Jednou z ďalších možností hybridného prístupu je kvantovanie vstupu, pričom musí byť splnená podmienka opätovnej rekonštrukcie. Toto

vykonáva práve metóda **Counterpropagation**, ktorá kombinuje vyššie uvedené metódy nasledujúcim spôsobom, kde **prvá časť** pracuje na princípe konkurenčného učenia a **druhá časť** používa metódu BP. (viď obr. 6.2).



Obr. 6.2: Topológia NN pre hybridné učenie metódou Counterpropagation (A - konkurenčné učenie, B - kontrolované učenie)

Samotné učenie môže prebiehať buď sekvenčne alebo paralelne. Konkurenčné učenie prebieha podľa jedného adaptačného pravidla a učiaceho koeficientu γ_1 a druhá časť podľa druhého adaptačného pravidla a druhého koeficientu učenia γ_2. Celkový postup učenia môžeme zhrnúť do nasledovných bodov:

1. inicializácia jednej aj druhej časti FF NN

2. vstup do NN a výpočet hodnoty neurónu v skrytej vrstve podľa vzorca (5.11) a výpočet minimálnej hodnoty stavu jednotlivých neurónov podľa (5.12).

3. neurón, ktorý bol identifikovaný ako najbližší ku vstupu bude nastavený na **1** a ostatné na **0**

4. potom sa vypočíta rozdiel medzi výstupom z výstupnej vrstvy a očakávanou hodnotou na výstupe

$$e_i \;=\; (ev_i \;-\; ou_i) \tag{6.1}$$

5. teraz sa budú realizovať dve adaptačné pravidlá a to

   (a) pre konkurenčné učenie sa zmena SV vypočíta ako

$$\Delta w_{kj}(t) \;=\; \gamma_1(w_{kj}(t) \;-\; x_j(t)) \tag{6.2}$$

   ak "k" je víťazný neurón a sa ostatné SV **sa nemenia**

   (b) pre druhú časť, teda BP adaptačné pravidlo pre zmenu SV druhej časti NN sa vyjadrí podľa vzorca (4.34) pre prípad výstupného neurónu (vzorec (4.38)) s použitím lineárnej aktivačnej funkcie na výstupe NN. Teda

$$\Delta w_{ik}(t) \;=\; -\gamma_2(ev_i \;-\; ou_i(t))x_k \tag{6.3}$$

   Vzhľadom na to, že v skrytej vrstve majú všetky neuróny nulovú hodnotu až na víťazný, ktorý je nastavený na **1**, potom $x_k = 1$ a ostatné sú 0. Súčasne môžeme pre $ou_i(t)$ vzhľadom na lineárnu aktivačnú funkciu napísať

$$ou_i(t) \;=\; \sum_{j=1}^{N_z} w_{ij}x_j \;=\; w_{ik} \tag{6.4}$$

   Potom môžeme vzorec (6.3) upraviť do tvaru

$$\Delta w_{ik}(t) \;=\; \gamma_2(w_{ik}(t) \;-\; ev_i(t)) \tag{6.5}$$

6. návrat do bodu 2 a opakovanie procedúry až kým chyba nebude dostatočne malá a nastane konvergencia FF NN

Význam tohto prístupu je v **kvantovaní** vstupov zabezpečením maximálne možnej rekonštrukcie. Samotné kvantovanie zabezpečuje prvá časť siete, druhá časť zabezpečuje podmienky maximálnej možnej rekonštrukcie. Výsledky tohoto procesu nájdeme v hodnotách SV prvej časti siete, kde sa nachádzajú hodnoty centier zhlukov po kvantovacom procese vstupu.

6.3 Repetitórium č. 3

- **4. tutoriál**

1. Aká je logika nekontrolovaného učenia? Ake typy úloh sa dajú riešiť na dopredných sieťach s takýmto typom učenia?
2. Čím sa vyznačuje topológia MAXNET?
3. Aký je princíp učenia **víťaz berie všetko**?
4. Prečo je nutné normalizovať vstupné vektory pre konkurenčné učenie na dopredných NN? Čo vlastne vypočítame pri skalárnom súčine normalizovaných vektorov?
5. Čo je výsledkom celého procesu konkurenčného učenia na dopredných sieťach?
6. Čím sa Kohonenove siete líšia od základného konkurenčného učenia?
7. Vysvetlite graf váh Kohonenovej NN!
8. Aká je logika zhustenia dát pomocou nekontrolovaného BP učenia na doprednej sieti ?
9. Aký význam má metóda hlavných komponentov a k čomu slúži?
10. Odvoďte Ojove pravidlo!
11. Aký je rozdiel medzi nekontrolovaným BP a Counterpropagation sieťami?

Poznámky

Poznámky

Literatúra

[1] Barto A.G.,Anandan P.: Pattern-recognizing stochastic learning automata, IEEE Transaction on Systems, Man and Cybernetics 15 (1985),360-375.

[2] Carpenter G.A.,Grossberg A.: A Massively Parallel Architecture for Self-Organizing Neural Pattern Recognition Machine, Computer Vision, Graphics and Image Processing, 1987,vol.37, pp.54-115.

[3] Funahashi K.: On the Aproximate Realization of Continuous Mapping by Neural Networks, neural Networks, Vol.2, pp. 183-192, 1989.

[4] Halgamuge S.K. a kol. : Fast Perceptron Learning by Fuzzy Controlled Dynamic Adaptation of Network Parameters, in Fyzzy Systems in Computer Sciene, Editors R.Kruse at all, University of Amsterdam, 1994.

[5] Haykin S.: Neural Network - A Comprehensive Foundation Macmillian Publishing, 1994.

[6] Hertz J.,Krogh A.,Palmer R.: Introduction to the Theory of Neural Computation Addison Wesley publishing , 1991.

[7] Jacobs R.A.:Increased rate of convergence through learning rate adaptation, Neural Networks 1, 1988, pp. 295-307.

[8] Kohonen T.: Self-Organization and Associative Memory , Springer-Verlag Berlin, 1984.

[9] Lippmann R.P.: Review of Neural Networks for Speech Recognition,Neural Computation , Volume 1-N1, 1989, pp. 1-38.

[10] Lukasová A.,Šarmanová J.: Metody shlukové analýzy, SNTL, Praha, 1985.

[11] Minai A.A. Williams R.J.: Backpropagation heuristics: A study of the extended delta-bar-delta algorith., International Joint Conference on NN, Vol 1, pp. 595-600, 1990 San Diego, CA - USA.

[12] Novák V.: Fuzzy množiny a jejich aplikace, SNTL, Praha, 1990.

[13] Pao Y.H.: Adaptive Pattern Recognition and Neural Network, Addison-Wesley, 1989.

[14] Pearlmutter B.: Dynamic Recurrent Neural Networks , CMU-CS-90-196, 1990.

[15] Pineda,F.J.: Recurent Backpropagation and Dynamical Approach to Adaptive Neural Computation, Neural Computation, N.2, 1989.

[16] Rosenblatt F.: Principles of Neurodynamics Perceptron and the Theory of Brain Mechanism, Spartan Press, Washington, 1962.

[17] Sincak, P. , Andras, D. , Reiss R.: Comparative Study of Fuzzy and Non-fuzzy Approach to Speeding–up Backpropagation Learning Procedure, Conference on Fuzzy Logic Budapest, 1996.

[18] Sobajic, Pao Y.H.: Artificial Neural Net Based Security Assesment for Eletrical Power Systems, IEEE Transactions on Power Systems, 1989, Vol3.

Register

adaline, 45
adaptácia NN, 11
aplikácie NN
  typy riešených úloh, 8
Ashby, 6

Backpropagation, 7
  BP–momentum, 58

chybová funkcia, 41, 51

funkcionálne linky, 66
fázy činnosti NN
  učenie, 11
  život, 12
    algorithm–less systems, 12

Grossberg, 6

Hebb, 5
Hopfield, 7

inicializácia NN, 37, 71

konvergencia, 37

McCulloch & Pitts, 5
metóda hlavných komponent, 89
Minsky, 6

neurón, 5, 13
  aktivačná funckia
    signum, 16
  aktivačná funkcia, 13
    druhy aktivačných funkcií, 15
    lineárna, 16
    po častiach lineárna, 16
    sigmoidálna, 16
  dendrit, 13
  postsynaptický neurón, 14
  prah neurónu, 13
  predsynaptický neurón, 14
  výstupná funkcia, 14
Neurónová sieť, 3
  dopredná, 19
  implementácia, 4
  Kohonenove NN, 83
  rekurentná, 19
  simulácia, 4
  teória, 4
  time delay siete, 56
  topológia, 18
    návrh topológie NN, 69
    skrytá vrstva, 18
    vstupná vrstva, 18
    výstupná vrstva, 19

Perceptrón, 6, 20
  klasifikácia
    dichotomická, 21
  terminológia, 27
  topológia NN, 22

učenie NN, 23
veta o konvergencii, 25
príklad, 12
príkladový priestor, 12
príznak, 12

Rosenblat, 6

stabilita, 37
vyšetrovanie stability NN, 38
význam stability, 39
synapsia, 14
hrany, 18
synaptické váhy, 14, 17
excitačné, 17
inhibičné, 17
synaptické váhy
laterálne, 83

umelá inteligencia, 1
Univerzálna aproximačná teoréma, 70
učenie
Backpropagation, 52
Delta pravidlo, 51
chybový signál, 52
hybridné metódy, 95
Backpropagation, 95
Counterpropagation, 96
Least-Mean-Square(LMS), 44
metóda najstrmšieho zostupu, 41
nekontrolované, 77
konkurenčné, 78
kvantovanie, 77
metóda hlavných komponentov, 77
Ojove pravidlo, 89
zhluková analýza, 77
Steepest Descent, 43
urýchlenie učenia, 58
Delta–bar–Delta pravidlo, 59
fuzzy logika, 64
učiaci pomer, 44
veľkosť trénovacej vzorky, 72
učenie NN, 11
kontrolované, 31
error correction learning, 32
paradigmy, 32
reinforcement, 33
stochastické, 33
nekontrolované, 31
paradigmy, 34
nekotrolované
Hebbovo, 34
konkurenčné, 34

vlastnosti NN
učenie (learning), 2
reprezentácia vedomostí (knowledge representation), 2
uvažovanie (reasoning), 2
vzorka
reprezentatívna vzorka, 13
testovacia vzorka, 13
trénovacia vzorka, 13

Wiener, 5
Wienerov filter, 41
autokorekačná funkcia, 43
kroskorelačná funkcia, 43

XOR problém, 27

šírenie signálu v NN, 19

asynchróne, 20
blok-sekvenčné, 20
sekvenčné, 20
synchrónne, 20