

Neurónové siete - inžiniersky prístup

verzia 4.0

Peter Sinčák

Jún 2023

Zoznam obrázkov

1.1	Hlavní predstavitelia vedeckého seminára v roku 1956 v Dartmounthe USA, zdroj []	11
1.2	Kognitívna príepasť kognitívnych procesov a dva prístupy v umelej inteligencii	13
1.3	Hlavné komponenty všeobecného systému UI	20
1.4	Princíp integrácie a synergie znalostí experta a existencie dát s cieľom vytvorenia virtuálneho experta	22
2.1	ilustračný obrázok 2 + 1 rozmerného chybového priestoru a výpočtu gradientu na chybovom povrchu s cieľom hľadania globálneho minima chyby L	26
2.2	Štruktúra neurónu	28
2.3	Označenie neurónov	29
2.4	Príklady linearnej a prahovej aktivačnej funkcie neurónu	30
2.5	Sigmoidálna a po časťach lineárna aktivačné funkcie neurónu	31
2.6	Štruktúra doprednej NN	33
2.7	Štruktúra RC NN	34
3.1	Obrázok skutočného ľudského mozgu	35
3.2	Purkyňova bunka v mozočku. Príklad komplikovanej štruktúry niektorých buniek v mozgu. Prevzaté z [?].	39
3.3	Kľúčová publikácia, ktorá vytvorila základ kybernetiky a umelej inteligencie	45
3.4	Základné oblasti ľudského mozgu	46
3.5	Základná štruktúra biologického neurónu	48
3.6	Dva typy unipolárnych neurónov jeden bipolárny	49
4.1	Štruktúra doprednej NN	59
5.1	Lineárna separovateľnosť tried v príkladovom priestore	64

5.2	Topológia perceptrónu navrhnutého Rosenblattom	65
5.3	Grafické znazornenie funkcie XOR	69
5.4	Dve možné riešenia funkcie XOR pomocou skrytej vrtsvy	70
5.5	Schéma Wienerovho filtra	71
5.6	Chybový priestor pri učení neurónových sietí	73
5.7	Schéma siete Adaline	75
6.1	Zobrazenie toku chybového signálu z výstupu pre i -ty neurón	77
6.2	Zobrazenie toku chybového signálu z výstupu pre īty neurón	81
6.3	NN s vstupom časového signálu	83
6.4	ilustračný obrázok 2 + 1 rozmerného chybového priestoru a výpočtu gradientu na chybovom povrchu s cieľom hľadania globálneho minima chyby L	84
6.5	Príklad NN vyššieho rádu	92
6.6	XOR problém riešený pomocou funkcionálnej linky	93
6.7	Príklad topológie RBF sietí	94
6.8	Príklad topológie RBF sietí	95
7.1	Zmena váh, ktoré smerujú k vŕťaznému neurónu	102
7.2	Príklad jednoduchého zhľukovania do 2 zhľukov v dvojrozmernom príznakovom priestore	105
7.3	Štruktúra NN typu MAXNET (laterálne SV)	107
7.4	Typická topológia Kohonenovej NN	108
7.5	Možný tvar funkcie susednosti Λ_{ij}	108
7.6	Stav SV po inicializácii	109
7.7	Priklad NN pre hľadanie prvého hlavného komponentu	111
7.8	Ojova sieť pre výpočet 3 hlavných komponentov	113
8.1	Zhubšťovanie dát z M na N	116
8.2	Topológia NN pre hybridné učenie metódou Counterpropagation (A - konkurenčné učenie, B - kontrolované učenie)	117
9.1	Jednoduchá Hopfieldova sieť so 6 neurónmi, N=6	121
9.2	Návrh neurónovej siete pre TSP pri $n = 4$. Kvôli prehľadnosti nie sú zakreslené prepojenia neurónov. Riešenie je zobrazené pomocou čiar. Riešením je postupnosť miest 4-2-1-3, prípadne jej cyklická permutácia (permutácia znamená obmieňanie poradia cesty v danej množine 4 miest v danom riešení)	123
9.3	Jednoduchá štruktúra BS s obojstrannými prepojeniami	128
9.4	Jednoduchá štruktúra RBM s obojstrannými prepojeniami	130
9.5	Jednoduchá štruktúra RBM-stack a porovnanie s MLP	131

9.6 Ressistance-Capacity model neurónu	133
9.7 Topológia RC NN	134
9.8 Architektúra neurónovej siete ART1	142
9.9 Proces učenia v neurónovej sieti ART1	143
10.1 Základná bloková schéma modulárnej neurónovej siete pozostávajúcej z K expertných modulov a jedného bránového modulu.	151
11.1 Základná schéma rozpoznávacieho procesu	165
11.2 Porovnanie plytkých a hlbokých neurónových sietí	166
12.1 Príklad typickej topológie konvolučnej neurónovej siete	167
12.2 Ilustračný obrázok pre operáciu pooling na digitálnom obrazze	171
12.3 Ilustračný obrázok operácie unpooling na digitálnom obrazze	172
12.4 Ilustračný obrázok operácie padding na digitálnom obrazze	173
12.5 Ilustračný obrázok rôznych typov operácie padding na digitálnom obrazze	174
12.6 Ilustračný obrázok rôznych typov stridingu na digitálnom obrazze	175
12.7 Ilustračný obrázok konvolučnej operácie na digitálnom obrazze	176
12.8 Príklady vstupov do CNNSIMPLE neurónovej siete	177
12.9 Príklady vstupu čísla 8 ako matice čísel 28x28x1 do CNNSIMPLE neurónovej siete	178
12.10výstup z CNNSIMPLE (výstup z klasifikačnej časti CNN)	178
12.11Príklad jednoduchej konvolučnej neurónovej siete	179
12.12Vstupná časť CNNSIMPLE	180
12.13Druhá časť CNNSIMPLE	181
12.14Tretia časť CNNSIMPLE	182
12.15Výstupná časť CNNSIMPLE	184
12.16ilustračný obrázok 2 + 1 rozmerného chybového priestoru a výpočtu gradientu na chybovom povrchu s cieľom hľadania globálneho minima chyby L	188
12.17pripomenutie konvolučnej neurónovej siete	190
12.18prvá časť spätného šírenia chyby	191
12.19prvá časť spätného šírenia chyby	195
12.20pripomenutie konvolučnej neurónovej siete	197
12.21Prepočet nových hodnôt pre f, S2 a C2 pri spätnom šírení chyby	200
12.22druhá časť spätného šírenia chyby	201
12.23tretia časť spätného šírenia chyby	207
12.24pripomenutie konvolučnej neurónovej siete	209
12.25posledná časť spätného šírenia chyby	211

Obsah

1 Umelá inteligencia a jej definícia	11
1.1 Výpočtová inteligencia	19
1.2 Inteligentné technológie	20
2 Neurónové siete	25
2.1 Základné pojmy a princípy činnosti	25
2.2 Základné pojmy teórie učenia	26
2.2.1 Základné prvky NN	28
2.3 Poznámky k jednotlivým časťam neurónu	29
2.4 Synaptické spojenia a váhy	32
2.4.1 Topológia NN a spôsoby šírenia signálu	32
3 Mozog - procesor biologického systému	35
3.1 Biologicky motivované technické systémy	36
3.2 Modelovanie biologických neurosystémov	38
3.2.1 Použitie neurónových sietí v modelovaní	41
3.3 Inžinierske aplikácie	42
3.3.1 Prístupy a metódy	43
3.4 Budeme sa potrebovať učiť ?	44
4 Učenie neurónových sietí a ich princípy	51
4.1 Paradigmy kontrolovaného učenia	52
4.2 Paradigmy nekontrolovaného učenia	54
4.3 Globálna stabilita a konvergencia NN	56
4.3.1 Globálna stabilita NN	56
4.3.2 Konvergencia NN	58
4.3.3 Dôležité poznámky k návrhu FF NN	59
4.3.4 Návrh topológie NN	59
4.3.5 Problém inicializácie NN	61

4.3.6	Problém stanovenia veľkosti trénovacej vzorky	62
5	Kontrolované učenie na FF NN	63
5.1	Perceptrón	63
5.1.1	Algoritmus učenia perceptróna	65
5.1.2	XOR-problém, skrytá vrstva NN	69
5.1.3	Wienerov filter	70
5.1.4	Metóda najstrmšieho zostupu	72
5.1.5	Metóda najmenšej kvadratickej chyby	74
5.1.6	Adaline	75
6	Delta pravidlo	77
6.1	Metóda spätného šírenia chyby	79
6.1.1	Time-delay na FF NN	82
6.1.2	Spôsoby urýchlenia konvergencie BP	84
6.1.3	BP-momentum	84
6.1.4	Adaptívne parametre učenia NN	85
6.1.5	Delta-bar-Delta pravidlo	85
6.2	Funkcionálne linky v NN	92
6.3	Metóda Radial Basis function (RBF)	93
6.4	Kaskádna korelácia (CC)	97
6.4.1	Matematické základy	97
6.4.2	Algoritmus štandardnej CC	99
6.4.3	Algoritmus CC s pruningom (PCC)	99
7	Nekontrolované učenie na FF NN	101
7.1	Konkurenčné učenie	102
7.1.1	MAXNET	106
7.2	Kohonenove siete	107
7.3	Metóda hlavných komponentov	110
7.3.1	Ojove adaptačné pravidlo zmeny SV	110
8	Hybridné metódy učenia na FF NN	115
8.1	Nekontrolované učenie metódou BP	115
8.2	Metóda Counterpropagation	117
9	Rekurentné NN	119
9.1	Kontrolované učenie na RC NN	120
9.1.1	Hopfieldove siete	120
9.1.2	Problém obchodného cestujúceho (TSP)	122

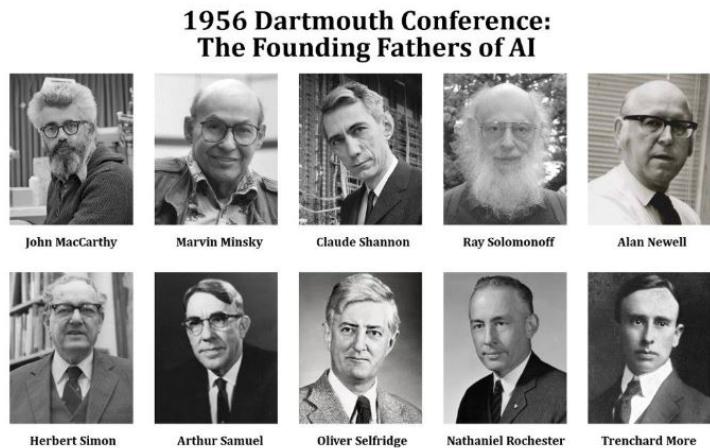
9.1.3	Boltzmanov stroj a jeho varianty	128
9.1.4	Ohraničené Boltzmanove stroje ako základ riešenia hľbo- kého učenia	130
9.1.5	Metóda spätného šírenia chyby (BP) na RC NN . . .	132
9.2	Nekontrolované učenie na RC NN	139
9.2.1	Teoretický popis metód Adaptive Resonance Theory (ART)	139
9.2.2	Siete triedy ART pre spracovanie binárnych dát . .	140
9.2.3	Štruktúra a dynamika sietí ART1	141
9.2.4	Teória sietí ART1	145
10	Modulárne neurónové siete	149
10.1	Štruktúra modulárnej siete	150
10.1.1	Pravdepodobnosná funkcia	152
10.2	Učiaci algoritmus pre regresiu	154
10.2.1	Úprava váh expertných modulov	155
10.2.2	Úprava váh bránového modulu	156
10.3	Učaci algoritmus pre klasifikáciu	158
10.4	EM algoritmus	161
11	Hlboké učenie neurónových sietí	165
12	Konvolučné neurónové siete	167
12.1	Základné pojmy	167
12.1.1	Pojmy pri CNN	168
12.2	Jednoduchá CNN na spracovanie obrazu	177
12.2.1	Topológia, parametre a dopredný prechod cez CNN- SIMPLE	178
12.2.2	Parametre CNNSIMPLE pre učenie	187
12.2.3	Metóda spätného šírenia chyby v konvolučnej sieti .	189
12.2.4	Učenie parametrov výstupnej časti siete	190
12.2.5	Inverzný výpočet zmien príznakových f, S2 a C2 pri spätnom šírení chyby (upsampling)	197
12.2.6	Učenie (adaptácia) parametrov druhej konvolučnej vrstvy filtra $k_{p,q}^2$ pri spätnom šírení z C_q^2	201
12.2.7	Výpočet adaptačného pravidla pre prah Δb_q^2 v kon- volučnej vrstve C_2	205
12.2.8	Prepočet upscalingu v rámci spätného šírenia chyby .	207
12.2.9	Učenie parametrov filtra $k_{1,p}^1$ z prvej konvolučnej vrstvy C_p^1	209

12.2.10 Učenie parametrov \mathbf{b}_p^1 prvej konvolučnej vrstvy	211
13 Otázky k tématam	213
13.1 Repetitórium č. 1	213
13.2 Repetitórium č. 2	215
13.3 Repetitórium č. 3	216
13.4 Repetitórium č. 4	217
13.5 Repetitórium č. 5	218

Kapitola 1

Umelá inteligencia a jej definícia

Problém definícii **Umelej inteligencie** (ďalej UI) je spojený s vysvetlením pojmu **inteligencia**. Samotný pojem “**inteligencia**” pochádza zo spojenia latinských slov “**Inter**” a “**lego**”, ktorý sa pozdejšie vyvinul do pojmu “**Intelligo**” čo v preklade má niekoľko významov napr. **pozrieť sa do vnútra, pochopiť a pod.**¹



Obr. 1.1: Hlavní predstaviteľia vedeckého seminára v roku 1956 v Dartmouththe USA, zdroj []

¹Pozri slovník na stránke <http://www.dictionary.com/others/>

Pojem "Umelá Inteligencia" sa stanovil v roku 1956 na vedeckom seminári v Darthmounte v USA. Na obrázku 1.1 sú protagonisti seminára, ktorý položil základy umelej inteligencii a navždy sa zapísal ako základný kameň nového vedného odboru.

Tento zaujímavý význam slova “**inteligencie**” vyjadruje metafyzický a nie jednoduchý význam tohto pojmu.

Definícia takéhoto pojmu je dosť obsiahla a nie je možné jej popis stručne popísť do niekoľkých definičných konštatovaní. Definícia je predmetom záujmu filozofov a psychológov. Z určitého aspektu je možné o inteligencii povedať napr. nasledovné:

- inteligencia je vlastnosť učiť sa zo skúseností,
- inteligencia je vlastnosť riešiť problémy za pomoci získaných skúseností,
- inteligencia je vlastnosť adaptovať sa v neznámom prostredí.

Z uvedeného je zrejmé, že inteligencia je spojená so znalosťami. Znalosti boli a sú tvorené výskumom a vedami. Už Decart [?] sa pokúsil o určitú systematizáciu vied. V podstate všetky vedy rozdelil do 2 základných skupín a to:

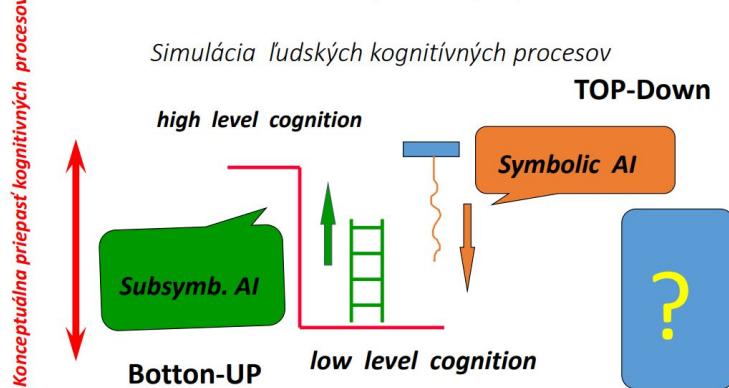
- vedy humanitné,
- vedy exaktné a prírodovedné.

Základným metodologicko-filozofickým rozdielom týchto dvoch skupín vied je úroveň práce s abstrakciou. Vo všeobecnosti môžeme povedať, že pre humanitné vedy (označované niekedy aj α vedy) je práca s abstrakciou veľmi veľká a pri prírodovedných vedách je precíznosť a používanie matematických prostriedkov (označované niekedy ako β vedy). Teda pri α vedách sa vo väčsine prípadov využívajú symboly a pri β vedách sa využívajú numerické hodnoty. Je treba podotknúť, že toto rozdelenie nie je možné robiť veľmi rigorózne ale aj napriek tomu môžeme z filozofického pohľadu hovoriť o α a β logike. Dokonca filozofi hovoria o α a β osobách.

Tieto filozofické úvahy majú veľa spoločné s umelou inteligenciou. Pojem umelá inteligencia nie je práve tým najčastnejším vyjadrením a v literatúre nájdeme aj pojem tzv. “**pseudo-inteligencia**”, čo objektívnejšie vystihuje ciele a ambície prostriedkov UI. Pri UI ide o napodobňovanie **kognitívnych**

procesov, ktorých paleta je veľmi široká včítane procesov logického uvažovania a až po základné percepčné procesy. Kognitívne procesy sa vo filozofii prirovnávajú k tzv. konceptuálnej prieplasti (viď. obrázok 1.2).

Aké sú základné dve paradigmá v UI ???



Obr. 1.2: Kognitívna prieplasť kognitívnych procesov a dva prístupy v umelej inteligencii

Na vrchole prieplasti je abstraktná logika a na dne prieplasti sú základné percepčné procesy. Jednotlivé kognitívne procesy sa nachádzajú v rôznych výškach tejto prieplasti. Modelovanie týchto procesov môžeme symbolicky chápať ako dostup k určitej výške v prieplasti. Tento dostup však môže byť z dvoch základných smerov a to **z hora alebo z dola**. Na základe týchto úvah existujú dve základné paradigmá modelovania kognitívnych procesov a to **symbolická a subsymbolická paradigma UI**. Pod paradigmou budeme rozumieť stratégiu dosiahnutia cieľa teda stratégiu **dostupu k bodu** v konceptuálnej prieplasti.

1. **Symbolická paradigma UI:** táto koncepcia je založená výhradne na manipulácii so symbolmi. Vedci uznávajúci iba symbolickú paradigmu UI sa snažia na "lane" zakotvenom na vrchole konceptuálnej prieplasti sa dostať čo najnižšie, resp. "lano" natiahnuť do najväčšej hĺbky. Na vrchole prieplasti sú prostriedky formálnej logiky, ktorými sa pokúšajú namodelovať všetky kognitívne procesy.
2. **Subsymbolická paradigma UI:** táto koncepcia je založená na odlišnom prístupe. Vedci uznávajúci iba subsymbolickú paradigmu UI sa

snažia na "rebríku" položenom na dne konceptuálnej priepastne dostať čo najvyššie a pokúsiť sa modelovať kognitívne procesy pomocou prostriedkov, ktorými sa modelujú základné percepčné procesy. Pri týchto prostriedkoch sa využívajú prostriedky formálnej logiky len minimálne. Pojem "Sub" sa používa preto lebo vyjadruje nižšiu mieru abstrakcie ako samotná "Symbolická"paradigma. Podstatnú časť týchto prostriedkov tvorí tzv. **výpočtová inteligencia (Computational Intelligence)**.

3. **Komplexná paradigma UI:** táto koncepcia je založená na hybridizácii predošlých dvoch prístupov. Filozoficky je založená na princípe komplexov prostriedkov UI a teda využíva raz "lanoä v inom prípade "rebríkä stale so zreteľom na dosiahnutie výsledného cieľa. Teda ak je zrejmé, na riešenie problému, je vhodné použiť symbolické prostriedky UI, tak sa použijú alebo v inom prípade ak sú vhodnejšie subsymbolické prostriedky UI tak sa vyberú. **Cieľom je efektívne dosiahnuť definovaný cieľ.**

Existuje niekoľko definícií umelej inteligencie napr.:

- **Richard E. Bellman (1920-1984)**



"Umelá inteligencia je automatizácia činností ktoré sú spojené s ľudským myslením, rozhodovaním, riešením problémov, učením a pod "

- John Haugeland (1945-2010)



“Umelá inteligencia je snaha vytvoriť počítače so schopnosťou myslieť – stroje s myšľou v plnom a doslovnom slova zmysle ”

- Ray Kurzweil (1948)



“ Je schopnosť vytvoriť stroje, ktoré riešia problémy, ktoré by človek urobil iba inteligenciou”

- Elaine Rich, 1950



“Umelá inteligencia je oblasť tvorby počítačov, ktoré vyriešili problémy, ktoré teraz človek vyrieší lepšie ako počítače”

- Marvin Minsky (1927-2014)



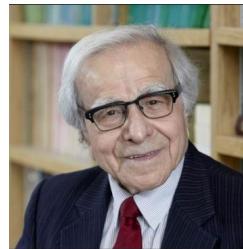
“Umelá inteligencia je veda o vytváraní strojov a systémov, ktoré budú pri riešení určitej úlohy používať taký postup, ktorý - ak by ho robil človek - by sme ho považovali za prejav inteligencie”

- Zdenek Kotek (1930-2004)



“Umelá inteligencia je vlastnosť človekom vytvorených systémov vyznačujúcich sa schopnosťou rozpoznávať predmety, javy a situácie, analyzovať vzťahy medzi nimi a tak vytvárať vnútorné modely sveta, v ktorých tieto systémy existujú a na tomto základe potom realizuje rozhodnutia a súčasne za pomoci svojich schopností predvída dôsledky týchto rozhodnutí a objavuje nové zákonitosti medzi rôznymi modelmi sveta”

- Simon Haykin, 1931



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

- David Lynton Poole, 1958



“Umelá inteligencia je štúdium návrhu inteligentných agentov”

Z predchádzajúcich definícií sú jasné nasledovné fakty, ktoré je vhodné si uvedomiť:

- “Umelá inteligencia” je vedná oblasť (nie je systém alebo nejaký prostriedok)
- Umelá inteligencia ako vedná oblasť sa zaoberá prostriedkami a metódami, ktoré riešia úlohy, ktoré by človek riešil len za pomoci svojich vedomostí.
- Ak ľubovoľný nebiologický systém využíva metódy UI, nazývame tento systém “inteligentný systém”.

Zaujímavé je z pohľadu pochopenia významu UI pre vývoj ľudstva jej prepojenie s filozofiou. Existujú dva základné prístupy z filozofického hľadiska k UI. Tieto dve základné pohľady definujú tzv:

- silnú UI (tzv. strong AI),
- slabú UI (tzv. weak AI).

Stručný prehľad o histórii UI je možné nájsť v práci [?].

1.1 Výpočtová inteligencia

Pojem "výpočtová inteligencia" zaviedol Prof. James Bezdek a tento termín sa natoľko preferoval, že v roku 1994 bola IEEE Svetový Kongres o výpočtovej inteligencii na Floride v USA. Počas tohto kongresu sa konali 3 svetové konferencie (Medzinárodná konferencia o neurónových sieťach, Medzinárodná konferencia o Fuzzy Systémoch a Medzinárodná konferencia a evolučných výpočtoch) a mnohé spoločné plenárne stretnutia.

Prof. Bezdek analyzoval pojem inteligencia a definoval nasledovné pojmy:

- výpočtová inteligencia (VI),
- umelá inteligencia (UI),
- biologická inteligencia (BI).

Súčasne s tými pojмami zavádza pojmy:

- biologické rozpoznávanie (BR),
- umelé rozpoznávanie (UR),
- výpočtové rozpoznávanie (BR).

Cieľom zavedenia pojmu **výpočtová inteligencie** boli nasledovné ciele:

- integrácia prostriedkov typu :
 - neurónove siete
 - fuzzy systémy
 - genetické algoritmy
 - evolučné programovania
 - umelý život
 - teória chaosu

a ich tvorba hybridných systémov, ktoré splňajú podmienky kladené pre umelo-inteligentný systém.

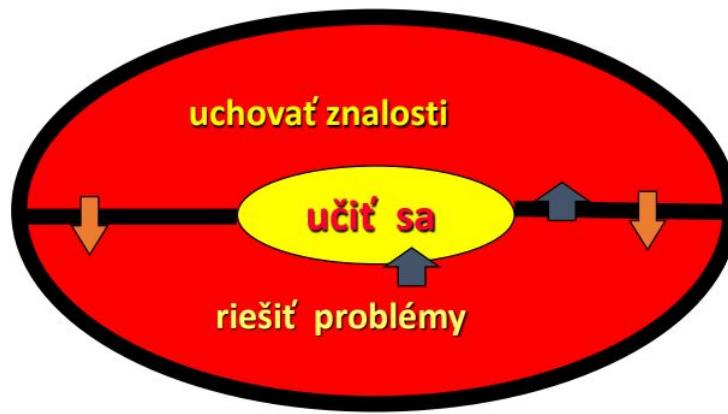
- hľadanie prostriedku, ktorý by bol aproximátorom neznámej funkcie na základe vstupno výstupných dát
- integrácia vedeckej komunity z týchto oblastí s cieľom modelovania biologickej inteligencie

Pojem **výpočtová inteligencia** bol navrhnutý prof. Bezdekom, ktorý ho vysvetľuje v kontexte s umelou a biologickou inteligenciou. Prof. Bezdek uvažuje o tzv. elementárnych prvkoch znalostí a nazýva ich **knowledge tidbits**. Tieto jednotky znalostí prispievajú ku zvýšeniu komplexity celého systému.

1.2 Inteligentné technológie

Z predchádzajúcich úvah je možné usúdiť, že inteligentný systém by mal mať nasledovné vlastnosti:

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



Obr. 1.3: Hlavné komponenty všeobecného systému UI

Všetky 3 požiadavky navzájom súvisia a doplňujú sa.

Inteligentný systém je teda systém, ktorý pri svojej činnosti využíva metódy UI. Obecne tieto metódy nazývame tzv. **inteligentné technológie**. Medzi prostriedky a metódy UI, ktoré tieto inteligentné technológie využívajú, môžu patriť:

- expertné systémy,
- prostriedky riešenia problémov s ohraničením,
- logické programovanie,
- neurónové siete,
- fuzzy systémy,
- neuro-fuzzy systémy,
- genetické algoritmy,
- metódy teórie chaosu,
- a ďalšie.

Z pohľadu používania reprezentácie vedomostí môžeme tieto prostriedky rozdeliť na :

- deklaratívne znalosti
- procedurálne znalosti

Ak napr. pri expertných systémoch je vedomosť reprezentovaná symbolmi, tak naproti tomu v neurónových sieťach je vedomosť ukrytá v numerických hodnotách a samotnej topológií siete. V poslednom prípade ide o množinu koeficientov (synaptické váhy) a o jednoduché matematické operácie (neuróny - topológia NS), ktoré sa aplikujú na vstup a následne produkujú výstup. Kým znalosti v klasických expertných systémoch sú deklaratívneho typu, tak v neurónových sieťach sú to procedurálne typy reprezentácie znalostí.

ďalším pohľadom na reprezentáciu znalostí je z hľadiska ich formy. Kosko definuje v [?] reprezentáciu znalostí ako ju vidíme v tabuľke 1.1.

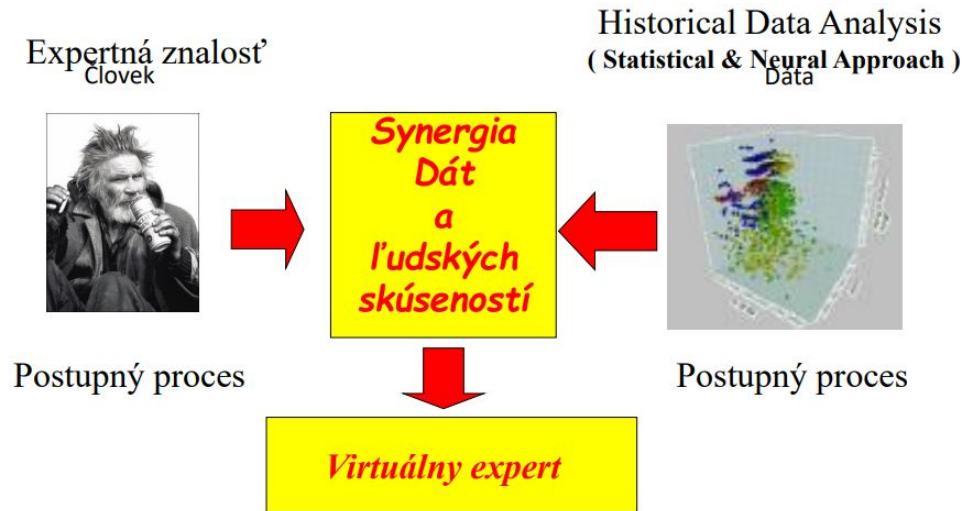
Teda štrukturovaná znalosť predstavuje prípad, kedy znalosti sú explícitne čitateľné a naviac je možné analyzovať samotnú reprezentáciu znalostí. Naopak, v prípade neštrukturovanej znalosti **nie je možné** analyzovať reprezentáciu znalostí. Určité nejasné postavenie v tejto tabuľke zastáva fuzzy

	symbolická	numerická
štrukturovaná	expertný systém	fuzzy regulátor
neštrukturovaná	-	neurónová siet

Tabuľka 1.1: Reprezentácia vedomostí podľa Dr. Koska

regulátor. Problém je v jednoznačnej definícii fuzzy množiny ako numerickej formy znalostí. Fuzzy množina pozostáva z prvku množiny, ktorý môže byť ľubovoľný symbol a z funkcie príslušnosti tohto prvku ku fuzzy množine. Z takého pohľadu nie je jednoznačne jasné, či skutočne ide o numerickú reprezentáciu znalostí.

Z pohľadu umelej inteligencie sa snažíme o synergiu poznatkov od experta a zo získaných dát. Tento prístup je najspoľahlivejší aj keď v súčasnosti sa preferuje hlavne inteligencia na základe dát - ak však ich máme k dispozícii. Niekedy je názor experta veľmi dôležitý a môže napomôcť efektívnej funkcionálite celého systému UI.



Obr. 1.4: Princíp integrácie a synergie znalostí experta a existencie dát s cieľom vytvorenia virtuálneho experta

UI ako vedná oblasť má v terminologickej rovine niekoľko fundamentálnych problémov. **Cieľom tejto časti je poukázanie na tieto problémy**

prejavujúce sa ako terminologická nejednotnosť v praxi. Jedným zo základných problémov je problém reprezentácie znalostí. V literatúre nájdeme nasledovné terminologické reprezentácie znalostí:

- symbolická verzus subsymbolická,
- symbolická verzus konekcionistická,
- symbolická verzus nesymbolická,
- symbolická verzus numerická.

Komentáre k tejto problematike nájdeme v v [?], [?].

Kapitola 2

Neurónové siete

2.1 Základné pojmy a princípy činnosti

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 \mathbf{W} ako maticu všetkých synaptických váh¹ neurónovej siete, tak pod učením budeme chápať stav kedy platí, že

$$\frac{\partial \mathbf{W}}{\partial \mathbf{t}} \neq 0 \quad (2.1)$$

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.

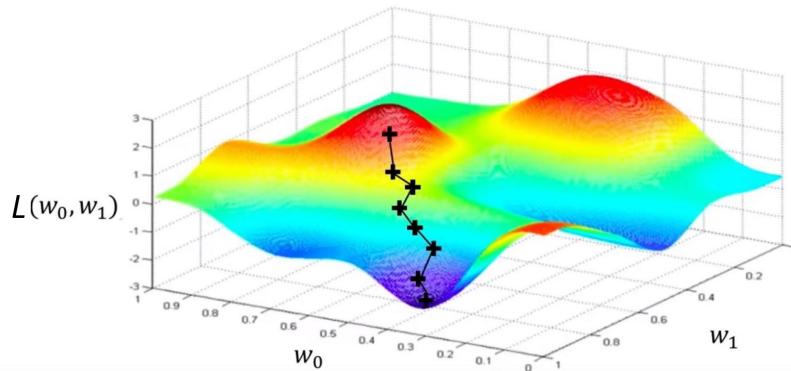
¹Pojem synaptická váha bude vysvetlený v časti [2.2.1](#)

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

$$\frac{\partial \mathbf{W}}{\partial t} = 0, \quad (2.2)$$

teda synaptické váhy sa **nemenia**.

Niekedy sa neurónové siete nazývajú aj ako bezalgoritmické systémy². Toto tvrdenie je pravdivé, ale vzťahuje sa **iba na fázu života** neurónových sietí. Naopak, vo fáze učenia prebieha v neurónových sietach **cieľavedomý** proces uchovávania poznatkov do synaptických vág neurónových sietí, ktoré sú nositeľmi znalostí.



Obr. 2.1: ilustračný obrázok $2+1$ rozmerného chybového priestoru a výpočtu gradientu na chybovom povrchu s cieľom hľadania globálneho minima chyby L

2.2 Základné pojmy teórie učenia

Učenie je základnou a podstatnou vlastnosťou neurónových sietí. Toto ich odlišuje od nám doposiaľ známejho používania počítačov, kde bolo potrebné vytvoriť algoritmus, podľa ktorého prebehol výpočet. Teda pri klasických

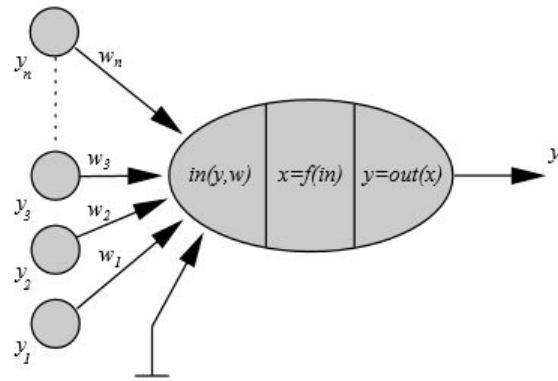
²algorithm-less systems

algoritmovch 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
- reprezentívna vzorka s je prvkom množiny $Y \times \{0, 1\}^m$. Preto reprezentativna vzorka predstavuje množinu usporiadaných dvojíc $s = ((y_1, d_1), (y_2, d_2), \dots, (y_m, d_m))$. 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 poznáť **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 extra hujú 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 dátá reprezentujú.
 2. **testovacia vzorka** – je to množina usporiadaných hodnôt, ktorá sa **používa vo fáze života** za účelom otestovania získaných znalostí počas učenia.

2.2.1 Základné prvky NN

Základným prvkom a procesnou jednotkou v NN je **neurón**. Štruktúra neurónu je na obrázku 2.2. Neurón pozostáva z nasledujúcich základných častí:

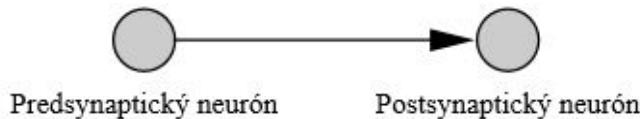


Obr. 2.2: Štruktúra neurónu

- vstup do neurónu (dendrit)
- prah neurónu - je hodnota θ_i , ktorá vlastne prispieva ku vstupu z externého sveta
- aktivačná funkcia neurónu $f(in_i)$, ktorej výsledkom je stav neurónu x_i
- 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 rozoznávame neuróny

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



Obr. 2.3: 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.

2.3 Poznámky k jednotlivým časťiam 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

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

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

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

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 $j = 1, \dots, N$, 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

$$x_i = f(in_i) \quad (2.5)$$

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.

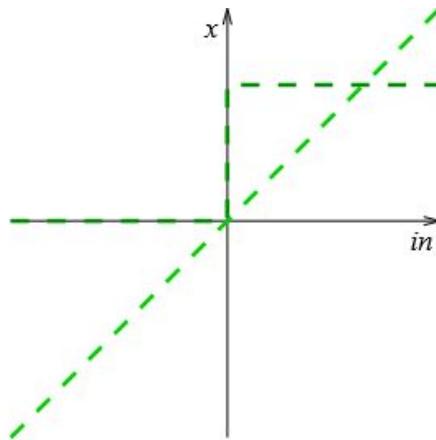
1. Lineárna funkcia

$$x_i = f(in_i) = in_i \quad (2.6)$$

2. Funkcia **signum**

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

Neuróny s takoto aktivačnou funkciu sa nazývajú tiež McCulloch-Pittsove neuróny. Tvar tejto funkcie je zobrazený na obr. 2.4



Obr. 2.4: Príklady linearnej a prahovej aktivačnej funkcie neurónu

3. Po častiach lineárna funkcia

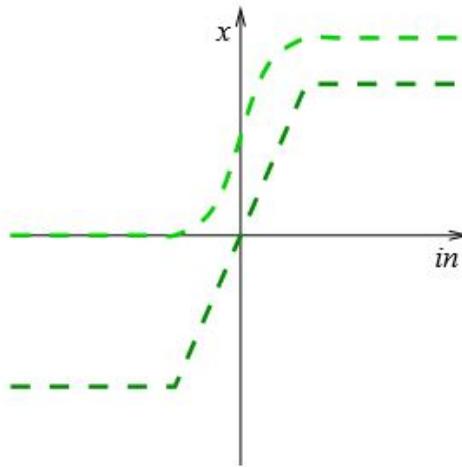
Táto funkcia má tvar

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

4. Sigmoidálna funkcia, ktorá má tvar

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

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



Obr. 2.5: Sigmoidálna a po častiach lineárna aktivačné funkcie neurónu

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

$$ou_i = f(x_i) \quad (2.10)$$

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

$$ou_i = O(x_i) = x_i = f(in_i) \quad (2.11)$$

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

2.4 Synaptické spojenia a váhy

Prepojenia medzi neurónmi patria medzi dôležité časti NN. Na týchto **orientovaných** 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 :

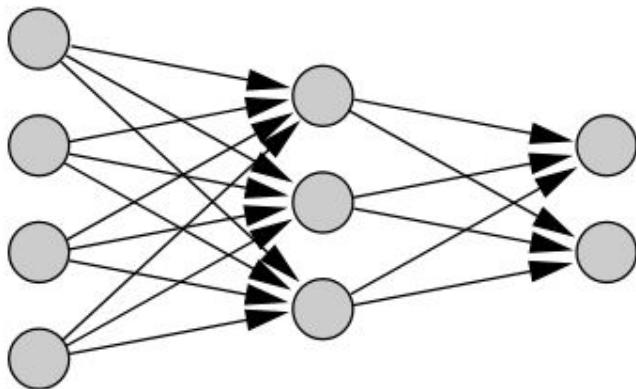
- kladné, teda **excitačné**
- záporné, teda **inhibičné**

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

2.4.1 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 ľahko 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. 4.1. V takých NN sú vrstvy pomenované. Rozlíšujeme 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
- výstupná je podobná ako skrytá vrstva, akurát je obvyklé, že výstup z tejto vrstvy vyúsťuje do externého sveta.



Obr. 2.6: Štruktúra doprednej NN

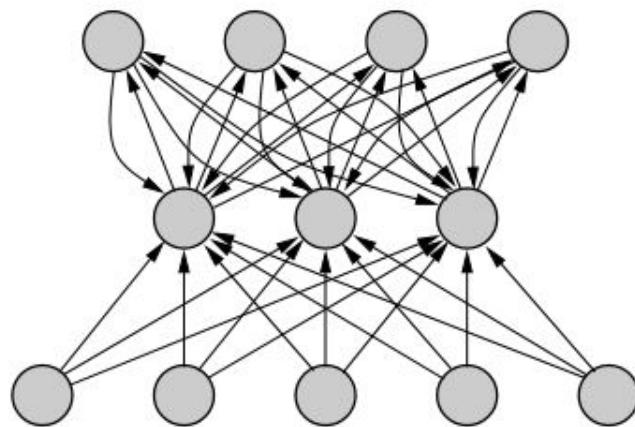
V náväznosti na túto situáciu rozpoznávame aj samotné neuróny ako **vstupné**, **skryté**, a **výstupné**.

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. 4.1.
- rekurentné NN (recurrent **RC NN**) - pri rekurentných sieťach je dosť ľahké rozdelenie vrstiev a neurónov na **vstupné**, resp. **výstupné**. Niektoré neuróny v rekurentných sieťach predstavujú vstupné ale aj výstupné typy neurónov a tým aj vrstiev (viď obrázok 2.7.). Špeciálnym prípadom sú tzv. čiastočne rekurentné NN, v ktorých je stanovená určitá požiadavka na štruktúru a na prepojenia. Napr. vrstvové čias- točne rekurentné siete pripustiajú šírenie signálu oboma smermi.

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

- synchrónne šírenie signálu - všetky neuróny menia svoj stav do taktu (prostredníctvom synchronizačných hodín)
- sekvenčné - neuróny menia svoj stav postupne pri šírení signálu



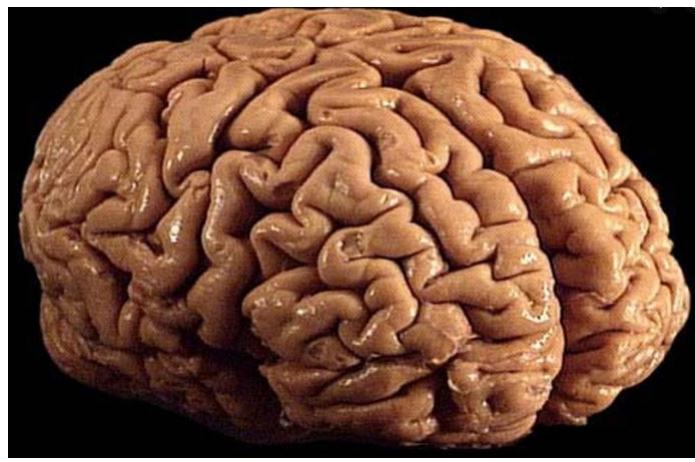
Obr. 2.7: Štruktúra RC NN

- blok-sekvenčné - aktivizujú sa len skupiny neurónov, podľa vopred určenej stratégie
- asynchronné - neuróny menia svoje stavy asynchronne, teda úplne nezávisle jeden od druhého

Kapitola 3

Mozog - procesor biologického systému

Poznanie činnosti biologických systémov ako aj ich modelovanie je a môže v budúcnosti byť veľkým zdrojom inšpirácie pre tvorbu technických systémov 21. storočia. Je predpoklad že inteligentné systémy budú predstavovať technické riešenia, ktoré budú mať svoju inšpiráciu v činnosti biologických systémov. Fakty, ktoré v tejto kapitole sú popísané dokumentujú úroveň zjednodušenia biologicky motivovaných systémov napr. neurónových sietí. Odbornosť výkladu je prispôsobená pre komunitu **nie biológie** znalých čitateľov. Na obrázku [3.1](#) je fotografia skutočného ľudského mozgu.



Obr. 3.1: Obrázok skutočného ľudského mozgu.

3.1 Biologicky motivované technické systémy

Dajme si niekoľko otázok a to nasledovných :

- čo sú to „**biologicky motivované neurónové siete**“?
- Aký má toto slovné spojenie vôbec význam?
- čím iným môžu byť umelé neurónové siete motivované, ak nie ich skutočným, neurobiologickým „vzorom“?
- Aké sú dôvody pre skúmanie týchto špecifických systémov?
- aký úžitok nám tieto systémy môžu priniesť a v ktorých oblastiach je ich prínos najvýraznejší?

Neurónové siete je dnes možné chápať ako výrazne interdisciplinárny smer vedeckej činnosti. Úspech pri ich používaní si vyžaduje určité znalosti z odborov ako

- matematika (vytvárajú sa zložité matematické modely, ktorých dynamický popis a analýza, napríklad z hľadiska stability, sú veľmi dôležité)
- umelá inteligencia (pôvod a využívanie podobných prostriedkov, vid ďalej),
- kybernetika (z jej hľadiska nie sú neurónové siete ničím viac ako nelineárnymi dynamickými systémami),
- neurobiológia (potreba poznáť to, čo modelujeme),
- výpočtová technika a programovanie (presnejšie povedané to, čo sa zahŕňa pod anglický pojem Computer Science - pretože všetky modely a aplikácie sa vytvárajú ako algoritmy a implementujú na počítačoch),
- fyzika (niektoré neurónové siete vychádzajú priamo zo štatistickej fyziky)
- a iné.

Pôvod neurónových sietí je najskôr možné spájať s umelou inteligenciou, vo vzťahu ku ktorej je možné definovať dve základné stratégie vysvetlovania kognitívnych procesov, a to takzvanú stratégiu zhora-dole (anglicky „top-down“) a stratégiu zdola-hore (anglicky „bottom-up“) (viď. napr. [?], [?], alebo [?]). Prvá stratégia je preferovaná takzvanou symbolickou vetvou

odboru umelej inteligencie (anglicky tiež občas označovanou ako „GOAI“ – Good Old Artificial Intelligence, čiže Stará dobrá umelá inteligencia, pretože pôvodne tento prístup dominoval celému odboru umelej inteligencie) a možno ju súhrnne opísť nasledovným výrokom Davida Marra [?]:

Vyššie [štrukturálne] úrovne sú do značnej miery nezávislé na nižších úrovniach. Preto je možné hľadať algoritmické riešenia bez potreby pochopenia ich fyzikálnej implementácie v mozgu.

Druhá stratégia, stratégia zdola-hore, vychádza z názoru, že kognitívne procesy prebiehajúce v biologických systémoch nie je možné vysvetliť a tým pádom ani modelovať bez pochopenia im podliehajúcich štruktúr a ich funkcií v mozgu.

Z hľadiska tohto delenia je možné neurónové siete jednoznačne zaradiť do druhej skupiny preferujúcej prístup zdola-hore.

V súčasnosti sa neurónové siete, spolu s genetickými algoritmami a s fuzzy logikou, začínajú zaraďovať pod spoločný názov Výpočtová inteligencia (anglicky Computational intelligence [?]). Tento vývoj je možné vysvetliť z viačerých hľadísk. Hlavným dôvodom je to, že neurónové siete je možné chápať ako paralelné systémy skladajúce sa z viacmenej autonómnych jednotiek (neurónov) vykonávajúcich určité výpočtové úkony. Na rozdiel od „GOAI“ sa tu teda nemanipuluje s nejakými vysoko abstraktnými symbolmi a hlavným prostriedkom nie je formálna logika, ale algoritmus definovaný ako sekvencia matematických úkonov. Z neurobiologického hľadiska je zaradenie neurónových sietí pod výpočtovú inteligenciu možné vysvetliť tým, že neurónové siete sú v stále väčšej miere čisto abstraktným matematickým modelom, bez opory v biologických systémoch. A toto je zároveň aj hlavný dôvod, prečo sa v tejto práci hovorí o „**biologicky motivovaných neurónových sietiach**“. Totižto, napriek súčasným trendom, pri vývoji sietí *Adaptive Resonance Theory (ART)*, ktoré sú v tejto práci primárny typom neurónových sietí, sa na neurobiologické aspekty tohto modelu kládol a kladie nemalý dôraz.

Cieľom tejto práce je teda:

1. Podať základnú charakteristiku biologických nervových systémov ako vzoru neurónových sietí.
2. Poskytnúť prehľad o neurónových sietiach typu *ART* a o spektre ich aplikácií.
3. Načrtnúť možnosti aplikácie sietí typu *Adaptive Resonance Theory* pri klasifikácii obrazu.

Jedným z motívov pre vypracovanie tejto práce v predkladanej forme bolo presvedčenie, že používanie určitého systému bez pochopenia súvislostí jeho vzniku a vývoja môže viesť k zavádzajúcim, a často chybným, záverom.

3.2 Modelovanie biologických neurosystémov

Popis súčasného stavu v oblasti modelovania biologických neurosystémov je vhodné začať krátkym náčrtom toho, čo o týchto systémoch v súčasnosti vieme (Pre hlbší rozbor súčasných znalostí o nervovom systéme viď. [?]). Priemerný ľudský mozog obsahuje odhadom 10 až 100 miliárd neurónov, pričom na každý neurón pripadá v priemere desaťtisíc synaptických spojení. Za hlavný spôsob predávania informácií medzi neurónmi sa považujú séria impulzov (anglicky „spike trains“), ktoré jednotlivé neuróny generujú. Špecificky sa za hlavnú informačnú zložku považuje stredná frekvencia generovaných impulzov. Táto frekvencia sa pohybuje okolo 80Hz. Použitím informácie o strednej dĺžke sérií impulzov sa dá ľahko odhadnúť, že jednou takouto sériou je možné preniesť informáciu na úrovni približne štyroch bitov. Z tohto je možné vyvodíť záver, že výkonnosť biologických systémov je založená v prvom rade na vysoko efektívnom paralelnom distribuovanom spracovaní informácií.

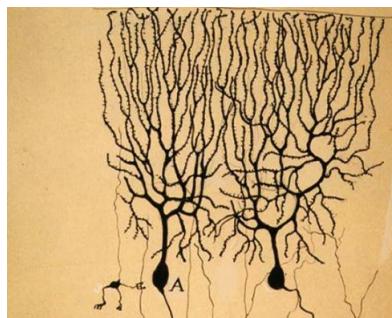
Štrukturálny popis nervového systému sa obyčajne robí na viacerých úrovniach. Tieto úrovne organizácie je možné krátko zhrnúť do nasledovnej tabuľky (Tabuľka ??). Podobne je možné definovať aj temporálne úrovne, a to od $10\mu s$ (otvorenie resp. zatvorenie jedného iónového kanálu v synapsii) až po dni a týždne (predstavujúce zmeny v pamäti napr. dlhodobou zmenu pokojového potenciálu synaptickej membrány). Z týchto faktov vyplýva množstvo závažných otázok. Napríklad: „S ktorými úrovňami by sa pri štúdiu mozgovej činnosti malo začať?“ Veľmi dôležitou a na teraz nezodpovedanou je aj otázka: „Do akej miery sa dajú jednotlivé úrovne organizácie modelovať nezávisle jedna na druhej?“ Jednoznačné odpovede na tieto otázky nateraz nie sú k dispozícii, preto sa vo všeobecnosti akceptuje každý prístup, ktorý môže objasniť aspoň niektoré hľadiska činnosti nervovej sústavy.

Základným stavebným prvkom modelovania pomocou neurónových sietí je neurón, chápáný ako výpočtová jednotka, ktorá má vykonávať jednoduché výpočtové úkony. Zjednodušene je možné rozdeliť neurón na tri časti (viď Obrázok 3.2):

- Dendritický strom, ktorým neurón prijíma signály z iných neurónov.
- Bunkové telo (soma), v ktorom sa uskutočňuje časová a priestorová

Štruktúra v nervovom systéme	Mierka
Centrálna nervová sústava	$1m$
Systémy (vizuálny, pohybový)	$10cm$
Mapy/stĺpce (senzorické mapy povrchu tela)	$1cm$
Siete	$1mm$
Neuróny	$100\mu m$
Synapsie	$1\mu m$
Molekuly	$1.10^{-10}m$

Tabuľka 3.1: Úrovne organizácie v nervovom systéme a ich veľkosti



Obr. 3.2: Purkyňova bunka v mozočku. Príklad komplikovanej štruktúry niektorých buniek v mozgu. Prevzaté z [?].

integrácia signálov. Na jeho rozhraní s axónom sa nachádza takzvaná spúšťacia zóna, ktorá v prípade dostatočnej úrovne vstupných signálov vyšle impulz ďalším neurónom.

- Axón, ktorým sa šíri impulz zo spúšťacej zóny do synaptických zakončení, kde sa tieto signály predávajú ďalším neurónom.

Obrázok 3.2 zároveň ilustruje aj to, aký komplikovaný neurón v skutočnosti môže byť. Popísanie matematicky, napríklad časovým a priestorovým integrálom, štruktúru uvedenú na tomto obrázku je takmer nemožné, a výpočtová náročnosť s tým spojená by model, ktorý by túto štruktúru verne zachytával, urobila nepoužiteľným. Naviac je potrebné pripomenúť, že sa pri uvedenej definícii neurónu ako základnej výpočtovej jednotky ignoruje celá

40 KAPITOLA 3. MOZOG - PROCESOR BIOLOGICKÉHO SYSTÉMU

štrukturálna komplexnosť na synaptickej úrovni (viď ďalej). Nad neurónovou štruktúrou sa nachádza ďalšia organizačná štruktúra, a to sú mapy respektíve stĺpce. Dobre známymi príkladmi z tejto oblasti sú takzvaní homunkulus oblasti popisujúci senzorické resp. motorické mapy v mozgu, alebo mapy zodpovedajúce povrchu sietnice alebo zvlášť každému oku vo vizuálnom kortexe. Čo sa týka systémovej štrukturálnej úrovne, tu je situácia značne komplikovaná. Na jednej strane sú dopodrobna popísané celé časti mozgu, na druhej strane tie časti, ktoré sú z hľadiska vyšších mozgových funkcií najzaujíma vejšie, sú známe len čiastočne, a naviac ich pozícia, veľkosť a prepojenie sú viacmenej individuálne pre každého jedinca. Z hľadiska celej centrálnej nervovej sústavy je zrejmé, že presný popis bude možný až po získaní dostačného množstva informácií z nižších štrukturálnych úrovní.

Veľmi dôležitou problematikou z hľadiska modelovania dôležitou otázkou je otázka komunikácie v nervovej sústave. Komunikácia sa uskutočňuje na úrovni neurónov a to prostredníctvom synapsí. V zásade existujú dva druhy synapsí:

- Elektrické synapsie, ktoré predstavujú priame spojenie medzi neurónmi, prostredníctvom ktorého sa šíria elektrické signály z jedného neurónu na iný. Tieto spojenia sú preto veľmi rýchle, ale zároveň majú krátky dosah kvôli nízkej vodivosti bunkovej membrány.
- Chemické synapsie, ktoré sa realizujú chemickou cestou, a to prostredníctvom takzvaných neurotransmitterov (neuronosičov), uvoľňovaných pri príchode elektrického signálu presynaptickými neurónmi do synaptickej štrbiny. Ak bolo do synaptickej štrbiny uvoľnené dostatočné množstvo neurotransmitterov, vyvolá ich chemická reakcia s receptormi v postsynaptickom neuróne takzvaný akčný potenciál, ktorý vlastne predstavuje vyššie zmienený impulz. Z tohto popisu je zrejmé, že chemické synapsie sú podstatne pomalšie ako elektrické synapsie. Táto ich nevýhoda je ale vysoko kompenzovaná možnosťou zosilnenia signálu pri jeho prechode synapsiou (zmenou množstva uvoľnených neurotransmitterov). Preto sa dá povedať, že rozhodujúca väčšina komunikácie v mozgu sa uskutočňuje prostredníctvom chemických synapsí.

Existuje aj oveľa komplikovanejší spôsob chemickej komunikácie medzi neurónmi, ktorého výsledkom môže byť zmena aktivity v jadre neurónu. O tom, aké dôsledky má táto zmena aktivity na fungovanie daného neurónu sa vie veľmi málo, ale z teoretického hľadiska sú tieto dôsledky neobmedzené, čo je pre modelovanie vážnym problémom.

Vplyv na komunikačné vlastnosti synapsí má celý rad faktorov. Názor-

ným príkladom je známy jed Kurare používaný juhoamerickými indiánmi na ochromenie ich koristi. Tento jed funguje tak, že zablokuje chemické receptor v svalových bunkách, čím sa organizmus úplne ochromí.

Ako sa uskutočňuje učenie v nervových systémoch vieme dnes vysvetliť len vo veľmi obmedzenej miere, a aj to väčšinou len pre nižšie živočíšne druhy. V zásade sa dá povedať, že proces učenia môže viesť k jednému z troch možných výsledkov:

1. Vytvorenie nového axónu alebo odumretie existujúceho axónu.
2. Zmena v sile synaptickej väzby (uskutočnená napríklad zmenou veľkosti synapsie, počtu synapsí, alebo zmenou koncentrácie neurotransmitterov).
3. Dlhodobou zmenou pokojového potenciálu synaptickej membrány.

Ktorá z týchto metód sa uplatní v určitom špecifickom prípade je vo všeobecnosti neznáme, a špeciálne s ohľadom na rast axónov nevieme vôbec vysvetliť, čo určuje smer, ktorým axón rastie.

3.2.1 Použitie neurónových sietí v modelovaní

Prvým modelom, ktorý vlastne začal celú éru neurónových sietí, bol model McCullocha a Pittsa [?]. Jeho hodnota je ale z dnešného pohľadu skôr historická. Vo všeobecnosti sa ale dá povedať, že veľká časť toho, čo sa dnes zahŕňa pod pojmom neurónové siete, bola skutočne v tej či onej forme aj použitá pre modelovanie neurobiologických systémov, respektíve v modeloch vysvetlujúcich určité psychologické procesy. Nasledujúca časť podáva stručný prehľad týchto modelov.

Modelov, ktoré by bolo skutočne možné namapovať na konkrétnie mozgové štruktúry, je len niekoľko. Vo všeobecnosti sa tieto modely vzťahujú na vizuálne alebo senzoricko-motorické mapy, pretože to sú časti mozgu, ktoré sú relatívne dobre popísané. Pravdepodobne najznámejším z týchto modelov je Von der Malsburgov model [?] z roku 1973 popisujúci samоорganizáciu neurónov citlivých na orientáciu čiar v primárnom vizuálnom kontexte. Veľa nasledovných prác, napríklad Kohonen [?], vychádza metodicky z úplne rovnakých pravidiel, ako pôvodná Von der Malsburgova práca. V rámci biologicky inšpirovaných modelov je zvyčajne cieľom vysvetliť určitý biologický alebo psychologický fenomén. Takým fenoménom môže byť napríklad vznik asociatívnej pamäte (Hopfield [?]), vytváranie podmienených reflexov (Grossberg [?]), schopnosť nepretržitého učenia sa a selektívneho

zabúdania (Grossberg [?]), a mnoho ďalších. U týchto modelov je charakteristické, že nie je možné exaktne vymedziť, ktoré mozgové oblasti popisujú, respektíve, či neurón v modeli reprezentuje jeden alebo celú skupinu reálnych neurónov. Na druhej strane ale tieto modely nezahŕňajú žiadne výpočty alebo úkony, ktoré by sa v reálnom nervovom systéme nemohli uskutočniť.

Do skupiny neviazaných modelov je možné zaradiť rôzne modely, v ktorých sa použila neurónová siet bez hlbšej analýzy toho, či daný typ neurónovej siete môže definovanú funkciu v mozgu skutočne vykonávať. často sa pri týchto modeloch používa neurónová siet s učením so spätným šírením chyby (anglicky Backpropagation of Error Learning). Známych je niekoľko príkladov použitia v modeloch sluchového vnímania, napríklad Neti [?].

Existuje aj niekoľko ďalších modelov. Tieto modely vychádzajú z čo najvernejšieho opisu skutočného neurónu. Do tejto skupiny patrí Integrate and Fire Neuron, alebo známy simulačný systém Genesis [?]. Cieľom týchto modelov je, v porovnaní s vyššie uvedenými modelmi, odpovedať na oveľa základnejšie otázky, ako napríklad, ako sa môže v sieti reálnych neurónov uskutočňovať operácia násobenia, prípadne či sa tam môžu uskutočňovať operácie Boolovskej logiky.

Súhrne je možné povedať, že neurónové siete z hľadiska modelovania biologických nervových systémov musia spáňať dve požiadavky. Prvou je maximálna vernosť z hľadiska skutočného systému. Druhou požiadavkou je jednoduchosť modelu z matematického hľadiska, pretože tieto modely by mali byť použiteľné aj výskumníkmi, ktorí nemajú hlboké matematické resp. technické znalosti. Tieto dve protirečivé požiadavky sú jednou z príčin, prečo existuje taká široka paleta zásadne odlišných modelov.

3.3 Inžinierske aplikácie

V technických oblastiach dnes existuje veľké množstvo aplikácií založených na neurónových sietiach. Tu vyvstáva otázka, prečo je z technického hľadiska zaujímavé používať napríklad pri rozpoznávaní obrazov, ovládaní mechanických systémov, analýze a syntéze zvuku, atď., riešenia založené na znalostiach z biológie. Všeobecne je možné odpovedať, že riešenia používané biologickými systémami sú zaručene funkčné a z mnohých hľadísk aj vysoko efektívne. Napríklad, ak vieme, že pri teste zameranom na vizuálne rozpoznávanie objektov trvá človeku rozpoznanie určitého objektu približne 500 ms, znamená to pri započítaní rýchlosť šírenia signálov v nervovom systéme, že medzi vstupom a výstupom rozpoznávacieho systému človeka nemôže byť viac ako približne sto krokov (tento záver môžeme urobiť bez toho, aby sme

presne vedeli, aké kroky boli vykonané). To inými slovami znamená, že príroda pozná algoritmus, ktorý dokáže úlohu analýzy vizuálneho prostredia vykonať v sto paralelných krokoch, zatiaľčo súčasné technické riešenia vyžadujú oveľa väčšie množstvo výpočtových kapacít.

3.3.1 Prístupy a metódy

Zatiaľčo primárnymi požiadavkami na neurónové siete z hľadiska neurobiologického modelovania sú ich verność a vysvetľovací potenciál, technické aplikácie si v prvom rade vyžadujú, aby bol systém stabilný, rýchly a podľa možností čo najjednoduchší (napr. z hľadiska počtu nastaviteľných parametrov), a to pri zachovaní potrebných vlastností. Preto sa oveľa väčší dôraz kladie na matematickú stránku modelu. Existuje len niekoľko modelov, pre ktoré je zaručená stabilita (napr. Perceptron [?], alebo ART [?]), a ešte menej je takých, pre ktoré je zaručené, že v procese učenia dokonvergujú k optimálnemu riešeniu (napr. Adaline/LMS algoritmus [?]). Sila neurónových sietí je ale v tom, že umožňujú nájdenie relatívne dobrého riešenia v akceptovateľnom čase. Z hľadiska Umelej inteligencie teda ponúkajú neurónové siete riešenie s presne opačnými vlastnosťami, ako majú riešenia ponúkané symbolickou Umelou inteligenciou, kde je obyčajne zaručené nájdenie optimálneho riešenia ale veľmi často sa čas potrebný na nájdenie tohto riešenia prakticky rovná nekonečnu. Z tohto hľadiska je neurónové siete možné chápať ako heuristické algoritmy schopné pri riešení využívať zovšeobecnenia, prípadne doplniť chýbajúcu informáciu. Rovnako dôležitým aspektom modelu, ako jeho stabilita, je aj otázka výpočtovej náročnosti modelu. Vo všeobecnosti sú prakticky použiteľné modely, ktoré môžu byť popísané v diskrétnom čase (t.j. nepožadujú numerické riešenie diferenciálnych rovníc), a modely, ktoré nie sú založené na stochastických vlastnostiach systémov. Tieto obmedzenia sú dané výpočtovou náročnosťou numerickej integrácie, resp. štatistického vyhodnocovania systémov.

Hlavne z dôvodov rozvoja technických aplikácií neurónové siete dnes už opustili rámec biologicky inšpirovaných systémov a existuje niekoľko typov neurónových sietí, v ktorých sa používajú aj poznatky z iných vedných oblastí. Príkladom tohto trendu je metóda založená na znalostiach zo štatistickej fyziky anglicky nazývaná Simulated Annealing (Simulované žíhanie) [?]. Táto metóda vychádza z analógie medzi správaním sa fyzikálneho systému s mnohými stupňami voľnosti v stave tepelnej rovnováhy pre niekoľko rozdielnych hodnôt teploty, tak ako to skúma štatistická fyzika, a medzi problémom nájdenia minima danej funkcie závislej na veľkom množstve parametrov, ktorý sa často rieši v oblasti kombinatorickej optimalizácie. ďal-

ším príkladom systémov inšpirovaných fyzikou je takzvaný Boltzmanov stroj [?], ktorý podobne predstavuje neurónovú sieť vychádzajúcu zo stochastickej formy učenia.

3.4 Budeme sa potrebovať učiť ?

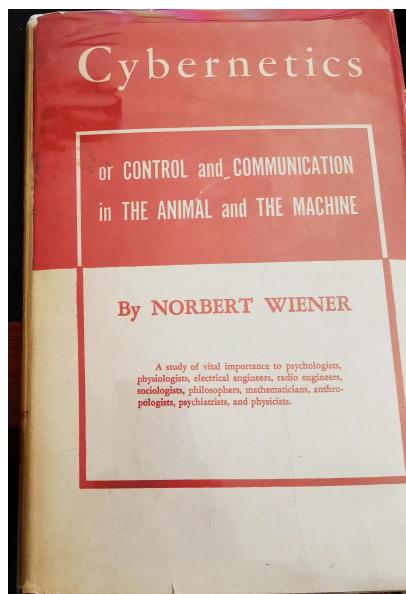
Mozog ako časť nervovej sústavy sa objavuje u primátov a vyšších biologickej systémov. Mozog človeka predstavuje jeden z najzložitejších orgánov, ktorého činnosť ešte **nie je úplne objasnená**. Cieľom mnohých vedných odborov je objasnenie procesov, ktoré sa realizujú v mozgu a súčasne získanie inšpirácie pre tvorbu dokonalejších technických systémov. Kognitívne procesy, organizácia a činnosť pamäte, ukladanie znalostí a ďalšie procesy, ktoré prebiehajú v mozgu ako napr. vedomie sú veľkou neznámou pre vedeckú komunitu. Plné pochopenie činnosti mozgu je súčasťou v nedohľadne ale prognózy výskumu v tejto oblasti sú veľmi zaujímavé.

Ak budú jasné princípy uchovávania informácií v mozgu a ich rozloženie je zrejmé, že sa bude intenzívne pracovať na problematike prenosu informácií do mozgu umelou cestou. Táto skutočnosť bude niesť so sebou mnohé problémy, ktoré budú riešiť okrem lekárov hlavne psychológovia.

Týmto spôsobom bude snaha o odstránenie procesu učenia z ľudskej činnosti a jeho nahradu priamou aplikáciou poznatkov do štruktúr mozgu.

Veľmi veľký význam má knižná publikácia, ktorá zohrala kľúčovú úlohu pri vzniku kybernetiky a umelej inteligencie je publikácia prof. Norberta Wienera **Cybernetics** z roku 1948. Pozri obrázok [?].

Prvýkrát sa mozog spomína v písomných dokumentoch zo 17 storčia pred Kristom, kedy na papyrusovom zázname bol spomínaný pojem tzv. **ys**. Tento historický dokument bol analyzovaný Dr. Jamesom Breastedom a publikovaný v roku 1930, kde pomocou tohto slova bol popísaný mozog pri fraktúre lepky dvoch Egypťanov. Ďalšie práce pojednávajúce o mozgu sú známe od Aristotela v roku 355 pre našim letopočtom (pr.n.l.), kde pojednáva o pamäti, snoch a iných problémoch. Súčasne Platon v roku 400 pr.n.l. tvrdí, že v mozgu je uchovaná naša večná duša. V roku 130 n.l. Roman Claudius Galenus spoznal, že senzorické nervy môžu byť poškodené, bez vplyvu na motorické centrum. Tvrdil, že nervy sú duté kanálky, kde tečie špeciálna tekutina, ktorá je dodávaná z potravy a tečie cez pečeň. O mnoho pozdejšie až okolo roku 1660 Isaac Newton začína tvrdiť že cez nervy sa šíria vibrácie.



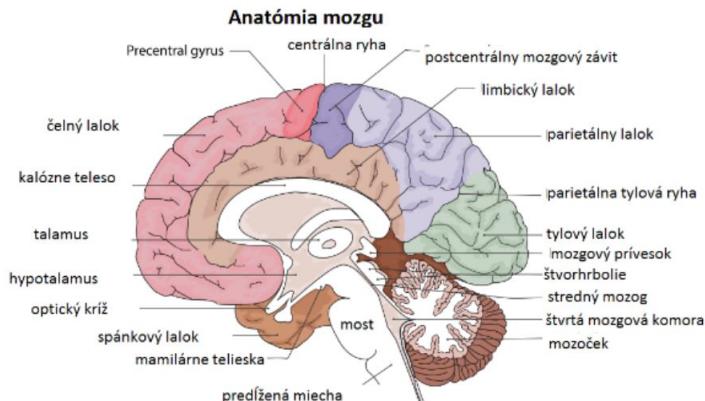
Obr. 3.3: Kľúčová publikácia, ktorá vytvorila základ kybernetiky a umelej inteligencie

V roku 1791 Luigi Galvany potvrdzuje, že existuje určitý elektrický potenciál v nervovej sústave žaby, čo aj dokazuje svojim experimentom. Celé 19. storočie skúmal tok signálov vo vnútri nervov. Odhady boli v rýchlosťach okolo 140 km/hod. Veľmi významným krokom výskumu mozgu bola práca Jána Purkyňu, ktorý prvý identifikoval samostatný neurón. Jeho výskum priniesol konštatovanie, že neurón pozostáva z tela a siete vlákien. Nervový systém pozostáva z neurónov a tieto sú prepájané synapsami, ktoré značne spomaľujú toky signálov. Signály sa nazývajú tzv. **spiky** a sú pulzného charakteru o veľkosti cca. 90 milivolt. Ďalšie zaujímavé informácie o mozgu je možné nájsť v publikácii ??...

O anatómii mozgu sa dnes vie veľmi veľa. **O mnoho menej sa vie o detailoch jeho fungovania.** Výskumu mozgu sa venujú vedci z rôznych oblastí medicíny a psychológie. Pre komunitu informatikov je mozog príkladom vrcholnej a geniálnej tvorby masívne paralelného a vysokovýkonného procesora. Mozog v ľudskom tele predstavuje orgán, ktorej základná anatómická štruktúra je na obrázku 3.4. Stručný popis časti mozgu predstavuje popis základných častí a nekladie si ambície na detailný anatómický popis. Medzi základné anatómické prvky patria nasledovné časti:

46 KAPITOLA 3. MOZOG - PROCESOR BIOLOGICKÉHO SYSTÉMU

- mozog je rozdelený na dve hemisféry a ľavá hemisféra kontroluje pravú časť tela a pravá hemisféra ľavú časť tela
- mozgové hemisféry, ktorá sa skladá hlavne zo šedej kôry mozgovej (cortexu), ktorej hrúbka je od 1 mm do 4.5 mm a celková jej plocha je 2200cm^2 . V mozgu teda existuje pravá a ľavá hemisféra. V ľavej hemisfére je tzv. Brocovo centrum reči, ktoré lokalizoval v roku 1864 Paul Brocca na lekárskej fakulte v Paríži. Cortex zmapoval Brodman a je známa tzv. Brodmanova mapa projekčných oblastí Cortexu, ktorá definuje celkom 52 projekčných oblastí v tejto časti mozgu.
- mozoček (cerebellum), ktorý sa venuje problémom rýchlosťi, rovnováhy a pohybu a ich koordinácii
- medzimozog, ktorý hlavne zahrňuje thalamus a hypothalamus. Thalamus sa venuje spracovaniu informácií z cortexu a hypothalamus vybrané procesy v ľudskom tele.
- stredný mozog, most mozgu, predĺžená miecha na ktorý je napojený mozoček - dostávajú sa k nim informácie z tváre a povrchu hlavy a krku a koordinuje pohyb v tejto oblasti
- miecha - vstupujú do nej informácie z povrchu tela a cez ňu idú základné motorické informácie.



Obr. 3.4: Základné oblasti ľudského mozgu

Anatomicky mozog v priečnom reze pozostáva z Cortexu tzv šedej kory mozgovej a tzv. bielej vrstvy mozgovej. Podstatnú časť šedej kôry mozgovej tvoria Somy neurónov a na druhej strane bielu časť mozgu tvorí splet

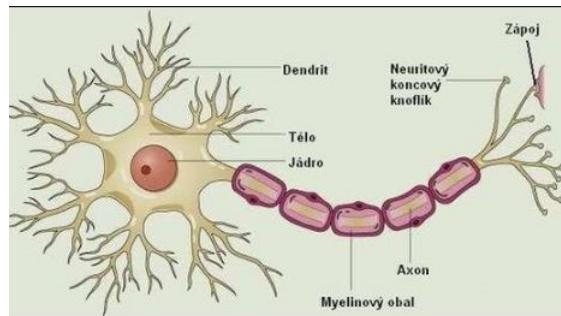
axonových vlákien. Niektoré axonové vlákna (hlavne dlhšie) sú pokryté tzv. Myelinovým povlakom, ktorý je prerusovaný tzv. Ranvierovým zárezom. Povlak vytvára podmienky pre rýchle šírenie signálov po axonovom vlákne. V roku 1976 Prof. Bullock prezentoval 46 rôznych parametrov neurónu. Z tohto je zrejme že úroveň zjednodušenia používaných umelých neurónových sietí je veľmi vysoká a z d'aleka nevystihuje biologickú realitu neurónu.

Jedným zo zaujímavých faktom mozgu je jeho váha. Samozrejme veľkosť ako aj váha mozgu **nie je funkciou inteligencie**. V priloženej tabuľke predstavujeme stredné hodnoty váh mozgu rôznych biologických systémov [3.2.](#)

48 KAPITOLA 3. MOZOG - PROCESOR BIOLOGICKÉHO SYSTÉMU

V priemere je šírka ľudského mozgu cca. 140 mm, dĺžka cca. 167 mm a výška cca. 93 mm. Základnými stavebnými časťami mozgu sú **neuróny**. Obecne mozog pozostáva z neurónov a tzv. gluiových buniek/footnotenázov buniek je odvodený od slovíčka glu (lep). Ich funkcia je komplementárna a podporná. Týchto buniek je omnoho viac ako neurónov a ich pomer v rôznych častiach mozgu sa pohybuje od 1:1 až do 1:40. Základná anatómia neurónu je na obrázku [??](#). Neurón sa skladá z nasledovných častí:

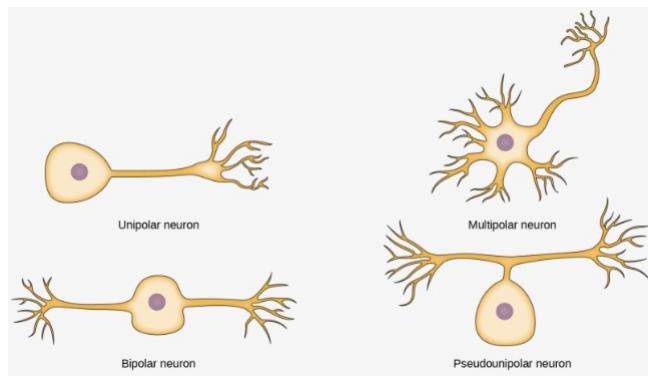
- Soma - telo neurónu,
- dentrity - vstupy do Somy
- axon - výstup zo somy (vždy je iba jeden axon),
- synapsia - synaptické prepojenie medzi axonom jedného neurónu a dendridom druhého neurónu.



Obr. 3.5: Základná štruktúra biologického neurónu.

V mozgu existuje niekoľko typov neurónov napr. unipolárne a bipolárne neuróny [3.6](#) ako aj multipolárne neuróny [??](#).

Veľmi zaujímavou je otázka počtu neurónov v mozgu. Odhaduje sa že v ľudskom mozgu - v cortexe je 10 miliard neurónov uložených zhruba v 6 vrstvách. **Celkový cortex má hrúbku od 1.5 - 4.5 mm.** Taktiež sa odhaduje, že v mozgu je cca. 60 000 miliar synaptických spojení. Súčasne v mozgu môže byť neurón prepojený s iným neurónom cca. 10 až 100 tisíc prepojeniami. Z uvedeného je zrejmé, že zložitosť takého systému je neporovnatelná aj so súčasnými paralelnými systémami. Hustota neurónov v cca. $7\text{-}8 \cdot 10^4$ neurónov na $1mm^3$ ale v cortexe je tato hodnota ešte výraznejšia.



Obr. 3.6: Dva typy unipolárnych neurónov jeden bipolárny

Rozmer Somy (tela) neurónu je niekoľko mikrometrov až do 10 mikrometrov. Dentity majú dĺžku cca. 3 mm. Axon môže mať rôznu dĺžku a môže dosahovať aj 1 meter.

Pri vývoji mozgu v embriu sa mozog vyvíja veľmi rýchle až 250 000 neurónov za minútu a naproti tomu pri starnutí človeku odumiera denne cca. 10 000 neurónov. Odhaduje sa že v 75 rokoch je to úbytok cca. 0.2-0.5 percent neurónov. Je dokázaná tvorba tŕňov na dentritoch ako aj neustála zmena prepájania neurónov v mozgu, ktorá koreluje s procesom učenia u ľudí.

50 KAPITOLA 3. MOZOG - PROCESOR BIOLOGICKÉHO SYSTÉMU

jedinec	váha mozgu v gramoch
dospelý človek	1300-1400 (muži), 1200 (ženy)
Pitecantrupus (praveký človek)	850-1000
novonarodené dieťa	350-400
gorila	465-540
šimpanz	420
orangutan	370
krava	425-458
kôň	532
slon	6000
delfín	1500
veľryba	7800
mrož	1130
ťava	762
žirafa	680
hroch	580
tiger	264
ovca	140
pes	72
mačka	30
zajac	10-13
aligátor	8.5
sova	2.2
potkan	2
škrečok	1.4
korytnačka	0.3
jašterica	0.08

Tabuľka 3.2: Tabuľka hmotnosti mozgu rôznych jedincov v gramoch

Kapitola 4

Učenie neurónových sietí a ich princípy

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 neelogické, 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 zatriedovať 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).

4.1 Paradigmy kontrolovaného učenia

Filozofia kontrolovaného učenia (teda spôsob zmeny SV) je ovplyvnený prítomnosťou **učiteľa** v celom procese učenia. **Prakticky to znamená, že NN musíme ponúknutť 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 činnosti¹ (**reinforcement learning**)

Poznámky k jednotlivým typom :

1. Učenie na základe korekcie chyby

Tento prístup predpokladá zmenu SV ako funkciu premenej e_i , kde e_i predstavuje rozdiel medzi očakávaným stavom neurónu $\hat{y}(ev_i)$ a vypočítaným stavom neurónu $\hat{y}(x_i)$ v procese učenia. Teda

$$e_i = ev_i - x_i \quad (4.1)$$

Potom môžeme napísat' všeobecný vzorec pre výpočet zmeny SV pre spojenie medzi výstupným neurónom "i" do neho vstupujúcim neurónom "j" v tvare

$$\Delta w_{ij} = \gamma x_j e_i \quad (4.2)$$

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).

2. Stochastické učenie

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

¹veľmi voľný preklad

- navrhne sa stochastická zmena SV a vypočíta sa **energia NN**.
- ak zmena priniesla **zniženie 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 napísť všeobecný tvar rovnice učenia:

$$\Delta w_{ij} = \gamma (r - \theta_i) e_{ij} \quad (4.3)$$

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

$$e_{ij} = \frac{\partial \ln g_i}{\partial w_{ij}} \quad (4.4)$$

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

$$g_i = P(x_i = ev_i | W_i, \Lambda) \quad (4.5)$$

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.

4.2 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 ponúknutím (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

$$|\Delta W(t) - \Delta W(t+1)| \leq \epsilon \quad (4.6)$$

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 stranach synapsie sú aktivované naraz (synchrónne), potom SV synapsie sa zvýší.
- ak 2 neuróny na opačných stranach synapsie sa aktivizujú v rôznych časoch (asynchronne), potom SV synapsie sa zníži alebo $\rightarrow 0$.

Takéto synapsie NN, ktoré splňujú spomenuté zásady nazývame **Hebbove synapsie**. Globálne potom môžeme Hebbove synapsie charakterizať 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 "jä i" ("j" vstupuje do i") nasledovne :

$$\Delta w_{ij}(t) = \text{funkcia}(x_i(t), x_j(t)) \quad (4.7)$$

resp.

$$\Delta w_{ij}(t) = \gamma x_i(t) x_j(t) \quad (4.8)$$

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 $x_i, i = 1, \dots, M$. Ak sledujeme NN z hľadiska dynamiky potom označme

$$\frac{\partial x_i}{\partial t} = V_i(X) \quad (4.9)$$

kde $X = (x_1, x_2, \dots, x_{M+N_z+N_0})$ je vektor aktivačných hodnôt neurónov v celej NN, teda všetky neuróny.

V_i predstavuje zmenu aktivácie neurónu za jednotku času. Ako k tejto zmene prispel iný neurón, môžeme vyšetriť, ak uvažujeme

$$\frac{\partial V_i}{\partial x_j} = \begin{cases} \leq 0 & \forall j \neq i \\ > 0 & \forall j \neq i \end{cases} \begin{array}{l} \text{concurrent learning} \\ \text{cooperative learning} \end{array} \quad (4.10)$$

Pre konkurenčné učenie platí tzv. princíp **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 **vítazom** a IBA váhy, ktoré smerujú k tomuto neurónu sú adaptované podľa nejakého pravidla.
- učenie sa končí ak adaptácia konverguje resp je blízka nule.

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

4.3 Globálna stabilita a konvergencia NN

V predchádzajúcich častiach už bola NN pripomatená ako dynamický systém, ktorý adekvátnie 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.

4.3.1 Globálna stabilita NN

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

- GS NN počas procesu učenia³
- GS NN mimo procesu učenia

Všeobecne pod GS NN rozumieme stav, kedy neuróny NN zotravávajú v nejakom stave⁴. Ak si predstavíme NN, ktorá má M neurónov, ktoré majú stavy $x_i \quad i = 1, \dots, M$, potom pod globálnou stabilitou rozumieme vo všeobecnosti stav

²winner takes all

³proces učenia je charakteristický zmenou váh NN v čase

⁴nemusí to byť taký stav, aký my chceme

$$\frac{\partial x_i}{\partial t} \rightarrow 0 \text{ pre } i = 1, \dots, M \quad (4.11)$$

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

$$R^M \rightarrow R^1 \quad (4.12)$$

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

1. ak

$$G(\mathbf{x}(t)) = \frac{\partial \mathbf{x}(\mathbf{t})}{\partial \mathbf{t}} \quad (4.13)$$

a platí že $G(\mathbf{x}(t)) = 0$ ak všetky $x_i = 0$

2. $G(\mathbf{x}(t))$ je holomorfna, t.j. existujú prvé derivácie podľa x_i
3. $\sum_{i=1}^M x_i(t) \leq H$ pre $t \geq t_0$, teda po nejakom čase ($\geq t_0$), bude suma stavov $x_i(t)$ menšia ako konštantu H .
4. ak vieme zstrojiť funkciu $L(X)$ pre ktorú platí, že

$$\sum_{i=1}^M \frac{\partial L}{\partial x_i} \leq 0 \quad (4.14)$$

pre všetky x_i

potom funkcia L sa nazýva **Ljapunovova funkcia energie NN**, popisuje GS NN a pre stabilné NN $L \rightarrow 0$. Samotný pojem **Ljapunovova funkcia** predstavuje ľubovoľnú funkciu, ktorá pri jej extréme dosiahne stabilnú neurónovú sieť.

Táto funkcia skutočne realizuje transformáciu 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 teóra).

Pre FF NN má význam hovoriť o GS NN len v procese učenia. V procese mimo učenia⁵ tam principiálne nemôže dôjsť k nestabilite NN. Pri RC NN však má význam hovoriť o stabilité pri učení ako aj mimo učenie, 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 nestabilnou⁶.

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

4.3.2 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

$$|x_i^n - x| \leq \epsilon \quad (4.15)$$

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

⁵váhy NN sa nemenia

⁶mimo učenia, cez rekurentné synaptické prepojenia

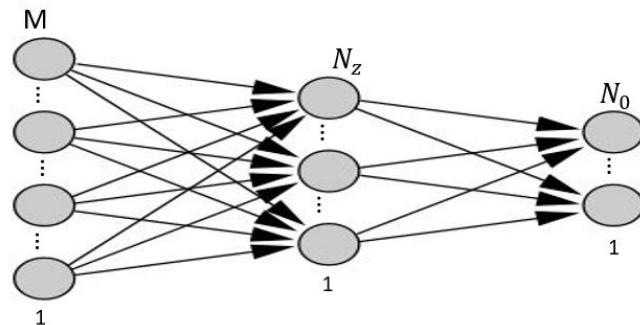
4.3.3 Dôležité poznámky k návrhu FF NN

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 **aproximáciách 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 approximačné ciele NN

4.3.4 Návrh topológie NN

V prípade aproximácie funkcie celý proces v konečnom dôsledku musí viesť k approximácii s nejakou presnosťou. Je to vlastne niečo podobné ako approximácia funkcie nejakým radom. Ak si predstavíme jednoduchú NN s jedným vstupom a jedným výstupom pri predpoklade, že aktivačné funkcie



Obr. 4.1: Štruktúra doprednej NN

Kde podľa obrázku 4.1 sú na

- vstupe neuróny indexované od $l = 1, \dots, M$
- neuróny v skrytej vrstve sú indexované $j = 1, \dots, N_z$
- neuróny vo výstupnej sú indexované od $i = 1, \dots, N_0$

v skrytej vrstve sú nelineárne a vo vstupe a výstupe je aktivačnou funkciou funkcia identity.

Teda v skrytej vrstve na vstupe zo vstupných neurónov dostaneme

$$ou_j(t) = \sum_{l=1}^M w_{jl}(t)x_l(t) - \theta_j \quad (4.16)$$

kde a po prechode skrytoou vrstvou na jej výstupe dostaneme

$$ou_i(t) = \sum_{j=1}^{N_z} w_{ij}(t)f \sum_{l=1}^M (w_{jl}(t)x_l(t) - \theta_j) - \theta_i \quad (4.17)$$

a tak dalej. Teda je položená otázka, ako daleko je potrebné pokračovať v rozvoji approximácie funkcie v zmysle (4.17). Odpoveď na túto otázku dala matematická analýza celého problému v práci [?], ktorá znala NN s **jednou, resp. max dvoma skrytými vrstvami je schopná approximácie ľubovoľnej funkcie** s dostatočnou presnosťou. Je zrejme, že viac ako 2-vrstvové NN to vedia tiež, ale za cenu vyšej zložitosti NN. Existuje tzv. **Univerzálna approximačná teorema** popísaná tiež v [?], ktorá tvrdí, že stačí jedna skrytá vrstva na approximáciu ľubovoľnej funkcie. Z praktického hľadiska, však nehovorí koľko neurónov má mať táto jediná skrytá vrstva. Ak budeme predpokladať, že priupustíme aj podľa [?] 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 approximovať **dvoma skrytými vrstvami**

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

- **prvá skrytá vrstva** nachádza **lokálne príznaky** approximovanej funkcie
- **druhá skrytá vrstva** spracováva výstup z prvej skrytej vrstvy a jej úlohou je určovať **globálne príznaky** approximovanej 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 topológií 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 ([?])
2. eliminačný algoritmus (tzv. pruning [?])

V prvom prípade ide o postupné budovanie NN, nahradzanie 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 topológií musíme nastavovať parametre učenia, takže ide o dosť náročný proces.

Eliminačný prístup naproti tomu začína s tzv. maximálnou verziou topológie NN a postupnými elimináciami neurónov alebo SV dochádza k optimálnej topológií NN. K tejto forme hľadania optimálnej topológie NN existujú špeciálne prístupy napr. Optimal Brain Damage, Optimal Brain Surgeon⁷ a pod.

4.3.5 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 situáciám je možné uvažovať o nasledovných doporučeniacach:

- 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 lineárnych neurónoch dochádza k saturácií 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 [?] je uvedený empirický interval pre inicializačné hodnoty v tvare

$$\left(\frac{-2.4}{NI_i}, \frac{2.4}{NI_i} \right) \quad (4.18)$$

⁷ tieto nájdete v SNNSv3.3

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 konvergentným stavom NN.

4.3.6 Problém stanovenia veľkosti trénovacej vzorky

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

$$n \geq \frac{32W}{\epsilon} \ln\left(\frac{32m}{\epsilon}\right) \quad (4.19)$$

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

$$n > \frac{W}{\epsilon} \quad (4.20)$$

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

Kapitola 5

Kontrolované učenie na FF NN

5.1 Perceptrón

V tejto časti budeme prezentovať najjednoduchšiu neurónovú sieť, ktorá predstavuje základ pre komplikovanejšie do predné siete. Perceptrón bol navrhnutý Rosenblattom [?] a je určený na **dichotomickú** klasifikáciu, tj. rozdelenie do dvoch tried, pri ktorých sa predpokladá, že triedy sú **lineárne separovateľné** v príkladovom priestore.

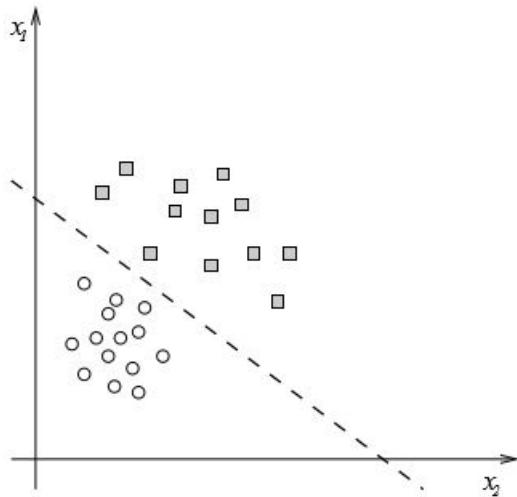
Nech je daná množina vektorov $\mathbf{X} = \{\mathbf{x}^j\}, j = 1, \dots, k, \dots, \infty$ v n -rozmernom priestore. O každom z týchto vektorov vieme, že určite patrí do triedy **CL1** alebo **CL2**. Pod lineárной separovateľnosťou dvoch tried rozumieme situáciu, keď existuje možnosť oddeliť objekty v príkladovom priestore pomocou nadroviny napr: priamka v 2-rozmernom alebo rovina v 3-rozmernom priestore. Príklad lineárnej separovateľnosti v rovine je na obr. 5.1

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

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.



Obr. 5.1: Lineárna separovateľnosť tried v príkladovom priestore

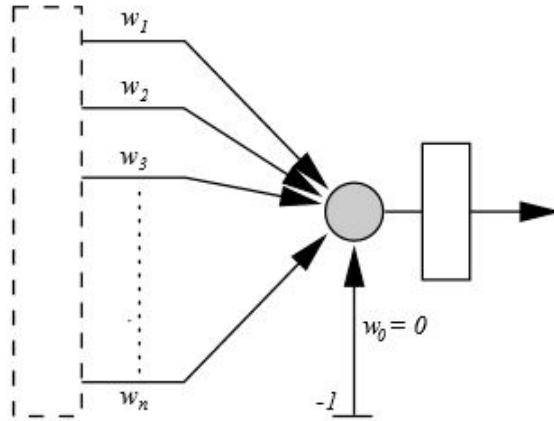
Spojenie medzi senzorovou vrstvou a asociatívou má pevné váhy, teda sa na procese učenia nezúčastňuje. Spojenie medzi asociatívou vrstvou a výstupným neurónom je prepojené synapsami s premenlivými SV. Teda vstup do výstupného neurónu bol daný rovnicou

$$in(t) = \sum_{j=1}^n w_j(t)x_j(t) - \theta \quad (5.1)$$

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

$$ou(t) = \begin{cases} 1 & \text{ak } in(t) \geq 0 \\ 0 & \text{ak } in(t) < 0 \end{cases} \quad (5.2)$$

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



Obr. 5.2: Topológia perceptrónu navrhnutého Rosenblattom

$$\sum_{j=1}^n w_j(t)x_j(t) - \theta = 0 \quad (5.3)$$

Z praktického hľadiska označme vektor $\mathbf{w}(t) = (w_0(t), w_1(t), w_2(t), \dots, w_n(t))$, vektor $\mathbf{x}(t) = (x_0(t), x_1(t), x_2(t), \dots, x_n(t))$ 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 operáciach budeme pre prehľadnosť vyniechať 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.

5.1.1 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:

Prepočladajme, že $(\mathbf{x}(1), d^1), (\mathbf{x}(2), d^2), \dots, (\mathbf{x}(m), d^m)$ je trénujúca vzorka vektorov, na ktorej sa bude perceptrón učiť, $\mathbf{x}(t) \in R^n$, $d^t = 1$, ak $\mathbf{x}(t) \in \mathbf{CL1}$, $d^t = -1$, ak $\mathbf{x}(t) \in \mathbf{CL2}$.

- 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 $\mathbf{ev}(t)$), potom

$$\mathbf{w}(t+1) = \mathbf{w}(t) \quad (5.4)$$

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 (5.4)

- 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 $\mathbf{ev}(t)$ teda $d^t = -1$) potom

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \gamma \mathbf{x}(t) \quad (5.5)$$

2. ak $\mathbf{w}(t)\mathbf{x}(t) < 0$ a $\mathbf{x}(t)$ skutočne patrí do **CL1** (podľa $\mathbf{ev}(t)$ teda $d^t = 1$), tak potom

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma \mathbf{x}(t) \quad (5.6)$$

kde γ je učiaci parameter a môže predstavovať ľubovoľnú kladnú premennú, ktorá je po dobu učenia konštantná alebo sa môže aj meniť teda $\gamma \rightarrow \gamma(t)$. Vyššie uvedená rovnica môže byť zapísaná v tvare

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma (d^t - ou(t))\mathbf{x}(t), \quad t = 1, 2, \dots \quad (5.7)$$

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

$$\mathbf{w}(0) = 0 \quad (5.8)$$

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{z}(t), \quad \text{ak } \mathbf{z}(t)\mathbf{w}(t) \leq 0 \quad (5.9)$$

$$\mathbf{w}(t+1) = \mathbf{w}(t), \quad \text{ak } \mathbf{z}(t)\mathbf{w}(t) > 0 \quad (5.10)$$

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**.

Veta o konvergencii perceptrónu

Predpokladajme, že separujúca nadrovina existuje, tj. triedy sú separovaťné. Cieľom tejto časti je dokázať, že JPR po konečnom počte krokov, tj. ak dostane "k" rôznych vstupov¹ z množiny \mathbf{X}^n , $k < S$, kde S je konečné prirodzené číslo, dosiahne stabilný stav, tj. už bude vedieť klasifikovať hocjaký 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, t.j. platí

$$\mathbf{x}\mathbf{v}(t) > 0$$

pre $t = 1, 2, \dots$ a teda v je aproximaciou spravnej vahy w . Predpokladajme, že nastáva situácia, že v t -tom kroku je nesprávna klasifikácia teda prípad (5.9), t.j.

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{z}(t) \quad (5.11)$$

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

$$\mathbf{w}(t+1) - \phi\mathbf{v} = \mathbf{w}(t) - \phi\mathbf{v} + \mathbf{z}(t) \quad (5.12)$$

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

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 = \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 + 2\mathbf{z}(t)(\mathbf{w}(t) - \phi\mathbf{v}) + \|\mathbf{z}(t)\|^2 \quad (5.13)$$

¹k - omylov

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

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - 2\phi\mathbf{v}\mathbf{z}(t) + \|\mathbf{z}(t)\|^2 \quad (5.14)$$

Zavedieme nasledujúce označenie

$$\alpha = \min_{j=0,\dots,k}(\mathbf{v}\mathbf{z}(j)) \quad (5.15)$$

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

$$\beta = \max_i \|\mathbf{z}(i)\| \quad (5.16)$$

Nerovnosť (5.14) je možné upraviť nasledujúcim spôsobom

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - 2\phi\alpha + \beta^2 \quad (5.17)$$

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

$$\|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(t) - \phi\mathbf{v}\|^2 - \beta^2 \quad (5.18)$$

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

$$0 \leq \|\mathbf{w}(t+1) - \phi\mathbf{v}\|^2 \leq \|\mathbf{w}(0) - \phi\mathbf{v}\|^2 - k\beta^2 \quad (5.19)$$

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

$$k_0 = \|\mathbf{w}(0) - \phi\mathbf{v}\|^2 / \beta^2 \quad (5.20)$$

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

$$k_0 = \phi^2 \|\mathbf{v}\|^2 / \beta^2 \quad (5.21)$$

Na základe (5.21) sme dokázali, že k existuje a je to konečné číslo, a tým môžeme vysloviať nasledovné tvrdenie :

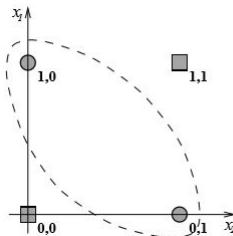
Majme trénovaciu množinu vektorov X , ktoré môžu patriť len do dvoch rôznych tried CL1 a CL2, ktoré sú lineárne separovateľné. Perceptrón po realizovaní "k₀" mylova sa určite dostane do stavu, keď nebude meniť svoje SV, kedy konverguje. To znamená, že bude spoľahlivo klasifikovať vektorov do príslušných tried.

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

Tabuľka 5.1: XOR binárna funkcia

5.1.2 XOR-problém, skrytá vrstva NN

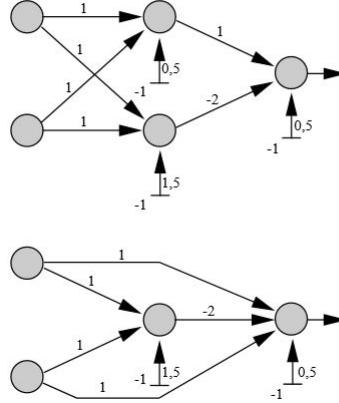
V prípade JPR sme predpokladali, že na vstup budú vstupovať príznaky objektov, ktoré sú v príznakovom priestore lineárne separovateľné. V prípade, že táto podmienka nie je splnená, JPR je nie schopný spoľahlivo realizovať svoju dichotomickú klasifikáciu. Príkladom **nelineárne separovateľnej** funkcie je funkcia XOR.



Obr. 5.3: Grafické znazornenie funkcie XOR

Klasickým JPR nevieme zabezpečiť separáciu výsledkov tejto funkcie v žiadnom prípade, čo je zrejme z obr. 5.3. Je potrebné poznamenať, že stále uvažujeme o **jednoduchej aktivačnej funkcií** napr. typu signum. V prípade iných aktivačných funkcií 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 znova 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



Obr. 5.4: Dve možné riešenia funkcie XOR pomocou skrytej vrstvy

topológie NN, ktoré vedia riešiť XOR-problém. Je potrebné si uvedomiť, že tu ide o aproximáciu binárnej funkcie uvedenej v tabuľke a grafe.

Záverom tejto celej kapitoly je potrebné upozorniť na terminologickú zameňiteľnosť medzi metódou učenia JPR a topológiou typu perceptrón. Siete s rovnakou topológiou môžu mať rôzne algoritmy učenia. 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 topológiu typu perceptrón môže byť realizované principiálne kontrolované učenie rôzneho druhu.

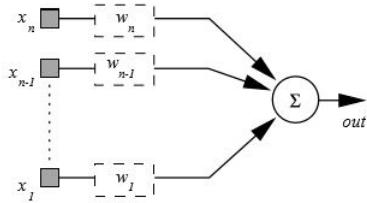
5.1.3 Wienerov filter

Predstavme si n -senzorov umiestnených na rôznych miestach.(viď obr. 5.5) Tieto senzory produkujú signály in_1, \dots, 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

$$ou = \sum_{k=1}^n w_k in_k \quad (5.22)$$

Nech je známy chybový rozdiel

$$e = ev - ou \quad (5.23)$$



Obr. 5.5: Schéma Wienerovho filtra

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 usporiadane dvojice \$(in, ev)\$ patriace do mnohých skupín a v podstate sa tu nejedná o dichotómiu. Topologický sú z pohľadu NN Wienerov filter a perceptron veľmi podobné. Teda môžeme definovať všeobecnú chybovú funkciu v tvare

$$J = 0.5 E(e^2) \quad (5.24)$$

kde \$E(e^2)\$ je stredná hodnota kvadrátov všetkých rozdielov. Základným problémom je teda nájsť také \$w_1, \dots, w_n\$, pri ktorých \$J \rightarrow 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 (5.24) dosadíme (5.23) resp. (5.22), dostaneme nasledovný výraz:

$$J = 0.5 E(ev^2) - E\left(\sum_{j=1}^n w_j in_j ev\right) + 0.5 E\left(\sum_{j=1}^n \sum_{k=1}^n w_j w_k in_j in_k\right) \quad (5.25)$$

Stredná hodnota je lineárna operácia preto môžeme urobiť nasledovné úpravy:

$$J = 0.5 E(ev^2) - \sum_{j=1}^n w_j E(in_j ev) + 0.5 \sum_{j=1}^n \sum_{k=1}^n w_j w_k E(in_j in_k) \quad (5.26)$$

teraz si skúsme popísat jednotlivé funkcie v rovnici (5.26) 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 (5.26) prepísať do tvaru

$$J = 0.5r_{ev} - \sum_{k=1}^n w_k r_{in,ev}(k) + 0.5 \sum_{j=1}^n \sum_{k=1}^n w_j w_k r_{in,in}(jk) \quad (5.27)$$

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

$$\frac{\partial J}{\partial w_l} = -r_{in,ev}(l) + \sum_{j=1}^n w_j r_{in,in}(jl) \quad (5.28)$$

potom logicky pri hľadaní váh počítame s podmienkou $\frac{\partial J}{\partial w_l} = 0$ a dostávame z (5.28)

$$\sum_{j=1}^n w_j r_{in,in}(jl) = r_{in,ev}(l) \quad (5.29)$$

Potom systém l-rovníc (5.29) o n-neznámych $w_j, l = 1, \dots, n$ nazývame Wienerovým systémom rovníc a filter s vypočítanými váhami voláme v teórii signálov **Wienerovým filtrom**.

5.1.4 Metóda najstrmšieho zostupu

K tomu, aby sme vedeli vyriešiť rovnice (5.29), by sme potrebovali vypočítať maticu ($n \times n$) a jej inverziu, čo je dosť náročný výpočet². Existuje aj iná forma výpočtu hľadaných SV a to metódou najstrmšieho zostupu (**steppest descent**). Tento výpočet budeme realizovať iteračným spôsobom v t -iteráciách a budeme vypočítavať zmenu SV, ktorú môžeme vyjadriť nasledovne pre synapsiu i

$$\Delta w_i(t) = -\gamma \frac{\partial J(t)}{\partial w_i(t)} \quad (5.30)$$

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

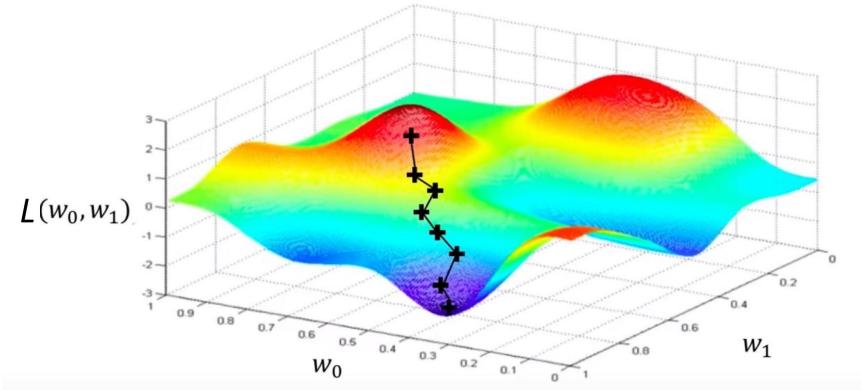
$$w_i(t+1) = w_i(t) + \Delta w_i(t) \quad (5.31)$$

potom z rovníc (5.28) a (5.30) dostaneme z rovnice (5.31) upravený tvar

$$w_i(t+1) = w_i(t) + \gamma(r_{in,ev}(k,t) - \sum_{j=1}^n w_j(t) r_{in,in}(jk,t)) \quad (5.32)$$

²Niekedy 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

kde $i = 1, \dots, n$, n -je počet senzorov. Teda v konečnom dôsledku nájdeme príslušné SV po určitom počte iterácií aj takým spôsobom, avšak výpočtová náročnosť vzorca (5.32) 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, \dots, w_n , dostali by sme hyperplochu, ktorá sa nazýva **povrch chybovej funkcie (error surface)**. Táto zvlnená hyperplocha má svoje **globálne minimum**, ktoré zodpovedá nejakým w_1, \dots, w_n a to sú **práve** SV, ktoré pre filter hľadáme. Pod filtrom budeme rozumieť množinu synaptických váh, ktoré hľadáme.



Obr. 5.6: Chybový priestor pri učení neurónových sietí

5.1.5 Metóda najmenšej kvadratickej chyby

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

$$\bullet \quad \hat{r}_{in,in}(ji, t) = in_j(t)in_i(t) \quad (5.33)$$

$$\bullet \quad \hat{r}_{in,ev}(i, t) = in_i(t)ev(t) \quad (5.34)$$

Tieto odhady $\hat{r}_{in,in;ev,in}$ sa spočítavajú v každej iterácii. Ak toto dosadíme do rovnice (5.32) a budeme uvažovať odhady jednotlivých SV, potom

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \gamma(in_i(t)ev(t) - \sum_{j=1}^n \hat{w}_j in_l(t)in_i(t)) \quad (5.35)$$

čo môžeme upraviť do tvaru - vyberieme $in_i(t)$ mimo zátvorku

$$\hat{w}_i(t+1) = \hat{w}_i(t) + \gamma(ev(t) - \underbrace{\sum_{j=1}^n \hat{w}_j in_l(t)}_{ou(t)} in_i(t)) \quad (5.36)$$

Teda konečný tvar rovnice (5.36) je

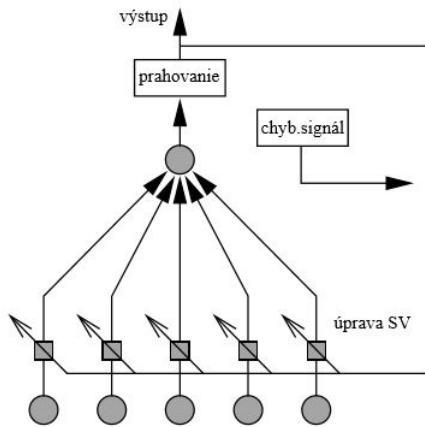
$$\hat{w}_i(t+1) = \hat{w}_i(t) + \gamma \underbrace{(ev(t) - ou(t))}_{e(t)} in_i(t) \quad (5.37)$$

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

- **inicializácia** $\forall j = 1, \dots, n; w_j(t=0) = 0$
- **filtrácia** pre $t = 1, \dots$ 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 $k = 1, \dots, n$
 4. zvýšime (t+1) a návrat do bodu 1

5.1.6 Adaline

Adaline³ bol popísaný Widrowom a Hoffom v roku 1960 (viď obr. ??). Predstavuje najjednoduchšiu NN, topológiou zhodnú s JPR.



Obr. 5.7: Schéma siete Adaline

Principiálny rozdiel je v učiacej funkcií, kým v prípade adaline ide o učenie typu LMS a v prípade JPR ide o iné učenie. Vstup do výstupného neurónu má tvar

$$in(t) = \sum_{j=1}^n w_j(t) in_j(t) \quad (5.38)$$

a potom výstup

$$ou(t) = \begin{cases} 1 & ak \ in(t) \geq \theta \\ -1 & ak \ in(t) < \theta \end{cases} \quad (5.39)$$

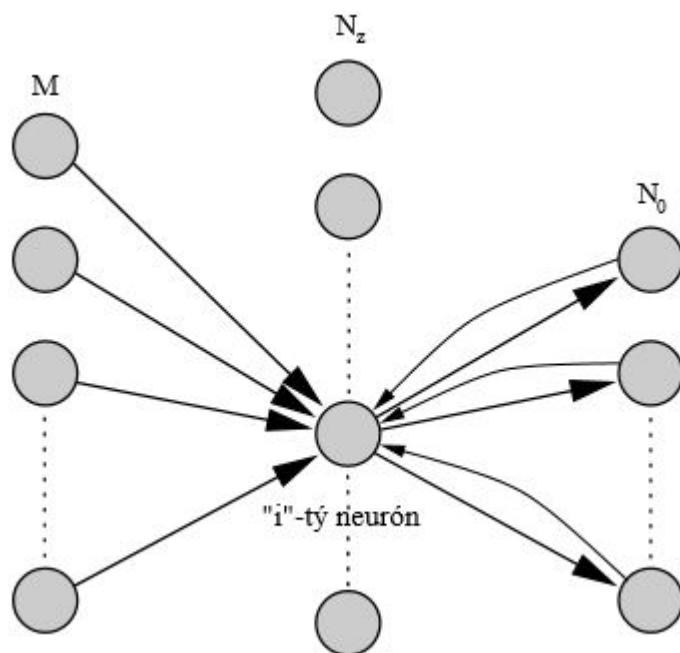
Podľa metódy LMS potom $e(t)$ (viď vzorec 5.23) môže nadobúdať hodnoty $\{-2, 0, 2\}$ čo je zrejmé z možnosti výstupov a možnými rozdielmi medzi nimi.

³Adaptive linear element

Kapitola 6

Delta pravidlo

Delta pravidlo (ďalej DP) predstavuje veľmi dôležitý postup pri výpočte zmeny SV ($\Delta w_{ij}(t)$) a je typu **LMS**. Majme neurónovú sieť s nasledovnou topológiou :



Obr. 6.1: Zobrazenie toku chybového signálu z výstupu pre i -ty neurón

DP rozširuje LMS na prípad ne-McCullochových neurónov¹. Teda pre jednoduchú jednovrstvovú sieť s M vstupnými neurónmi a jedným výstupným neurónom "i" dostaneme

$$x_i = f(in_i) = in_i = \sum_{j=1}^M x_j w_{ij}(t) + \theta_i \quad (6.1)$$

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 $x_i = ou_i$

$$J(t) = \sum_{i=1}^{N_0} J^i(t) = 0.5 \sum_{i=1}^{N_0} (ev_i(t) - x_i(t))^2 \quad (6.2)$$

kde $ev(t)$ je očakávaná a $x_i(t)$ vypočítaná hodnota na i -tom výstupe 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

$$\Delta w_{ij}(t) = -\gamma \frac{\partial J(t)}{\partial w_{ij}(t)} \quad (6.3)$$

kde γ je učiaci pomer². Pravú stranu v (6.3) je možné upraviť

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = \frac{\partial J(t)}{\partial x_i(t)} \frac{\partial x_i(t)}{\partial w_{ij}(t)} \quad (6.4)$$

Prvý člen pravej strany sa ďalej rovná

$$\frac{\partial J(t)}{\partial x_i(t)} = -(ev_i(t) - x_i(t)) \quad (6.5)$$

a druhý člen na základe (6.1) sa rovná

$$\frac{\partial x_i(t)}{\partial w_{ij}(t)} = x_j(t) \quad (6.6)$$

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

$$\Delta w_{ij}(t) = \gamma \delta(t) x_j(t) \quad (6.7)$$

¹ niekedy ich nazývame kontinuálne neuróny

² alebo koeficient proporcionality

Takto vypočítaná zmena SV podľa **delta pravidla - d'alej DP** dala základ ďalším modifikáciám delta pravidla, čo prispelo k jeho rozšíreniu a aplikácii pri učení NN. Signál $\delta(t)$ nazývame chybovým signálom.

6.1 Metóda spätného šírenia chyby

V predošej časti matematicky odvodené DP vlastne predstavuje základ **učenia so spätným šírením chyby**³ a umožňuje použitie v podstate ľuboľnej aktivačnej funkcie f aj nelineárneho typu, ktorá splňuje podmienku diferencovateľnosti, t.j. platí

$$x = f(in) \neq in \quad (6.8)$$

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

$$x_i(t) = f(in_i(t)), \quad (6.9)$$

kde

$$in_i(t) = \sum_{j=1}^M w_{ij}(t)x_j(t) + \theta_i. \quad (6.10)$$

Z predchádzajúceho DP vieme, že

$$\Delta w_{ij}(t) = -\gamma \frac{\partial J(t)}{\partial w_{ij}(t)}. \quad (6.11)$$

$J(t)$ má tvar

$$J(t) = 0.5 \sum_{j=1}^{N_0} (ev_i(t) - x_i(t))^2, \quad (6.12)$$

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

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = \frac{\partial J(t)}{\partial in_i(t)} \frac{\partial in_i(t)}{\partial w_{ij}(t)} \quad (6.13)$$

³Back-propagation of Error

Označme

$$\frac{\partial J(t)}{\partial in_i(t)} = -\delta_i(t) \quad (6.14)$$

a

$$\frac{\partial in_i(t)}{\partial w_{ij}(t)} = x_j(t), \quad (6.15)$$

potom dostaneme obvykly zápis výpočtu zmeny SV v tvaru

$$\Delta w_{ij}(t) = \gamma \delta_i(t) x_j(t) \quad (6.16)$$

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

Pre príslušné $\delta_i(t)$ môžeme ďalej písť na základe (6.14)

$$\delta_i(t) = -\frac{\partial J(t)}{\partial in_i(t)} = -\frac{\partial J(t)}{\partial x_i(t)} \frac{\partial x_i(t)}{\partial in_i(t)} \quad (6.17)$$

Najprv vyriešme druhý člen pravej strany (6.22). Vzhľadom na nelineárny neurón je zrejmé, že môžeme napísť pomocou (12.30), že

$$\frac{\partial x_i(t)}{\partial in_i(t)} = f'(in_i(t)) \quad (6.18)$$

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

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

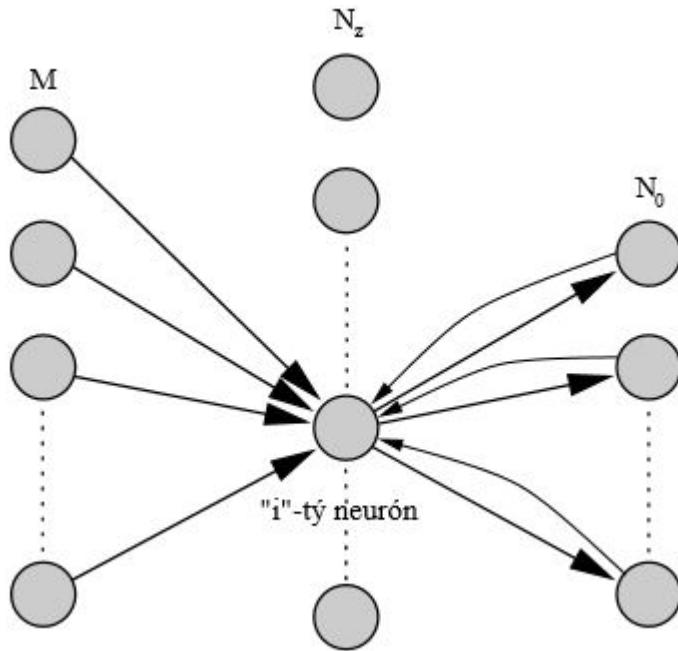
$$\frac{\partial J(t)}{\partial x_i(t)} = -(ev_i(t) - x_i(t)) \quad (6.19)$$

a tým máme výpočet $\delta_i(t)$ pre tento prípad vyriešený pomocou (6.18) a (6.19) v tvaru

$$\delta_i(t) = (ev_i(t) - x_i(t))f'(in_i(t)) \quad (6.20)$$

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

$$\frac{\partial J(t)}{\partial x_i(t)} = \sum_{h=1}^{N_0} \frac{\partial J(t)}{\partial in_h(t)} \frac{\partial in_h(t)}{\partial x_i(t)} \quad (6.21)$$

Obr. 6.2: Zobrazenie toku chybového signálu z výstupu pre i -ty neurón

kde N_0 je počet neurónov vo výstupnej vrstve, resp. napravo od i , čo je znázornené na Obr. 6.2. Z matematického hľadiska pri výpočte derivácie chybovej funkcie J , ktorá popisuje celkovú chybu na **výstupnej vrstve**, podľa $x_i(t)$ ⁴, je nutné vyjadriť J ako funkciu $x_i(t)$. Preto rovnica (6.21) má takýto tvar. Súčasne prvý člen pravej strany je jasný z rovnice (6.22) a teda platí, že

$$\frac{\partial J(t)}{\partial in_i(t)} = -\frac{\partial J(t)}{\partial x_i(t)} \frac{\partial x_i(t)}{\partial in_i(t)} = \delta_i(t) \quad (6.22)$$

Tu je potrebné poznamenať, že $x_h(t)$ v rovnici (6.21) je z inej vrstvy ako $x_i(t)$ v rovnici (6.21), čo sa týka druhého člena rovnice (6.21), tam samotný člen $in_h(t)$ predstavuje vstup do výstupného neurónu h a môžeme ho nahradieť nasledovne

$$in_h(t) = \sum_{l=1}^{N_z} w_{hl}(t) x_l(t) \quad (6.23)$$

⁴to je stav neurónu vo vrstve, ktorá nie je výstupná

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

$$\frac{\partial in_h(t)}{\partial x_i(t)} = w_{hi}(t) \quad (6.24)$$

teda v konečnom dôsledku

$$\frac{\partial J(t)}{\partial x_i(t)} = - \sum_{h=1}^{N_0} \delta_h(t) w_{hi}(t) \quad (6.25)$$

a konečne hľadaný koeficient $\delta_i(t)$ bude mať tvar na základe (6.18) a (6.25)

$$\delta_i(t) = f'(in_i(t)) \sum_{h=1}^{N_0} \delta_h(t) w_{hi}(t) \quad (6.26)$$

Je potrebné dobre si všimnúť **rekurzívnosť** tohto vzťahu. Ide o výpočet koeficientu $\delta_i(t)$ neurónu i , ktorý **nie** je výstupným 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. 6.2).

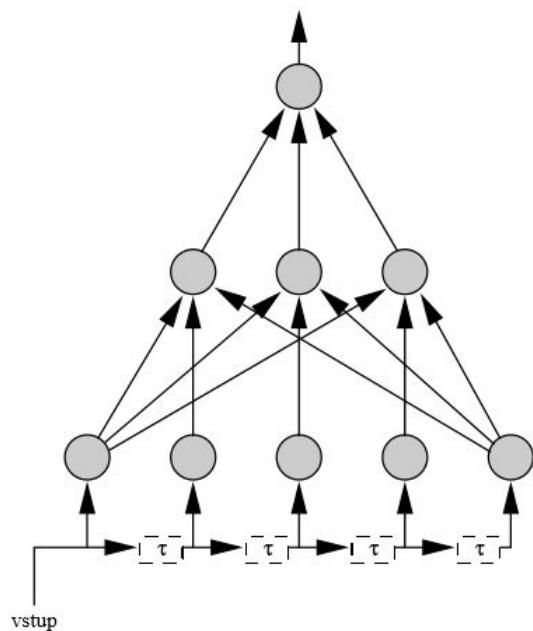
Existuje modifikácia vzťahu (12.15) a to v tvare

$$\delta_i(t) = (f'(in_i(t)) + c) \sum_{h=1}^{N_0} \delta_h(t) w_{hi}(t) \quad (6.27)$$

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

6.1.1 Time-delay na FF NN

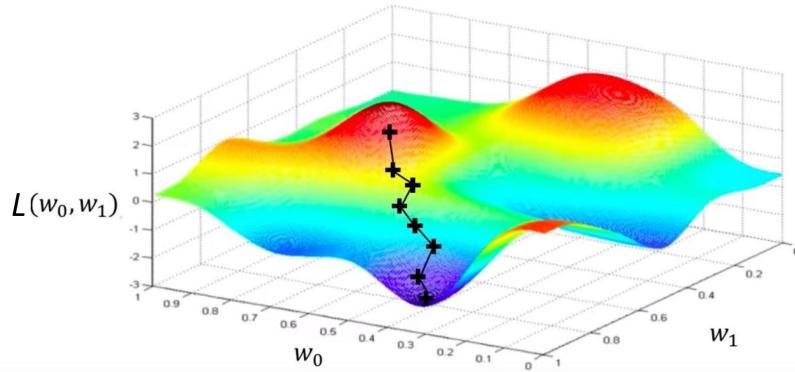
Tento prístup predstavuje klasickú metódu BP s upraveným vstupom. Topológia TD FF NN je znázornená na obrázku 6.3, kde (z^{-1}) predstavujú časové oneskorovacie členy. Táto topológia je vhodná pre vstup určitej série meraní, ktoré budú do NN vstupovať ako celok. V podstate ide o vytvorenie klzavého okna nad vstupmi, ktorého šírka predstavuje samotné celkové časové oneskorenie. Matematicky je TD BP málo odlišné od klasickej formy BP. Aplikácie tohto prístupu sú veľmi časté v rôznych oblastiach napr. v ekonomike, riadení, spracovaní dynamických obrazov a pod.



Obr. 6.3: NN s vstupom časového signálu

6.1.2 Spôsoby urýchlenia konvergencie BP

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



Obr. 6.4: ilustračný obrázok 2+1 rozmerného chybového priestoru a výpočtu gradientu na chybovom povrhu s cieľom hľadania globálneho minima chyby L

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

6.1.3 BP-momentum

Použitie BP s momentom vychádza z poznatkov o chovaní sa siete. Ak vo vzťahu (6.7) pre SV, ktoré prepájajú neuróny “ j ” a “ i ” (smeruje do “ i ”), tj.

$$\Delta w_{ij}(t) = \gamma \delta_i(t) x_j(t) \quad (6.28)$$

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

$$\Delta w_{ij}(t) = \gamma \delta_i(t) x_i(t) + \alpha \Delta w_{ij}(t - 1) \quad (6.29)$$

Novopridanú časť $\alpha \Delta w_{ij}(t - 1)$ nazývame momentový výraz (**momentum term**) a teda dochádza k zmene vzorca (6.28) a 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.

6.1.4 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

6.1.5 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 iteráciách, 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 iteráciách, 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 iterácii⁵ 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:

⁵ pojem iterácia môžeme chápať tiež ako čas resp. čas zmeny

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

$$\mathcal{J}(t) = 0.5 \sum_{j=1}^{N_0} (ev_j(t) - x_j(t))^2 = 0.5 \sum_{j=1}^{N_0} e_j^2(t) \quad (6.30)$$

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

$$\Delta w_{ij}(t-1) = -\gamma \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \quad (6.31)$$

Túto zmenu SV použijeme vo vzťahu

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t-1) \quad (6.32)$$

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

$$w_{ij}(t) = w_{ij}(t-1) - \gamma_{ij}(t) \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \quad (6.33)$$

Súčasne platí, že

$$\gamma_{ij}(t) = \gamma_{ij}(t-1) + \Delta \gamma_{ij}(t) \quad (6.34)$$

Je potrebné si dobre uvedomiť indexy, ak porovnáme rovnice (6.32) a (6.34). Súčasne môžeme vstup do neurónu i"vyjadriť

$$in_i(t) = \sum_{j=0}^M w_{ij}(t)x_j(t) + \theta_i \quad (6.35)$$

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 :

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \underbrace{\frac{\partial \mathcal{J}(t)}{\partial x_i(t)}}_A \underbrace{\frac{\partial x_i(t)}{\partial in_i(t)}}_B \underbrace{\frac{\partial in_i(t)}{\partial \gamma_{ij}(t)}}_C \quad (6.36)$$

Jednotlivé členy A, B, C vo vzťahu (6.36) môžeme vypočítať nasledovne

- **člen A** je zrejmý z rovnice (6.30) pre chybovú funkciu

$$\frac{\partial \mathcal{J}(t)}{\partial x_i(t)} = -(ev_i - x_i(t)) = -e_i(t) \quad (6.37)$$

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

$$\frac{\partial x_i(t)}{\partial in_i(t)} = f'(in_i(t)) \quad (6.38)$$

- **člen C** ak do rovnice (6.35) dosadíme zrejmú rovnicu

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t-1) \quad (6.39)$$

dostaneme výraz pre $in_i(t)$

$$in_i(t) = \sum_{j=1}^M \underbrace{(w_{ij}(t-1) - \gamma_{ij}(t) \frac{\partial J(t-1)}{\partial w_{ij}(t-1)})}_{w_{ij}(t)} x_j(t) + \theta_i \quad (6.40)$$

ked' predošlú rovnicu zderivujeme podľa $\gamma_{ij}(t)$, dostaneme

$$\frac{\partial in_i(t)}{\partial \gamma_{ij}(t)} = -x_j(t) \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \quad (6.41)$$

teda v konečnom dôsledku dostaneme

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \underbrace{-e_i(t)f'(in_i(t))}_{A+B} \underbrace{-x_j(t)\frac{\partial J(t-1)}{\partial w_{ij}(t-1)}}_C \quad (6.42)$$

Teraz pomocou (6.16) môžeme vyjadriť

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = \delta_i(t)x_j(t) \quad (6.43)$$

potom za $\delta_i(t)$ môžeme dosadiť z (6.19) a skutočne dostaneme za $\frac{\partial J(t)}{\partial w_{ij}(t)}$ výrazy $A + B$ a teda

$$\frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = -\frac{\partial J(t)}{\partial w_{ij}(t)} \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \quad (6.44)$$

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

$$\Delta \gamma_{ij}(t) = -\varphi \frac{\partial \mathcal{J}(t)}{\partial \gamma_{ij}(t)} = \varphi \frac{\partial J(t)}{\partial w_{ij}(t)} \frac{\partial J(t-1)}{\partial w_{ij}(t-1)} \quad (6.45)$$

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 [?] v tvare pre zmenu $\gamma_{ij}(t)$ v tvare

$$\Delta \gamma_{ij}(t) = \begin{cases} \kappa & ak S_{ij}(t-1)D_{ij}(t) > 0 \\ -\beta \gamma_{ij}(t) & ak S_{ij}(t-1)D_{ij}(t) < 0 \\ 0 & inak \end{cases} \quad (6.46)$$

kde S a D definujeme ako

$$S_{ij}(t-1) = (1-\zeta)D_{ij}(t-1) + \zeta S_{ij}(t-2) \quad (6.47)$$

a

$$D_{ij}(t) = \frac{\partial J(t)}{\partial w_{ij}(t)} \quad (6.48)$$

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 (6.46), 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 (6.46) 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

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = - \sum_{k=1}^B \delta_i^k(t) x_j^k(t) = D_{ij}(t) \quad (6.49)$$

a teda zmena SV potom bude mať tvar

$$\Delta w_{ij}(t) = \alpha \Delta w_{ij}(t-1) + \gamma_{ij}(t) \sum_{k=1}^B \delta_i^k(t) x_j^k(t) \quad (6.50)$$

a nová hodnota SV bude vypočítana podľa obvyklého vzťahu

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) \quad (6.51)$$

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 (6.45) 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 stáhuje 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 [?], 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 [?]. Majme množinu U, A a množinu L . Nech $\mathcal{A} \in \mathcal{U}$. Potom definujme funkciu μ a to nasledovne

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

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 funkcií $J(t)$, chceme na základe tej 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 [?] 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 **d'aleko** 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 majú definované nasledovné lingvistické premenné:

- **CE(t)** : veľmi malé, malé, stredné, veľké, veľmi veľké ;
- **QCE(t)**: malé, stredné, veľké, veľmi veľké ;
- **$\xi(t)$** : malé, stredné, veľké ;

Samotná rozhodovacie tabuľka vyzerá nasledovne:

Výsledné hodnoty $\xi(t)$ sú v tabuľke, t.j. ak $CE(t) = \text{stredn}$ a $QCE(t) = \text{vemi vek}$, potom hľadané $\xi(t)$ bude **veľké**. Potom podľa grafu určíme presnú hodnotu $\xi(t)$.

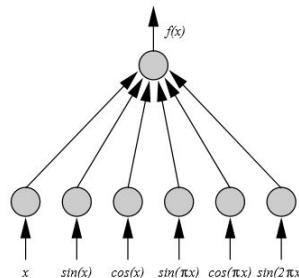
	QCE			
CE	malé	stredné	veľké	veľmi veľké
veľmi malé	malé	malé	stredné	veľké
malé	malé	stredné	stredné	veľké
stredné	malé	stredné	stredné	veľké
veľké	malé	stredné	veľké	veľké
veľmi veľké	malé	stredné	veľké	veľké

6.2 Funkcionálne linky v NN

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

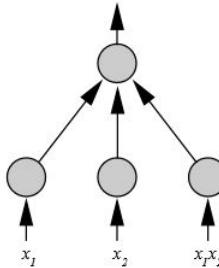
$$in^2, \sin(in), \cos(in), \sin(\pi in), \cos(\pi in)$$

a podobne vid' obrázok 6.5. NN s funkcionálnymi linkami sa taktiež nazývajú **NN vyšších rádov**. V prípade použitia takého prvku je možné do určitej miery zjednodušiť NN. Teda napr. známy XOR problém sa dá pomocou funkcionálnej linky vyriešiť bez skrytej vrstvy (viď obr. 6.6).



Obr. 6.5: Príklad NN vyššieho rádu

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



Obr. 6.6: XOR problém riešený pomocou funkcionálnej linky

6.3 Metóda Radial Basis function (RBF)

Princíp Radial Basis Function (ďalej RBF) je odvodený od teórie aproximácie funkcií. Ak máme N usporiadaných dvojíc (\vec{x}_i, y_i) ($\vec{x}_i \in R^n, y_i \in R$), budeme hľadať funkciu f v tvare :

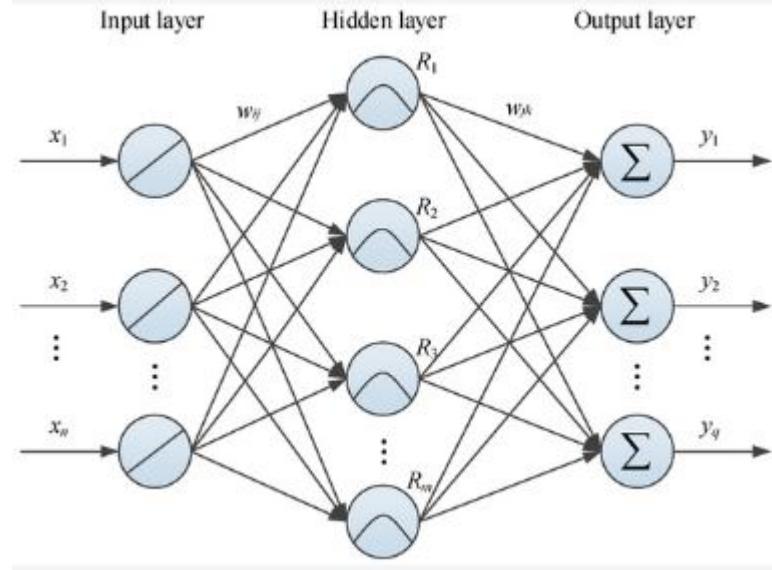
$$f(\vec{x}) = \sum_{i=1}^K c_i h(|\vec{x} - \vec{t}_i|) \quad (6.52)$$

h je funkcia *radial basis* a \vec{t}_i predstavuje K stredových vektorov, ktoré treba určiť. Koeficienty c_i sú tiež neznáme a treba ich vypočítať. \vec{x}_i a \vec{t}_i sú prvkami n -rozmerného vektorového priestoru. h je aplikovaná na euklidovskú vzdialenosť každého stredového vektora \vec{t}_i a daného argumentu \vec{x} . Obvykle sa používa funkcia, ktorá má svoje maximum pri vzdialosti 0, najčastejšie gaussovská funkcia. V tomto prípade, pre hodnoty \vec{x} , ktoré sú rovné stredovému vektoru \vec{t} , nadobúda h hodnotu 1.0 kým pre väčšie vzdialosti hodnotu 0.

Funkcia f má byť approximáciou daných N usporiadaných dvojíc (\vec{x}_i, y_i) a preto musí minimalizovať nasledujúcu chybovú funkciu H :

$$H[f] = \sum_{i=1}^N (y_i - f(\vec{x}_i))^2 + \lambda ||Pf||^2 \quad (6.53)$$

Prvá časť definície H (suma) je podmienkou, ktorá minimalizuje celkovú chybu approximácie, t.j. núti f approximovať N daných bodov. Druhá časť



Obr. 6.7: Príklad topológie RBF sietí

H ($\|Pf\|^2$) je stabilizátor, ktorý núti f nadobudnúť takú plynulosť, ako je možné. Parameter λ určuje mieru vplyvu stabilizátora.

Za určitých podmienok je možné ukázať, že súbor koeficientov c_i možno vypočítať tak, aby H bola minimálna. Tento výpočet zavisí na stredových vektoroch, ktoré musia byť dopredu zvolené. Požitím nasledujúcich vektorov a matíc :

$$\vec{c} = (c_1, \dots, c_k)^T, \vec{y} = (y_1, \dots, y_N)^T \quad (6.54)$$

$$G = \begin{pmatrix} h(|\vec{x}_1 - \vec{t}_1|) & \cdots & h(|\vec{x}_1 - \vec{t}_k|) \\ \vdots & \ddots & \vdots \\ h(|\vec{x}_N - \vec{t}_1|) & \cdots & h(|\vec{x}_N - \vec{t}_K|) \end{pmatrix} \quad (6.55)$$

$$G_0 = \begin{pmatrix} h(|\vec{t}_1 - \vec{t}_1|) & \cdots & h(|\vec{t}_1 - \vec{t}_k|) \\ \vdots & \ddots & \vdots \\ h(|\vec{t}_K - \vec{t}_1|) & \cdots & h(|\vec{t}_K - \vec{t}_K|) \end{pmatrix} \quad (6.56)$$

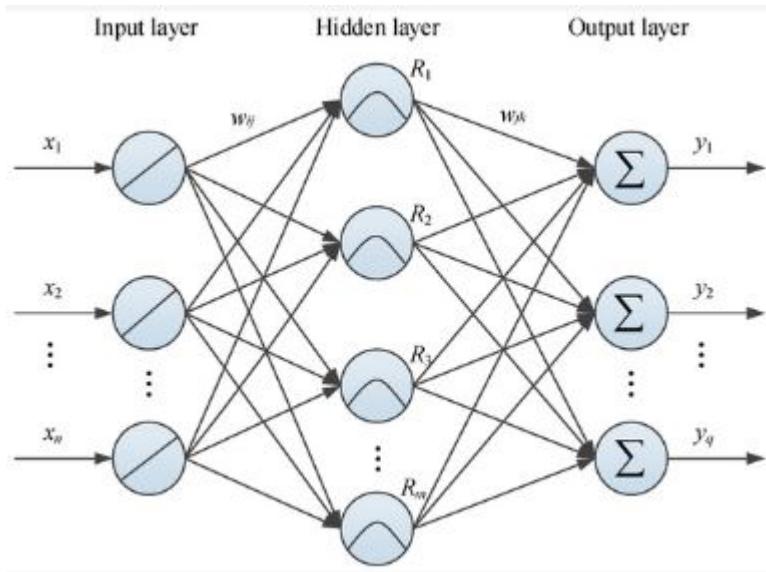
možno súbor neznámych parametrov c_i vypočítať pomocou vzorca:

$$\vec{c}_i = (G^t \cdot G + \lambda G_0)^{-1} \cdot G^T \cdot \vec{y} \quad (6.57)$$

Nastavením parametra λ na 0 tento vzorec sa stane identický k výpočtu Moore Penroseovej inverznej matice, ktorá dáva najlepšie riešenie nedeterminovanej sústavy lineárnych rovníc. V tomto prípade je lineárna sústava presne to, čo vychádza z podmienok presnej interpolácie daného problému:

$$f(\vec{x}_j) = \sum_{i=1}^K c_i h(|\vec{x}_j - \vec{t}_i|) = y_j \quad , \quad j = 1, \dots, N \quad (6.58)$$

Metóda RBF sa dá ľahko interpretovať pomocou doprednej trojvrstvovej NS. Vstupná vrstva pozostáva z n neurónov, ktoré reprezentujú prvky vektora \vec{x} .



Obr. 6.8: Príklad topológie RBF sietí

K komponentov sumy v definícii f je reprezentovaných neurónmi skrytej vrstvy. Linky medzi vstupnou a skrytou vrstvou predstavujú prvky vektora \vec{t}_i . Neuróny skrytej vrstvy počítajú euklidovskú vzdialenosť medzi vstupným vektorom a linkami vedúcimi do daného neurónu. Aktivácia neurónou

skrytej vrstvy sa vypočíta aplikovaním funkcie h na euklidovskú vzdialenosť.

Do výstupného neurónu vstupujú linky zo všetkých skrytých neurónov. Linky vedúce do výstupných neurónov sú predstaviteľmi koeficientov c_i . Aktivácia výstupných neurónov je určená váženou sumou ich vstupov.

Vyššie popísanú architektúru NS, ktorá realizuje aproximáciu použitím RBF, možno ľahko rozšíriť o niekoľko užitočných prvkov. Možno použiť viac ako jeden výstupný neurón, čo umožní aproximáciu niekoľkých funkcií f okolo rovnakej množiny stredových neurónov \vec{t}_i . Aktiváciu výstupných neurónov možno vypočítať použitím nelineárnej invertovateľnej funkcie σ , alebo sigmoidy. Bias výstupných neurónov a linky medzi vstupnou a výstupnou vrstvou (Shortcut connection) možno použiť na zlepšenie kvality aproximácie. Vo všeobecnosti je NS schopná nasledujúcej množiny aproximácií:

$$o_k(\vec{x}) = \sigma \left(\sum_{j=1}^K c_{j,k} h(|\vec{x} - \vec{t}_j|, p_j) + \sum_{i=1}^n d_{i,k} + b_k \right) = \sigma(f_k(\vec{x})) , \quad k = 1, \dots, m \quad (6.59)$$

Tento vzorec opisuje správanie plne prepojenej (Full connected) doprednej NS s n vstupmi, K skrytými a m výstupnými neurónmi. $o_k(\vec{x})$ je aktivácia výstupného neurónu k pri vstupe $\vec{x} = x_1, x_2, \dots, x_n$ do výstupných neurónov. Koeficienty $c_{j,k}$ reprezentujú linky medzi skrytou a výstupnou vrstvou. Shortcut connections predstavujú $d_{i,k}$. b_k je bias výstupných neurónov a p_j je bias neurónov skrytej vrstvy, ktorý determinuje presné charakteristiky funkcie h . Aktivačnú funkciu výstupných neurónov predstavuje σ .

Veľkou výhodou RBF je možnosť priameho výpočtu koeficientov $c_{i,k}$ a biasu b_k . Tento výpočet vyžaduje vhodný výber stredových vektorov \vec{t}_j . Kvôli nedostatku informácií o stredových vektoroch je potrebné urobiť niekoľko učiacich cyklov potom, ako boli vypočítané váhy. Ani váhy liniek medzi vstupnou a výstupnou vrstvou nemôžu byť vypočítané priamo. Musí byť aplikovaná špeciálna učiaca procedúra pre NS používajúce RBF. Odporúča sa používať rôzne učiace parametre pre jednotlivé skupiny učených parametrov. Nasledujúci súbor vzorcov obsahuje všetky informácie potrebné pre učiacu procedúru:

$$J = \sum_{k=1}^m \left(\sum_{i=1}^N (y_{i,k} - o_k(\vec{x}_i))^2 \right) \quad (6.60)$$

$$\Delta \vec{t}_j = -\eta_1 \frac{\partial J}{\partial \vec{t}_j} \quad (6.61)$$

$$\Delta p_j = -\eta_1 \frac{\partial J}{\partial p_j} \quad (6.62)$$

$$\Delta c_{j,k} = -\eta_3 \frac{\partial J}{\partial c_{j,k}} \quad (6.63)$$

$$\Delta d_{i,k} = -\eta_3 \frac{\partial J}{\partial d_{i,k}} \quad (6.64)$$

$$\Delta b_k = -\eta_3 \frac{\partial J}{\partial b_k} \quad (6.65)$$

často je vhodné používať *momentum*. Tento výraz zvyšuje učiaci parameter na plynulom chybovom povrchu a znižuje ho pri členitom chybovom povrchu. Nasledujúci vzorec opisuje účinok *momenta* na učenie všeobecného parametra w (napríklad synaptická váha) v závislosti od dodatočného parametra μ . Δw_{t+1} je zmena w počas časového kroku $t + 1$ a Δw_t je zmena v kroku t :

$$\Delta w_{t+1} = -\eta \frac{\partial J}{\partial w} + \mu \Delta w_t \quad (6.66)$$

Iná užitočná úprava učiacej procedúry je definícia maximálnej povolenej chyby na výstupnom neuróne. Týmto možno predchádzať preučeniu siete. Ak je chyba menšia ako táto hodnota považuje sa za 0, a v zodpovedajúcich linkách nedochádza k zmene.

6.4 Kaskádna korelácia (CC)

6.4.1 Matematické základy

Učenie sa pokúša minimalizovať sumu kvadratických odchýlok na výstupných neurónoch E :

$$J = \sum_p \frac{1}{2} \sum_o (y_{po} - t_{po})^2 \quad (6.67)$$

kde t_{po} je požadovaný a y_{po} je získaný výstup výstupného neurónu o pre vzorku p . Chyba J je minimalizovaná metódou najstrmšieho zostupu používajúcou:

$$e_{po} = (y_{po} - t_{po}) f'_p(\text{net}_o) \frac{\partial J}{\partial w_{io}} = \sum_p e_{po} I_{ip} \quad (6.68)$$

kde f'_p je deriváciou aktivačnej funkcie výstupného neurónu o a I_{ip} je hodnota vstupného alebo skrytého neurónu i pre vzorku p . w_{io} je váhou linky medzi neurónom i a výstupným neurónom o .

Po fáze učenia sú kandidátske neuróny adaptované tak, že je korelácia C medzi hodnotou y_{po} kandidátskeho neurónu a výslednou chybou e_{po} výstupného neurónu maximálna. Korelácia podľa Fahlmana je daná:

$$\begin{aligned} C &= \sum_o \left| \sum_p (y_{po} - \vec{y}_o)(e_{po} - \vec{e}_o) \right| \\ &= \sum_o \left| \sum_p y_{po} e_{po} - \vec{e}_o \sum_p y_{po} \right| \\ &= \sum_o \left| \sum_p y_{po} (e_{po} - \vec{e}_o) \right| \end{aligned} \quad (6.69)$$

kde \vec{y}_o je priemerná aktivácia kandidátskeho neurónu a \vec{e}_o je priemerná chyba výstupného neurónu za všetky vzorky p . Maximalizácia C metódou najstrmšieho zostupu používa:

$$\delta_p = \sum_o \sigma(e_{po} - \vec{e}_j) f'_p \quad (6.70)$$

$$\frac{\partial C}{\partial w_i} = \sum_p \delta_p I_{pi} \quad (6.71)$$

kde σ_o je označenie korelácie medzi výstupom kandidátskeho neurónu a konečnej chyby na výstupe o .

6.4.2 Algoritmus štandardnej CC

CC kombinuje dve myšlienky. Prvá je **kaskádna architektúra**, pri ktorej sa **neuróny do skrytej vrstvy pridávajú postupne** po jednom. Druhou je učiaci algoritmus, ktorý vytvára a inštaluje nové neuróny do skrytej vrstvy. Pre každý nový neurón skúša maximalizovať koreláciu medzi výstupom nového neurónu a výsledným chybovým signálom siete.

Algoritmus je vykonávaný nasledovne:

- 1.** CC začína s **minimálnou sieťou** pozostávajúcou len zo vstupnej a výstupnej vrstvy. Medzi týmito vrstvami je plné spojenie (fullconnection).
- 2.** Trénuje bežným učiacim algoritmom všetky linky končiace na výstupných neurónoch pokiaľ chyba siete viac neklesá.
- 3.** Vytvorí takzvané **kandidátske neuróny**. Každý kandidátsky neurón je spojený linkou so všetkými neurónmi vstupnej vrstvy a so všetkými existujúcimi neurónmi skrytej vrstvy. Medzi kandidátskymi neurónmi a neurónmi výstupnej vrstvy nie je spojenie.
- 4.** Skúsi **maximalizovať koreláciu** medzi aktiváciou kandidátskych neurónov a výslednou chybou siete tréovaním všetkých liniek vedúcich do kandidátskych neurónov. Učenie prebieha pomocou bežného učiaceho algoritmu. **Učenie sa končí ak sa korelačné koeficienty viac nezlepšujú.**
- 5.** Vyberie kandidátske neuróny s **maximálnou koreláciou**, zmrazí vstupné linky a pridá ich do siete. Aby sa kandidátsky neurón zmenil na neurón skrytej vrstvy vytvorí linky medzi ním a neurónmi výstupnej vrstvy. Skok na bod **2.**

Algoritmus sa opakuje dovtedy pokým celková chyba siete neklesne pod danú hodnotu.

6.4.3 Algoritmus CC s prunningom (PCC)

Cieľom kaskádnej korelácie s prunningom (PCC) je **minimalizovať oča-**

kávanú chybu na testovacej množine namiesto minimalizácie chyby na trénovacej množine. PCC sa snaží určiť optimálny počet neurónov skrytej vrstvy a odstrániť nepotrebné linky potom, ako je zavedený nový neurón. V SNNS sú implementované výberové kritériá len pre lineárne modely.

Algoritmus pracuje nasledovne:

- 1.** Učia sa linky smerujúce do výstupnej vrstvy.
- 2.** Spočíta sa výberové kritérium.
- 3.** Učia sa kandidáti.
- 4.** Vsunie sa nový neurón.
- 5.** Spočíta sa výberové kritérium.
- 6.** Váha každej linky novo vsunutého neurónu sa nastaví na 0.0 a spočíta sa výberové kritérium. Ak existujú linky, ktorých odstránenie by znížilo výberové kritérium, **odstráni sa linka**, ktorá znižuje výberové kritérium najviac. Pokračuje sa krokom **5.**, pokiaľ ďalšie odstránenie nezvýši výberové kritérium.
- 7.** Spočíta sa výberové kritérium. Ak je jeho hodnota vyššia, ako hodnota spočítaná pred zaradením nového neurónu, je užívateľ upozornený, že NS sa stáva príliš veľká. Skok na bod **1.**

Kapitola 7

Nekontrolované učenie na FF NN

V prípade nekontrolovaného učenia na dopredných sietiach my vlastne poskytujeme NN **len vstupy** a očakávame, že samotná NN s týmto vstupmi niečo urobí. V podstate nevieme výsledok spracovania, resp. nevieme ho presne. Aj v tomto prípade pojednávame o učení, ktoré je spojené so zmenou SV v NN. Samotné učenie sa skončí, ak sa NN dostane do stavu **GS** (nie konvergencie¹) 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 nekontrolovaným učením sú:

- zhluková analýza
- kvantovanie vektorov, zhustovanie dát
- zníženie dimenzie² v príznakovom priestore

Pripomeňme si niektoré princípy nekontrolovaného učenia:

1. **Modifikácia SV vedie ku seba-zosilneniu.** Tu je vhodné si pomenúť princíp tzv. rozšíreného Hebbovho učenia spomenutý v 2.

¹o konvergencii nemá význam hovoríť pri nekontrolovanom type učenia

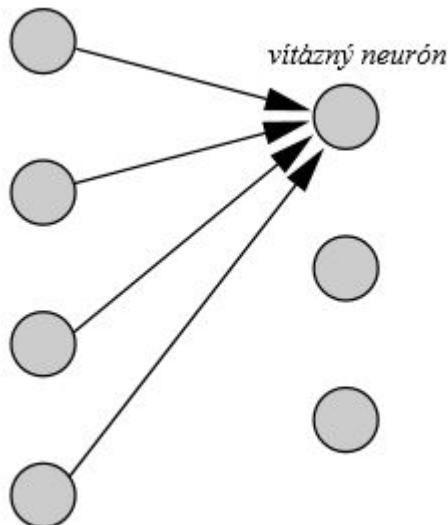
²metóda hlavných komponentov

kapitole vo vzťahu (4.7). 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, keď sa upravujú **iba tie SV**, ktoré smerujú k **vítaznému neurónu**.
3. Pri konkurenčnom učení sa po určitom čase prejavujú prvky **kooperatívneho správania sa**. Tento fakt je založený na poznaní, že ak niejaká SV je výrazne silná, tak potom sa postupne zosilňuje aj jej okolie v zmysle Hebbovho adaptačného prístupu.

7.1 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 7.1). Teda na vstupnej vrstve je M neurónov a na výstupe N_c neurónov. Pri konkurenčnom učení môžeme zhrnúť úvahy do nasledujúcich bodov:



Obr. 7.1: Zmena váh, ktoré smerujú k **vítaznému** neurónu

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

$$y_i(t) = \sum_{j=1}^M w_{ij}(t)x_j(t). \quad (7.1)$$

Ku vzťahu (7.1) sa vrátime v ďalšom **dôležitou** poznámkou.

2. Vyberieme **maximálnu hodnotu** $x_i(t)$ z $i = 1, \dots, N_c$ a urobíme nasledovnú operáciu

$$y_i(t) = \begin{cases} 1 & \text{ak } y_i = \max\{y_i, \forall i = 1, \dots, N_c\} \\ 0 & \text{inak} \end{cases} \quad (7.2)$$

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

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. Klíčovým momentom je stanovenie chybovej funkcie. V **konkurenčnom učení** ide o zámer, aby sa jednotlivé vstupy **u** premietli do hodnôt SV, ktoré smerujú k **vítaznému** neurónu, ktorý prísluší vstupu **u**. Teda v konečnom dôsledku budú mať rôzne vstupy **u** rôzne víťazné neuróny a stredy rôznych zhľukov by sa mali premietnuť do hodnôt SV, ktoré smerujú k jednotlivým víťazným neurónom. V podstate budeme mať N_c chybových funkcií typu³ :

$$J(t) = 0.5 \sum_{j=1}^M (w_{ij}(t) - x_j(t))^2. \quad (7.3)$$

Teda ide o rozdiel hodnôt SV, ktoré smerujú k i -temu víťaznému neurónu a hodnôt vstupných neurónov. Teda minimalizáciou tohto rozdielu, dochádza k premietnutiu vstupov do hodnôt SV, ktoré smerujú k jednotlivým víťazným neurónom.

Samotné odvodenie zmeny SV bude jednoduché, teda:

$$\Delta w_{ij}(t) = -\gamma \frac{\partial J(t)}{\partial w_{ij}(t)}, \quad (7.4)$$

³ $\mathbf{x}_j(t)$ je vstup !!!

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

$$\frac{\partial J(t)}{\partial w_{ij}(t)} = \begin{cases} (w_{ij}(t) - u_j(t)) & \text{ak } i \text{ je vitazny neuron} \\ 0 & \text{inak.} \end{cases} \quad (7.5)$$

Na základe (7.5) dostaneme zmenu SV v tvare

$$\Delta w_{ij}(t) = -\gamma(w_{ij}(t) - x_j(t)) \quad (7.6)$$

a z toho dostaneme vzťah pre novú hodnotu SV v tvare

$$w_{ij}(t+1) = w_{ij}(t) + \gamma(x_j(t) - w_{ij}(t)) \quad (7.7)$$

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 vrátim k vztoru (7.1), tak vidíme, že ide o skalárny súčin dvoch vektorov $\mathbf{w}_i(t)$ a $\mathbf{u}(t)$, ktorý má tvar :

$$\mathbf{w}(t)\mathbf{x}(t) = |\mathbf{w}| |\mathbf{x}| \cos(\phi), \quad (7.8)$$

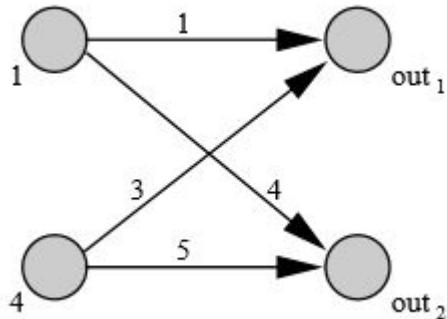
kde ϕ je uhol medzi vektormi $\mathbf{w}(t)$ a $\mathbf{u}(t)$. V prípade, že sú $|\mathbf{w}(t)|$ a $|\mathbf{u}(t)|$ rovné 1, potom samotný $\cos(\phi)$ sa rovná⁴

$$\cos(\phi) = \sum_j^M w_{ij}(t)u_j(t). \quad (7.9)$$

Ak je hodnota $|\mathbf{x}(t)|$ rovná 1, vstup je normalizovaný. Potrebu normalizácie si ozrejmíme na nasledujúcim príklade. Vzhľadom na zhlukovací charakter celej procedúry je potrebné dať pozor na prípady, keď sú vektori nesprávne zaradené, resp. vyhľadáva **nesprávny** neurón. Majme situáciu na obr. 7.2.

Ak na vstup dám vektor (1,4) a SV majú také hodnoty, ako je naznačené na obrázku, potom na základe vzťahu (7.1) dostaneme pre $\mathbf{x}_1 = 13$ a pre $\mathbf{x}_2 = 24$, teda podľa stratégie **WTA** vyhľadáva neurón 2. Ak si pripomenieme, že vlastne naším cieľom je premapovať vstupy

⁴Tomuto výrazu hovoríme **skalárny súčin**



Obr. 7.2: Príklad jednoduchého zhlukovania do 2 zhlukov v dvojrozmernom príznakovom priestore

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 (7.5) a dosaneme očakávané výsledky. Tu treba dať pozor, aký normalizačný prístup zvolíme. Napríklad ak zvolíme

$$\mathbf{u}_{\text{norm}} = \frac{\mathbf{u}}{\sqrt{\sum_{i=1}^M u_i^2}}, \quad (7.10)$$

kde M je počet neurónov na výstupe a to isté urobíme pre SV, dostaneme vektory $(\frac{1}{\sqrt{17}}, \frac{4}{\sqrt{17}})$, $(\frac{1}{\sqrt{10}}, \frac{3}{\sqrt{10}})$ a $(\frac{4}{\sqrt{41}}, \frac{5}{\sqrt{41}})$. Potom⁵ pre $x_1 = 0.997$ a pre $x_2 = 0.7$. Teda pomocou príslušnej normalizácie vstupných vektorov a samotných SV sme dosiahli **správne správanie** sa procedúry zhlukovania. Vyriešením tohto problému sme však dospeli k problému pre určité typy vektorov. Napr. v prípade, že použijeme normalizáciu, nebudeme schopní rozlísiť vektorov, ktoré majú rovnaký smer, ale rôzne veľkosti, lebo my v podstate spočítavame uhol medzi týmito vektorami. Riešením môže byť doplnenie vstupného vektorov a SV o ďalší príznak⁶, teda budeme pracovať nie v n-rozmernom priestore ale v $(n+1)$ -rozmernom priestore. Po normalizácii už tieto vektorov nebudú mať rovnaký smer. Nedostatok normalizácie sa prejavuje aj v prípade,

⁵**Pozor** - po normalizácii musí byť výraz $\sqrt{\sum_{i=1}^M u_i^2}_{\text{norm}} = 1 !!!$

⁶ napr. súčin prvkov vektorov – funkcionálne linky

ked' sú si dva vektory blízke z hľadiska smeru, ale majú rôznu dĺžku. Ak chceme predísť týmto problémom, musíme zmeniť prístup k výpočtu hodnoty pre **vítazný** neurón, teda vzťah (7.1).

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

$$y_i = \sum_{j=1}^M (w_{ij} - x_j)^2, \quad (7.11)$$

kde w_j sú hodnoty vstupu a x_j sú hodnoty SV, ktoré smerujú k neurónu i . Tento výpočet ale mení vzťah (7.2) do tvaru

$$y_i(t) = \begin{cases} 1 & \text{ak } y_i = \min\{y_i, \forall i = 1, \dots, N_c\} \\ 0 & \text{inak.} \end{cases} \quad (7.12)$$

Tento prístup má výhodu voči predošlému v tom, že **nie je** požadovaná normalizácia a odpadávajú problémy s vektormi rovnakých smerov. Ak sa vrátimy k predošlému prípadu, tak dostaneme výsledky $y_1 = 1$ a $y_2 = \sqrt{10}$, čo znamená, že vyhráva neurón 1.

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

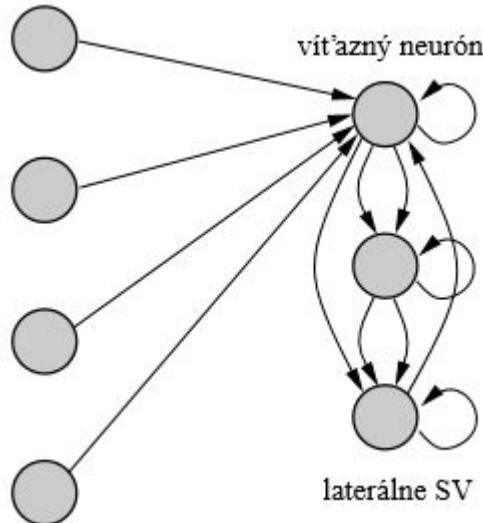
7.1.1 MAXNET

V práci [?] je konkurenčné učenie typu WTA doplnené tzv. **laterálnymi** prepojeniami medzi neurónmi tej istej vrstvy (viď obr. 7.3). Teda existujú SV aj v rámci jednej vrstvy a ich hodnoty sú definované ako

$$w_{ik} = \begin{cases} -\epsilon & \text{ak } i \neq k \\ 0 & \text{inak} \end{cases} \quad (7.13)$$

kde $i, k \in \langle 1, N_c \rangle$

Toto prepojenie pomôže zvýrazneniu rozdielu medzi víťazným neurónom a ostatnými neurónmi. Vzťah (7.13) je vhodný pre situáciu, keď vypočítavame novú hodnotu podľa (7.1). Ak použijeme na výpočet vzorec (7.11), tak $-\epsilon$ sa musí zmeniť na $+\epsilon$.



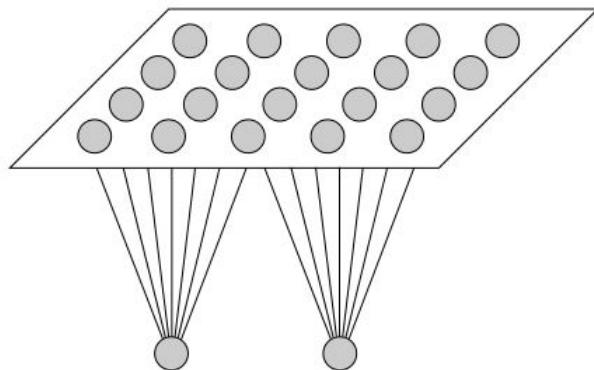
Obr. 7.3: Štruktúra NN typu MAXNET (laterálne SV)

7.2 Kohonenove siete

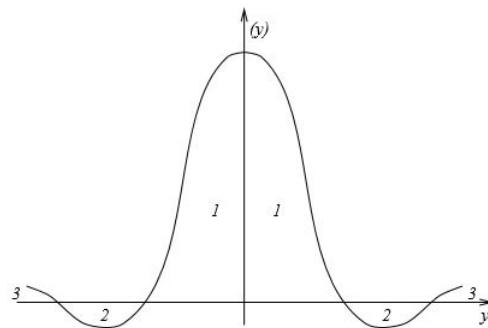
Kohonenove siete predstavujú veľmi dôležité rozšírenie konkurenčného učenia. Biologické systémy majú podobný tvar hlavne vo svojej percepčnej časti - oku. Rozšírenie konkurenčného učenia spočíva v tom, že sa pripúšťa princíp **viacerých víťazov - multiply WTA**. ďalej výstup z takejto NN je geometricky usporiadaný do nejakého útvaru napr. vedľa seba do obdlžníka, a teda existuje možnosť určenia **suseda**. Túto vrstvu nazývame **Kohonenovou vrstvou**. Súčasne sa samotné zhľukovanie organizuje takým spôsobom, že susedné neuróny resp. SV, ktoré k nim smerujú, majú podobné hodnoty a tie, ktoré sú ďalej od seba, majú rozdielnejšie hodnoty. Príčinou toho celého je zavedenie tzv. **funkcie susednosti** Λ_{ij} . Vo väčšine prípadov ide o funkciu v tvare:

$$\Lambda_{ij} = h(t) \cdot e^{-\left(\frac{(d_j)}{r_i(t)}\right)^2}, \quad (7.14)$$

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 . Funkcia susednosti môže byť vyjadrená v tvare **mexického klobúka**, ako je uvedené na obr. 7.5.



Obr. 7.4: Typická topológia Kohonenovej NN

Obr. 7.5: Možný tvar funkcie susednosti Λ_{ij}

Potom nastane modifikácia vzorca (7.6) do tvaru:

$$\Delta w_{ij}(t) = -\gamma \Lambda_{ij}(w_{ij}(t) - u_j(t)) \quad (7.15)$$

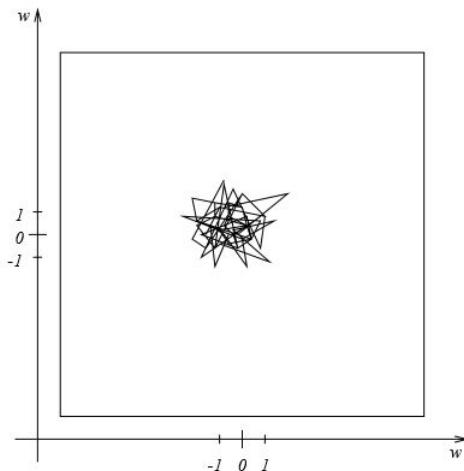
a v konečnom dôsledku bude výpočet novej hodnoty SV mať tvar⁷ :

$$w_{ij}(t+1) = w_{ij}(t) + \gamma \Lambda_{ij}(u_j^k(t) - w_{ij}(t)). \quad (7.16)$$

Cez túto funkciu susednosti sú teda spojené jednotlivé neuróny v geometricky usporiadanej výstupnej vrstve.

Proces učenia sa takejto neurónovej sieti je možné demonštrovať (v prípade kohonenových sietí používajúcich Euklidovu vzdialenosť) na grafe kohonenovej siete, ktorého tvar pre dvojrozmerný vstupný vektor je uvedený na

⁷voči vzorcu (7.15) sa vymenili argumenty v zátvorke



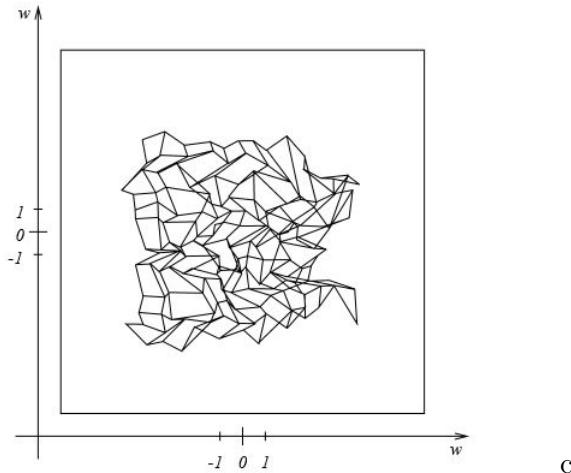
Obr. 7.6: Stav SV po inicializácii

obrázkoch 7.6 a ?? (Jednotlivé uzly predstavujú zobrazenie váhových vektorov vedúcich k jednotlivým neurónom v kohonenovej vrstve a prepojenia uzlov vyjadrujú topologickú susednosť týmto uzlom zodpovedajúcich neurónov).

Je nutné si uvedomiť, že uvedené grafy zobrazujú situáciu SV vedúcich len z dvoch vstupov. V prípade, ak je vstupov viac, sa pre potreby vizualizácie postupuje tak, že sa vyberajú a graficky zobrazujú dvojice SV. Zobrazenie v trojrozmernom priestore je dosť zriedkavé.

Ako demoštráciu je vhodne si pozrieť video na youtube kanaly s adresou

<https://www.youtube.com/watch?v=zyYZuAQZWtM>.



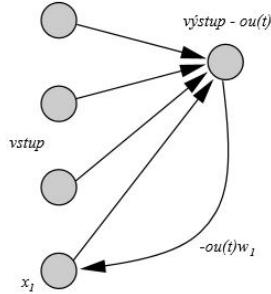
7.3 Metóda hlavných komponentov

Majme dátu z M-rozmerného príznakového priestoru. Tieto dátu majú svoju informačnú hodnotu. Predpokladajme, že **chceme prejsť** z M-rozmerného priestoru do N-rozmerného priestoru, kde $N < M$, 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 (popísaný napr. v [?]). Cieľom tejto časti je ukázať, že pomocou NN je tiež možné túto transformáciu zvládnuť. Konečná výhoda NN implementácie je vo využití výpočtovej výhodnosti, hlavne pri **paralelných** systémoch. Táto procedúra predstavuje **zhustenie** informácie pri zachovaní čo najväčzej informačnej hodnoty dát. Dôležité je uvedomiť si, že tento prístup predstavuje zhustenie, kde **úplná** rekonštrukcia dát je problematická.

7.3.1 Ojove adaptačné pravidlo zmeny SV

Majme jednoduchú NN s \mathbf{n} vstupmi a jedným výstupným neurónom (vid obr. 7.7). Nech výstupný neurón je **lineárneho typu**, teda

$$x(t) = \sum_{i=1}^n w_i(t)u_i(t) = \mathbf{w}(t)\mathbf{u}(t). \quad (7.17)$$



Obr. 7.7: Priklad NN pre hľadanie prvého hlavného komponentu

Predpokladajme **Hebbove synapsie** medzi neurónmi, teda je zrejmé, že adaptačné pravidlo bude mať tvar⁸

$$\Delta \mathbf{w}(t) = x(t)\mathbf{u}(t), \quad (7.18)$$

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

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \gamma x(t)\mathbf{u}(t) \quad (7.19)$$

pre $\forall i = 1, \dots, n$. Problém vzorca (7.19) spočíva v tom, že pre $t \rightarrow \infty w_i$ neúmerne rastie, čo pri reálnych systémoch spôsobuje problémy. Predĺžť tejto saturáciu môžeme určitou formou normalizácie výrazu (7.19). Oja navrh hol nasledovnú formu (výrazy sú vo vektorovom tvare):

$$\mathbf{w}(t+1) = \frac{\mathbf{w}(t) + \gamma \Delta \mathbf{w}(t)}{L(\gamma)} \quad (7.20)$$

kde $L(\gamma)$ je $|\mathbf{w}(t) + \gamma \Delta \mathbf{w}(t)|$ a po dosadení zo vzorca (7.18) má tvar

$$L(\gamma) = \sqrt{|\mathbf{w}(t)|^2 + 2\gamma x(t)\mathbf{w}(t)\mathbf{u}(t) + \gamma^2 x(t)^2 |\mathbf{u}(t)|^2} \quad (7.21)$$

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 $|\mathbf{w}(t)| = 1$, viď. 7.20):

$$\frac{\partial L}{\partial \gamma}_{\gamma=0} = x(t) \underbrace{\mathbf{w}(t)\mathbf{u}(t)}_{x(t)} + 0 \quad (7.22)$$

⁸vo vektorovom zápisе pre prehľadnosť označenie tranponovania vyniechávame

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

čo po úprave znamená, že

$$\frac{\partial L}{\partial \gamma} \Big|_{\gamma=0} = x^2(t), \quad (7.23)$$

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

$$L(\gamma) = 1 + \gamma x^2(t), \quad (7.24)$$

ale v podstate my máme v zmysle situácie v (7.20) tvar

$$\frac{1}{1 + \gamma x^2(t)}, \quad (7.25)$$

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

$$\frac{1 - \gamma x^2(t)}{1 - \gamma^2 x^4(t)}. \quad (7.26)$$

Pretože γ môže byť zvolené dostatočne malé, $\gamma^2 x^4(t) \rightarrow 0$, teda menovateľ sa blíži k 1 a môžeme písat'

$$\frac{1}{1 + \gamma x^2(t)} \doteq 1 - \gamma x^2(t). \quad (7.27)$$

Ak to dosadíme späť do vzorca (7.20) a znova **zanedbáme** γ^2 , dostaneme

$$\mathbf{w}(t+1) = (\mathbf{w}(t) + \gamma x(t)\mathbf{u}(t))(1 - \gamma x^2(t)), \quad (7.28)$$

čo v konečnom dôsledku znamená

$$\mathbf{w}(t+1) = (\mathbf{w}(t) + \underbrace{\gamma x(t)(\mathbf{u}(t) - x(t)\mathbf{w}(t))}_{\mathbf{u}'(t)}), \quad (7.29)$$

kde je zrejmé, že

$$\Delta \mathbf{w}(t) = \gamma x(t)(\mathbf{u}(t) - x(t)\mathbf{w}(t)), \quad (7.30)$$

teda nová hodnota SV sa vypočíta ako

$$\mathbf{w}(t+1) = (\mathbf{w}(t) + \gamma x(t)\mathbf{u}'(t)), \quad (7.31)$$

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

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

Vzorec (7.29) sa tiež nazýva **Ojove adaptačné** pravidlo zmeny SV. Po nájdení GS NN dostaneme v hodnotách vektora \mathbf{w} **prvý hlavný komponent**. To, že NN s takýmto adaptačným pravidlom určite nájde svoju GS, je ukázané v [?].

Rozšírime teraz náš zámer o hľadanie ďalších hlavných komponentov zhusteného priestoru komponentov. Teda majme jednoduchú NN s M vstupmi a N výstupmi (viď obr. 7.8). Súčasne platí, že $N < M$ a výstupné neuróny sú lineárneho typu. Pre zjednodušenie zápisu si označme jednotlivé neuróny vo vstupnej vrstve nasledovne:

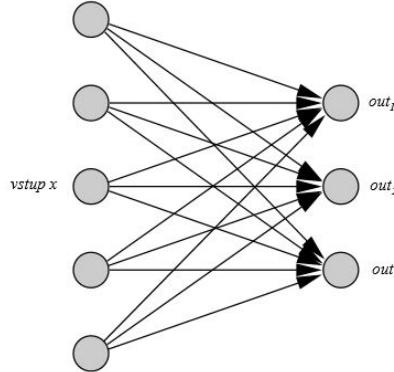
vstupné

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

a výstupné

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

Potom výstup (t.j. aktiváciu) neurónu i vypočítame ako (v skalárnom výjadrení)



Obr. 7.8: Ojova sieť pre výpočet 3 hlavných komponentov

$$x_i(t) = \sum_{l=0}^{M-1} w_{ij} u_j. \quad (7.32)$$

Vo vzorci (7.33) potom dostaneme tzv. zovšeobecnený tvar Hebbovho adaptačného pravidla v tvare¹⁰

$$\Delta w_{ij}(t) = \gamma x_i(t) \left(u_j(t) - \sum_{l=0}^i w_{lj}(t)x_l(t) \right). \quad (7.33)$$

Ak do vzorca (7.33) dosadíme $i = 0$, tak dostaneme vzorec pre adaptačné pravidlo, ktoré sme už odvodili vo vzťahu (7.30).

Teraz opäť pre následnú výhodnosť si vzorec (7.33) prepíšeme do tvaru

$$\Delta w_{ij}(t) = \gamma x_i(t) \left(u_j(t) - \underbrace{\sum_{l=0}^{i-1} w_{lj}(t)x_l(t)}_{u'_j(t)} - w_{ij}(t)x_i(t) \right). \quad (7.34)$$

V podstate tým dostávame situáciu, keď máme jedno adaptačné pravidlo, kde sa efektívny vstup do jednotlivých výstupných neurónov mení. V rámci jednej učebnej procedúry sa nám podľa (7.33) 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**, kedy $u'(t)$ má podľa vzorca (7.34) tvar ($\mathbf{i} = \mathbf{0}$)

$$u'_j(t) = u_j(t), \quad (7.35)$$

2. **druhý hlavný komponent ($\mathbf{i} = \mathbf{1}$)**

$$u'_j(t) = u_j(t) - w_{0j}(t)x_0(t), \quad (7.36)$$

3. **tretí hlavný komponent ($\mathbf{i} = \mathbf{2}$)**

$$u'_j(t) = u_j(t) - w_{0j}(t)x_0(t) - w_{1j}(t)x_1(t), \quad (7.37)$$

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 ne-lineárnych neurónov bol Ojo-om vyvinutý podobný postup.

¹⁰ berte do úvahy označenie neurónov

Kapitola 8

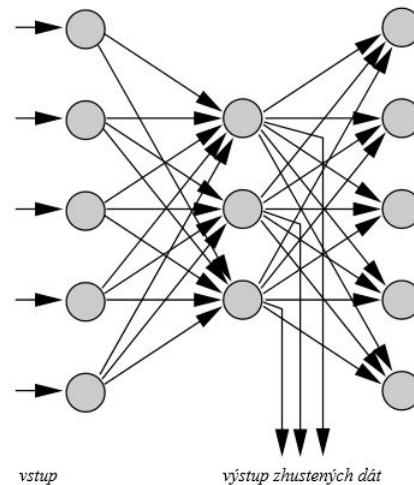
Hybridné metódy učenia na FF NN

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.

8.1 Nekontrolované učenie metódou 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 topológií FF NN v tvare M-N-M, kde $N < M$ a väčšinou $N = \log_2 M$. Príklad takejto topológie je na obrazku 8.1. Takáto NN sa dá vhodne využiť na kompresiu dát. Je použitý klasický BP prístup, ale s tým, že na vstup a výstup dávame tie isté dátá. Zhustené dátá 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 dátá (predstavujú množinu všetkých možností), ktoré môžu v dátach nastaviť a také isté dátá očakávame na výstupe. Snažíme sa dostať do stavu, keď NN vie spoľahlivo realizovať celý proces a SV sa už nemenia, teda do stavu GS.
- Potom sme v stave, keď môžeme spoľahlivo realizovať kompresiu dát a získať zhustené dátá zo skrytej vrstvy.



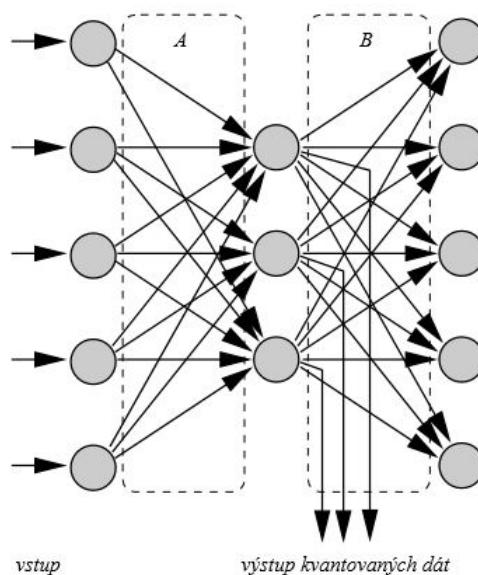
Obr. 8.1: Zhustňovanie dát z M na N

- Tým máme zabezpečenú aj spätnú rekonštrukciu dát.

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

8.2 Metóda Counterpropagation

Jednou z ďalších možností hybridného prístupu je kvantovanie vstupu, pričom musí byť splnená podmienka opäťovnej rekonštrukcie. Práve toto vykonáva metóda **Counterpropagation**, ktorá kombinuje vyššie uvedené metódy nasledujúcim spôsobom: **Prvá časť** pracuje na princípe konkurenčného učenia a **druhá časť** používa metódu BP (viď obr. 8.2).



Obr. 8.2: 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. Prezentovanie vstupov do NN a výpočet hodnoty neurónu v skrytej vrstve podľa vzorca (7.11). Výpočet minimálnej hodnoty stavu jednotlivých neurónov podľa (7.12).

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

$$er_i = (e_i - y_i). \quad (8.1)$$

5. Teraz sa budú realizovať dve adaptačné pravidlá a to

- (a) pre konkurenčné učenie sa zmena SV vypočíta ako

$$\Delta w_{kj}(t) = \gamma_1(w_{kj}(t) - u_j(t)) \quad (8.2)$$

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

- (b) pre druhú časť, teda BP, sa adaptačné pravidlo pre zmenu SV tejto časti NN vyjadrí podľa vzorca (6.16) pre prípad výstupného neurónu (vzorec (6.20)) s použitím lineárnej aktivačnej funkcie na výstupe NN. Teda

$$\Delta w_{ik}(t) = -\gamma_2(e_i - y_i(t))u_k. \quad (8.3)$$

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

$$y_i(t) = \sum_{j=1}^{N_z} w_{ij}u_j = w_{ik}. \quad (8.4)$$

Potom môžeme vzorec (8.3) upraviť do tvaru

$$\Delta w_{ik}(t) = \gamma_2(w_{ik}(t) - e_i(t)). \quad (8.5)$$

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

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

Kapitola 9

Rekurentné NN

V nasledujúcich kapitolách je poskytnutý prehľad základných metód učenia pre rekurentné NN. Pod rekurentnými NN (ďalej RC NN) rozumieme NN, ktoré majú synapsie orientované **rôznymi** smermi. V takejto štruktúre, môže byť neurón **súčasne vstupný aj výstupný**. V terminológii NN v takom prípade hovoríme o **duálnom neuróne**. V [?] je poskytnutý základný konceptný prehľad RC NN. Otázka „**kedy použiť RC NN**“ je dosť zložitá. Zvyčajne sa dodržiava pravidlo, že RC NN sa použije až keď:

- je zrejmé, že sme neuspeli s FF NN,

alebo

- činnosť FF NN je neefektívna.

Vo všeobecnosti môžeme konštatovať, že RC NN predstavujú z hľadiska dynamiky zložitejší systém ako FF NN. Problémy **neurodynamiky** sú pri RC NN veľmi aktuálne z hľadiska GS a konvergencie RC NN. Takisto ako pri FF NN rozdeľujeme metódy umenia aplikované na RC NN topológie do dvoch základných skupín:

- kontrolované učenie na RC NN (príkladom môže byť Hopfieldova sieť ako simulácia pamäte, BP pre RC NN, BP pre Elmanove RC NN)
- nekontrolované učenie na RC NN (tu sú dominantné hlavne metódy učenia ART so širokou aplikačnou oblasťou).

V rámci RC NN existuje podskupina čiastočne rekurentých sietí (**partialy RC NN**). U čiastočne rekurentných sietí sa predpokladá existencia skupín neurónov, ktoré nesplňajú vlastnosť rekurencie. Špeciálne topológie RC NN navrhli **Elman a Jordan**. Sú určené väčšinou na spracovanie sekvenčných vzoriek ako celku.

9.1 Kontrolované učenie na RC NN

Je zrejmé, že pod kontrolovaným učením rozumieme proces, kedy ponúkame na vstup NN nejaké vzorky, a k nim príslušné výsledky očakávame na výstupe. Očakávaný výstup sa môže rovnať vstupu, hlavne keď ide o simuláciu pamäte. Na rozdiel od **nekontrolovaného učenia BP**, ide tu o výstup, kým v spomínanom BP prístupe nás zaujímali výsledky na skrytej vrstve.

9.1.1 Hopfieldove siete

Hopfieldove siete (ďalej HS) predstavujú najjednoduchšiu verziu architektúry RC NN. Je potrebné pripomenúť, že táto sieť sa skladá z **duálnych** neurónov (viď obrázok 9.1). Ďalej je potrebné si uvedomiť, že v prípade HS ide o asynchronnu činnosť celej HS, teda jednotlivé neuróny menia svoje stavy **asynchronne**. Tiež je nutné poznamenať, že zmeny SV sa realizujú podľa Hebbovho pravidla. Využitie takejto siete môže byť napríklad pri simulácii asociatívnej pamäte. Teda HS budeme učiť nejaké vzorky si uchovať a potom po naučení takejto siete ak dámme na vstup nejakú vzorku, očakávame, aby na výstupe bola tá istá, to znamená aby ju HS mala v pamäti. Celý proces teda môžeme rozdeliť do dvoch fáz, a to:

- učenie (zmena SV),
- testovanie.

Vstup do jednotlivých neurónov je vstup z externého sveta a výstupy z ostatných neurónov.

$$in_i(t) = \sum_{j=1}^N w_{ij}x_j(t) + u_i(t) + \theta_i ; \quad w_{ii} = 0 \text{ pre } i = 1, \dots, N \quad (9.1)$$

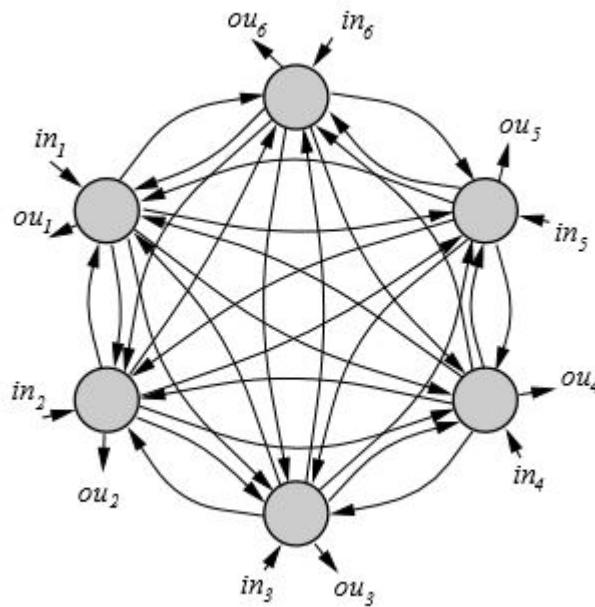
Pre stavy neurónov platí:

$$x_i(t) = \begin{cases} +1 & \text{ak } in_i(t) > T \\ -1 & \text{ak } in_i(t) < T \\ x_i(t-1) & \text{inak} \end{cases} \quad (9.2)$$

Kde T je prah (threshold), zvolený človekom.

Adaptačné pravidlo je **Hebbovo typu**. Hovorí, že SV sa zosilní, ak pred- a postsynaptický neurón sú rovnakého znamienka a zoslabí sa, ak majú rôzne znamienka. Teda pre zmenu váh¹ platí:

¹Pozor tu ide o Δw a nie w !!!



Obr. 9.1: Jednoduchá Hopfieldova sieť so 6 neurónmi, N=6

$$\Delta w_{ij}(t) = \gamma x_i(t)x_j(t) \quad (9.3)$$

Po naučení P prvkov má SV tvar

$$w_{ij} = \begin{cases} \sum_{p=1}^P x_i^p x_j^p & ak \ i \neq j \\ 0 & inak \end{cases} \quad (9.4)$$

Po určitej dobe siet dosiahne stav GS. Ak budeme predpokladať symetrické SV teda $w_{ij} = w_{ji}$ a ak si zvolíme **Ljapunovovu funkciu** pre energiu Hopfieldovej siete v tvare

$$E = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N x_i x_j w_{ij} - \sum_{i=1}^N \theta_i x_i \quad (9.5)$$

kde θ_i - je váha prahového prepojenia pre i-ty neurón a $N = 6$.

Pod Ljapunovovou funkciou rozumieme funkciu, ktorá pre ľuboľný dynamicky systém (napr. neurónovú sieť) vie nám zabezpečiť v jej extréme (minime alebo maxime) stabilitu daného dynamického systému.

Ak teda chceme aby HS našla svoju GS tak ΔE musí byť záporná a prejavovať klesajúci charakter, čo sa dá aj dokázať viď [?]. Problémom HS je malé množstvo uchovateľných vzoriek. Jedná sa zhruba o $0.15N$ vzoriek, kde N je počet neurónov HS a túto vlastnosť nazývame kapacita Hopfiel-dovej siete. Vidíme že je veľmi nízka to znamená potrebujeme veľmi veľa neurónov na zapamätanie vzoriek. Teda na napr. na zapamätanie 15 vzoriek potrebujeme 100 neurónov v HS.

9.1.2 Problém obchodného cestujúceho (TSP)

V ďalšom si ukážeme riešenie optimalizačného problému pomocou Boltzmannovej siete a riešením problému obchodného cestujúceho.

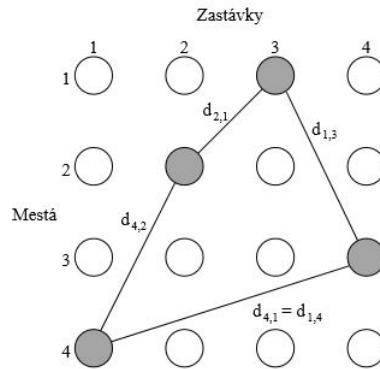
Majme danú množinu n miest a vzdialenosť $d_{i,j}$ pre všetky dvojice miest i, j . Potrebujeme zistiť, v akom poradí má obchodný cestujúci prechádzať mestá tak, aby sa vrátil do mesta, z ktorého vyrazil a zároveň každé mesto na trase navštívil práve jedenkrát. Podmienkou však je, že vzdialosť, ktorú precestoval, je najkratšia možná. Teda vieme koľko máme miest a vieme aké sú medzi mini vzdialenosťami. Chceme od systému aby nám vypočítal postupnosť ktorá nám zabezpečí že prejdeme najkratšiu cestu. Výsledok bude postupnosť miest resp. ich cyklické opakovanie (permutacia) poradia.

Riešenie tohto problému je prípustné, ak sú splnené nasledujúce obmedzenia :

- obchodný cestujúci prechádza každým mestom práve jedenkrát,
- jeho cesta začne a skončí v jednom a tom istom meste..

Prípustnosť riešenia je teda daná nájdením mieru optimality riešenia popisuje súčet vzdialostí medzi mestami.

Neurónová sieť, ktorá tento problém rieši, musí vyjadriť pre dané mesto, ktoré mesto mu predchádza a ktoré mesto za ním nasleduje. Toto je vyjadriteľné v sieti s n^2 neurónmi pri maticovom usporiadane uvedenom na obr. 9.2



Obr. 9.2: Návrh neurónovej siete pre TSP pri $n = 4$. Kvôli prehľadnosti nie sú zakreslené prepojenia neurónov. Riešenie je zobrazené pomocou čiar. Riešením je postupnosť miest 4-2-1-3, prípadne jej cyklická permutácia (permutácia znamená obmieňanie poradia cesty v danej množine 4 miest v danom riešení)

Je zrejmé, že usporiadanie v tomto tvaru pomáha pri pochopení riešenia, inak je nepodstatné. V tejto situácii je nutné si uvedomiť že synaptické váhy budú mať tvar $w_{ij,kl}$ teda budú **prepájať mesto i ak bude j-ta zástavka a s mestom k ak to mesto bude l-ta zástavka**.

Vzťah pre energiu siete je možné upraviť pomocou zavedenia dvoch indexov pre jeden neurón x_i^j . Jeden index neurónu predstavuje mesto a druhý index poradie mesta v ceste. Jedná sa len o technickú úpravu vzťahu, význam zostane nezmenený. Teraz sformulujeme túto úlohu pomocou novej sústavy dvojstavových premenných tak, aby hľadanie prípustného riešenia mohlo byť vyjadrené ako minimalizácia funkcie týchto nových premenných.

Zadefinujme maticu \mathbf{x} typu $n \times n$ s prvkami x_i^j , ktoré nadobúdajú hodnoty 0 alebo 1, pre $i, j \in \langle 1, n \rangle$. $x_i^j = 1$ vtedy, keď obchodný cestujúci prechádza mestom i v j -tom kroku. V opačnom prípade $x_i^j = 0$. V podstate hodnoty x_i^j sú aktivácie neurónov.

Prípustnému riešeniu pôvodnej úlohy teraz zodpovedá stav matice, ktorý sa dá popísť nasledovne :

1. v každom riadku je najviac jedna jednotka, tj. pre a -te mesto platí

$$x_a^j x_a^l = 0 \quad (9.6)$$

ak $j \neq l$,

2. v každom stĺpci je najviac jedna jednotka, tj. pre b -ty krok obchodného cestujúceho platí

$$x_i^b x_k^b = 0 \quad (9.7)$$

ak $i \neq k$,

3. v matici je práve n jednotiek, tj.

$$\sum_i^n \sum_j^n x_i^j = n \quad (9.8)$$

4. súčet vzdialenosť medzi mestami je určený maticou vzdialenosťí, pričom celková dĺžka cesty je

$$D = \frac{1}{2} \sum_i^n \sum_{k \neq i}^n \sum_b^n d_{ki} x_k^b (x_i^{b-1} + x_i^{b+1}) \quad (9.9)$$

Teraz vytvoríme funkcie E_1, E_2, E_3 a E_4 , ktoré nadobúdajú minimálne hodnoty pri splnení predchádzajúcich podmienok. Funkcie sú vytvárané na základe vzťahov uvedených v týchto podmienka.

$$E_1 = \frac{\gamma_1}{2} \sum_a^n \sum_j^n \sum_{l \neq j}^n x_a^j x_a^l \quad (9.10)$$

$$E_2 = \frac{\gamma_2}{2} \sum_i^n \sum_b^n \sum_{k \neq i}^n x_i^k x_k^b \quad (9.11)$$

$$E_3 = \frac{\gamma_3}{2} \left(\sum_i^n \sum_j^n x_i^j - n \right)^2 \quad (9.12)$$

$$E_4 = \frac{\gamma_4}{2} \sum_i^n \sum_{k \neq i}^n \sum_b^n d_{ki} x_k^b (x_i^{b-1} + x_i^{b+1}) \quad (9.13)$$

Vo vzťahoch (9.10) – (9.13) $\gamma_1, \gamma_2, \gamma_3$ a γ_4 sú voliteľné parametre učenia. Takže riešenie je optimálne, ak

$$E = E_1 + E_2 + E_3 + E_4$$

nadobúda svoje minimum.

Na tomto mieste je vhodné pripomenúť si, ako vyzerá energetická funkcia pre diskrétny model **Hopfieldovej siete** pri maticovom očíslovaní neurónov:

$$E = -\frac{1}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n w_{ij,kl} x_i^j x_k^l - \sum_i^n \sum_j^n \theta_i^j x_i^j \quad (9.14)$$

Prepísaním (9.10), (9.11), (9.12) a (9.13) na tvar energetickej funkcie (9.14) dosiahneme to, že tvar „našej“ funkcie E bude ekvivalentný tvaru funkcie E pre HS, ktorá sa počas konvergencie siete minimalizuje (základná vlastnosť BS). Po tomto prepísaní nám teda bude stačí porovnať výsledky s (9.14) a ľahko vypočítame nastavenie váh a prahov. Nasledujúce úpravy spočívajú v zavedení symbolu Kroneckerovo delta, tj.

$$\delta_{ij} = \begin{cases} 1 & ak \quad i \neq j \\ 0 & ak \quad i = j \end{cases} \quad (9.15)$$

čo nám umožní doplniť do niektorých výrazov ďalšie sumy. Z (9.10) vyplýva:

$$E_1 = \frac{\gamma_1}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n \delta_{ik}(1 - \delta_{jl}) x_i^j x_k^l \quad (9.16)$$

Výraz $\delta_{ik}(1 - \delta_{jl})$ je rovný 0, ak $i \neq k$ alebo $j = l$. Tento výraz je vlastne výraz (9.14), kde

$$w_{ij,kl,1} = -\gamma_1 \delta_{ik}(1 - \delta_{jl}) \quad (9.17)$$

následne

$$E_2 = -\frac{\gamma_2}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n -\gamma_2 \delta_{ik}(1 - \delta_{jl}) x_i^j x_k^l \quad (9.18)$$

Analogicky z (9.11) dostoneme

$$w_{ij,kl,2} = -\gamma_2 \delta_{jl}(1 - \delta_{ik}) \quad (9.19)$$

Z (9.12) dostoneme

$$E_3 = \frac{\gamma_3}{2} \left(\sum_i^n \sum_j^n \sum_k^n \sum_l^n x_i^j x_k^l - 2n \sum_i^n \sum_j^n x_i^j + n^2 \right) \quad (9.20)$$

Prenásobením výrazu v zátvorke výrazom $\frac{\gamma_3}{2}$ dostoneme

$$E_3 = -\frac{1}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n -\gamma_3 x_i^j x_k^l + \sum_i^n \sum_j^n -\gamma_3 n x_i^j + \frac{\gamma_3}{2} n^2 \quad (9.21)$$

Pretože posledný člen $\frac{\gamma_3}{2} n^2$ nezmení polohu minima vyššie uvedenej funkcie E_3 , zanedbáme ho a dostávame

$$w_{ij,kl,4} = -\gamma_3, \quad \theta_{ij} = -\gamma_3 n. \quad (9.22)$$

Zo (9.13) vyplýva

$$E_4 = \frac{\gamma_4}{2} \sum_i^n \sum_k^n \sum_b^n d_{ki} x_k^b x_{i,b-1} + \frac{\gamma_4}{2} \sum_i^n \sum_k^n \sum_b^n d_{ki} x_k^b x_{i,b+1}) \quad (9.23)$$

pričom platí, že $d_{ik} = 0$, ak $i = k$. Zavedením Croneckerovho symbolu δ_{lx} a nahradením indexu b indexom l vo výraze pre E_4 dostávame

$$E_4 = \frac{\gamma_4}{2} \sum_i^n \sum_b^n \sum_k^n \sum_l^n d_{ki} \delta_{lb} x_{kl} x_{i,b-1} + \frac{\gamma_4}{2} \sum_i^n \sum_b^n \sum_k^n \sum_l^n d_{ki} \delta_{lb} x_{kl} x_{i,b+1} \quad (9.24)$$

$E_4 = 0$, ak $l \neq b$. Položíme $j = b-1$ v prvej časti výrazu a $j = b+1$ v druhej časti a dostaneme

$$E_4 = \frac{\gamma_4}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n (d_{ki} \delta_{l,j+1} x_{kl} x_i^j + d_{ki} \delta_{l,j-1} x_{kl} x_i^j) \quad (9.25)$$

$$E_4 = -\frac{1}{2} \sum_i^n \sum_j^n \sum_k^n \sum_l^n -\gamma_4 d_{ki} (\delta_{l,j+1} + \delta_{l,j-1}) x_k^l x_i^j \quad (9.26)$$

$$w_{ij,kl,4} = -\gamma_4 d_{ki} (\delta_{l,j+1} + \delta_{l,j-1}) \quad (9.27)$$

Takže nakoniec pre nastavenie váh prepojení v sieti medzi neurónmi x_i^j a x_{kl} bude platí

$$w_{ij,kl} = w_{ij,kl,1} + w_{ij,kl,2} + w_{ij,kl,3} + w_{ij,kl,4} \quad (9.28)$$

$$w_{ij,kl} = -\gamma_1 \delta_{ik} (1 - \delta_{jl}) - \gamma_2 \delta_{jl} (1 - \delta_{ik}) - \gamma_3 - \gamma_4 d_{ki} (\delta_{l,j+1} + \delta_{l,j-1}) \quad (9.29)$$

Podobne pre θ dostaneme

$$\theta_{ij} = -\gamma_3 n. \quad (9.30)$$

Po odvodení vyššie uvedených vzťahov môžeme uviesť nasledujúci **algoritmus pre TSP**:

Krok 1. Priradenie váh prepojeniam $w_{ij,kl} = -\gamma_1\delta_{ik}(1-\delta_{jl}) - \gamma_2\delta_{jl}(1-\delta_{ik}) - \gamma_3 - \gamma_4d_{ki}(\delta_{l,j+1} + \delta_{l,j-1})$, $\theta_{ij} = -\gamma_3n$ pre $1 \leq i, j, k, l \leq n$, $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ sú zvolené parametre siete (sú meniteľné),

Krok 2. Inicializácia

$$x_i^j(0) = r \quad (9.31)$$

pre $1 \leq i, j \leq n$,

V tejto formule $x_i^j(t)$ je výstup vrcholu „ij“ v 4ase $t = 0$ a r je náhodná premenná, ktorá nadobúda hodnotu 0 alebo 1.

Krok 3. Iterácia pokiaľ siet' ne konvergovala

$$x_i^j(t+1) = f\left(\sum_{k=1}^n \sum_{i=1}^n w_{ij,kl} x_k^l(t)\right) \quad (9.32)$$

pre $1 \leq l, j \leq n$, kde f je skoková aktivačná funkcia. Krok končí, ak siet' skonvergovala, tj. jej stav sa už nemení. Tu môže dôjsť k zacykleniu siete.

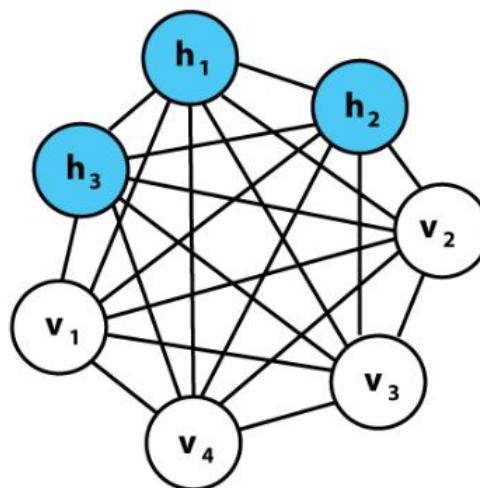
Krok 4. Opakovanie od kroku 2. Ak došlo k zacykleniu siete, je potrebný nový výpočet začínajúci nastavením nových počiatočných hodnôt siete. Tiež je možné zmeniť parametre siete a nastaviť nové váhy, tj. začať krokom 1.

Horeuvedený príklad ilustruje využitie Hopfieldovej siete pre riešenie optimalizačnej úlohy. Mnohé aplikácie HS je možné požiť na problematiku zakódovania vstupov a ich následného spätného odkodovania.

9.1.3 Boltzmanov stroj a jeho varianty

Boltzmanov stroj je špeciálny prípad Hopfieldovej siete, ktorá využíva myšlienku Boltzmanova z fyziky častíc, ktorá tvrdí že Teplota systému je priamo úmerná energii teda stav stability systému častíc závisí o ich teplôty systému, ktorá je naviazaná na rýchlosť častíc a táto rýchlosť je závislá od zrážok častíc v systéme. Energia je teda Ljapunova funkcia takého dynamického systému.

majme nasledovnú matematickú štruktúru neurónovej siete



Obr. 9.3: Jednoduchá štruktúra BS s obojstrannými prepojeniami

V topológii tejto neurónovej siete máme tzv V neuróny a H neuróny. (Visible and hidden neurons).

V princípe môžeme napísat že Boltzmanov stroj je Hopfieldova sieť, v ktorej platia nasledovné pravidlá:

- majme typickú štruktúru HS,kde neuróny môžu nadobúdať hodnoty 0 resp. 1
- váhy budú sa nastavovať podľa vzorca 9.3. Následne ak táto prinesie v energii vyššiu hodnotu takúto zmenu prijmeme, ak nie tak takúto zmenu odmietneme.
- vypočítame tzv. Consensualnej funkcie CF resp ΔCF , resp. Pravdepodobnosť že zmenu prijmeme pri danom ΔCF a normovaní koeficientom

T, ktorý nazývame Teplota procesu Boltzmanovho stroja

- Ak pravdepodobnosť je väčšia alebo rovná ako zvolený prahový parameter TR (TR - threshold ration $TR \in \langle 0, 1 \rangle$)
- Teplotu T znížime podľa pravidla $T(t + 1) = 0.95 * T(t)$
- Adaptáciu zastavíme ak T dosiahne definovanú hodnotu alebo systém nemení Energiu (CF) za definovaný počet iterácií za sebou nasledujúcich

Vo všeobecnosti teda môžeme napísat nasledovný postup pri Boltzmanom stroji o 6 neurónoch ako nasledovný (topológia je na obrázku 9.1) :

1. inicializujeme váhy v neurónovej sieti, zvolíme si parameter Teploty T a prahový parameter TR
2. vypočítame hodnotu váh podľa vzorca 9.3 a následne sa ukážu aktivácie na neurónoch
3. vypočítame hodnotu CF pre danú Boltzmanovu siet v tvare

$$CF = \sum_i^N \sum_{j \leq i}^N w_{ij} x_i x_j \quad (9.33)$$

a následne pre vypočítame hodnotu CF pre každý neurón ato nasledovne

$$\Delta CF_i = (1 - 2x_i)(w_{ii} + \sum_{i \neq j}^N w_{ij} x_i) \quad (9.34)$$

Prvý činiteľ $(1 - 2x_i)$ indikuje stav ak je $x_i = 0$ výsledok činiteľa je 1 resp. ak je $x_i = 1$ tak výsledok činiteľa je -1 .

4. následne sa vypočíta pravdepodobnosť prijatia zmeny váhy ktorý smeruje k neurónu i (pri teplote T) , označme to ako funkciu E

$$E(i, T) = \frac{1}{1 + e^{-\frac{\Delta CF_i}{T}}} \quad (9.35)$$

5. následne ak $E(i, T) \leq TR$ prijmime zmenu váhy resp. $E(i, T) > TR$ odmietnime zmenu váhy

6. ak odmietneme zmenu váhy tak sa vrátíme do bodu 3 pre ďalšie i (ďalší neurón)
7. ak prijmeme zmenu váhy tak upravíme Teplotu systému do tvaru

$$T_{new} = 0,95 * T_{old} \quad (9.36)$$

Celý systém zastaví v dvoch prípadoch ato

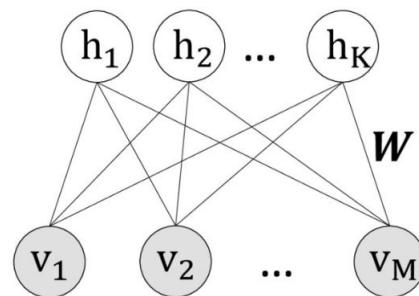
1. Teplota T dosiahne vopred definovanú hodnotu
2. systém nemení Energiu (CF) za definovaný počet iterácií za sebou nasledujúcich

Teda výsledkom BS je fakt že cez V-neuróny dostaneme vstup, ktorý sa nám zakóduje cez H neuróny. Teda proces z V->H nazývame kódovanie a z H->V nazývame dekódovanie alebo rekonštrukcia.

9.1.4 Ohraničené Boltzmanove stroje ako základ riešenia hĺbkého učenia

Ohraničené Boltzmanove stroje sú podobné štruktúry ako HS resp. Boltzmanov stroj (Restricted Boltzman machines RBM). Rozdiel je hlavne v tom že je organizovaný do vrstiev.

Majme nasledovnú matematickú štruktúru neurónovej siete zloženu z niekoľkých autoencoderov do tzv. Stacku (zasobníka). Porovnajme to s klasickým multilayer perceptrónom (MLP), ktorý je priloženom obrázku taktiež.



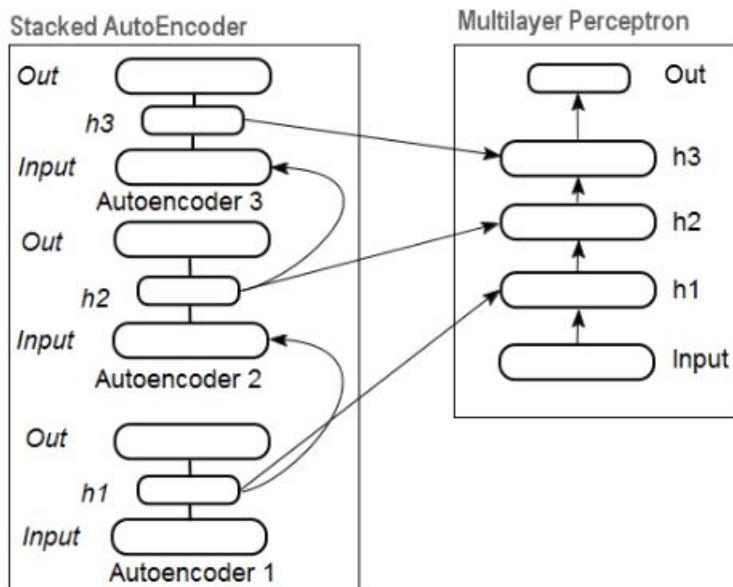
Obr. 9.4: Jednoduchá štruktúra RBM s obojstrannými prepojeniami

Ked' teda na horeuvedenej štruktúra nájdeme také obojstranné prepojenia ktoré z V neurónov budú smerovať ku H neurónom a z H neurónom

ku V neurónom tak pomocou učenia rovnakého ako v Boltzmanom stroji vlastne dostaneme systém, ktorý je funkčne podobný tzv. nekontrolovanému BP učeniu. Pri spomínanom učení sme na vstup dávali vstup X a na výstupe očakávali to isté ale na skrytej vrstve sme vedeli nájsť zakódované vstupy ktoré sme späť rozkódovali na výstupe.

To isté je teda aj v tomto prípade, len by sme učili pomocou Hebového učenia a pomocou Boltzmanovej funkcie, ktorá popisuje Boltzmanove pravdepodobnosťné rozdelenie, ktoré vlastne riadi učenie a zabezpečí konvergenciu učenia. Teda pôjde o proces kódovania vstupov do zhustenej informácie a následne spadne rozkódovanie tejto zhustenej informácie do pôvodného stavu. Tento funkčný model RBM nazývame encoder-decoder modul resp. tzv **AutoEncoder**.

Hintom pri riešení problému hlbokých štruktúr prišiel návrhom poskladať takéto autoencodery do hlbokej štruktúry. Základná myšlienka je na nasledovnom obrázku.



Obr. 9.5: Jednoduchá štruktúra RBM-stack a porovnanie s MLP

Z Obrázku je jasné že zakodovaná informácia z $h1$ je vstupom do Autoencodera 2 a následne vznika zakódovaná informácia zakódovanej informácie na vstupe do Autoencodera 3. Čo je zaujímavé na tom, je že vieme z výstupu Autoencodera 3 späť teda robíme v princípe so zakódovaným vstupom.

9.1.5 Metóda spätného šírenia chyby (BP) na RC NN

Zovšeobecnenie metódy BP na RC NN bolo prvýkrát prezentované v [?]. Logika celého procesu BP je veľmi podobná ako pre FF NN. Pri RC NN nás však do veľkej miery zaujíma dynamika činnosti RC NN z dôvodu existencie rekurentných synapsií. Už pri šírení signálu sa tu vytvárajú podmienky GS a až potom sa môže šíriť späť samotná chyba. Pod dynamikou NN systému budeme rozumieť funkciu

$$D(x(t)) = \frac{\partial x(t)}{\partial t}, \quad (9.37)$$

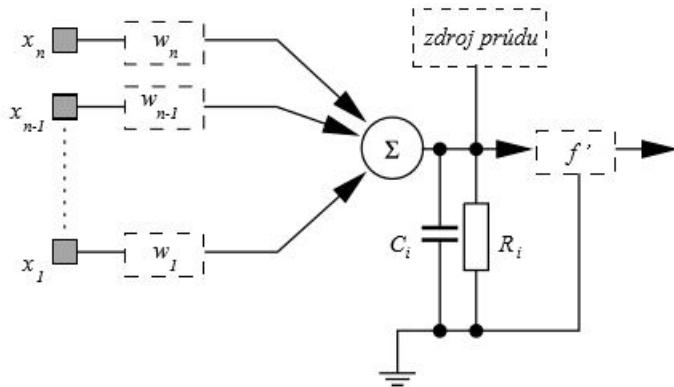
kde $x(t)$ je stav neurónu v čase t . Teda v prípade metódy BP na RC NN hľadáme také adaptačné pravidlo zmeny SV, ktoré :

- bude minimalizovať chybu v zmysle BP prístupu
- bude mať podmienky na dosiahnutie stabilného stavu v bode $x^*(t)$ kedy je $D(x^*(t)) = 0$.

Teda **hlavným rozdielom** medzi BP na RC NN a FF NN je problém neuro dynamiky. Správanie sa RC NN musí byť stabilné pri šírení signálu **dopredu ako aj pri spätnom šírení chyby**. Inak môžu vzniknúť **nekontrolované procesy** a celý systém bude nefunkčný. Je zrejmé, že exaktný popis modelu neurónu a jeho dynamiky bude mať význam pri štúdiu dynamiky celého systému. V literatúre tykajúcej sa NN sa používa tzv. **Ressistance-Capacity model** neurónu ako východisko pre popis RC NN viď obrázok 9.6.

Ak tento model budeme považovať za jednoduchý elektrický obvod, potom v bode \mathbf{V} bude platiť Kirchhoffov zákon o súčte prúdov. Tento model má aj svoju analógiu v biologických systémoch, kde

- w_{ij} vodivosť medzi neurónmi i a j
- $x_i(t)$ priemerný potenciál v miestach zakončení synapsí
- R_i odpor neurónovej membrány neurónu i
- C_i kapacita neurónovej membrány neurónu i
- $f(in_i(t))$ je prahová funkcia neurónu i



Obr. 9.6: Ressistance-Capacity model neurónu

Teda potom podľa Kirchoffovho zákona dostaneme :

$$C_i \frac{\partial x_i(t)}{\partial t} + \frac{x_i(t)}{R_i} = \sum_j w_{ij} x_j(t) + \theta_i \quad (9.38)$$

kde $x_i(t) = f_i(t)$. Táto rovnica vlastne **determinuje** podmienky rovnováhy v neuróne, t.j. $D(x_i(t)) = 0$ na uvedenom modeli neurónu. Rovnicu (9.38) je možné upraviť do tvaru (využijeme fyzikálne zákony)

$$\tau \frac{\partial x_i(t)}{\partial t} = -x_i(t) + \sum_j w_{ij} x_j(t) + \theta_i \quad (9.39)$$

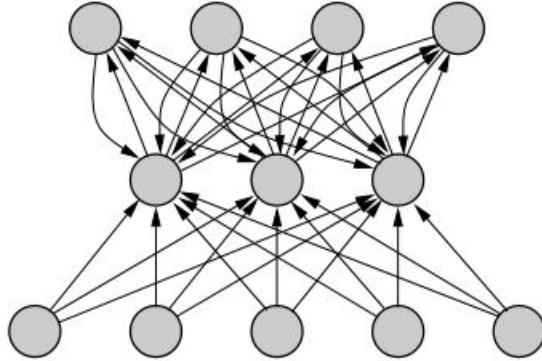
kde τ je funkcia R, C . Ešte raz je potrebné zdôrazniť, že (9.39) predstavuje podmienky takej zmeny stavu neurónu x_i , aby sa sieť dostala do stavu GS. Riešením tejto diferenciálnej rovnice je bod stability $x^*(t)$ v stavovom priestore.

Ak sa teraz vrátíme k reálnej topológií RC NN na obr. 9.7, tak celkovo môžeme vyjadriť vstup do ľubovoľného neurónu ako

$$i_i(t) = \sum_j w_{ij} x_j(t) + \theta_i \quad (9.40)$$

kde θ_i je vstup do neurónu z externého sveta. Ďalej môžeme definovať chybový rozdiel ako

$$er_i(t) = \begin{cases} \text{pre vstupný neurón} \\ 0 \\ \text{inak} \end{cases} \quad (9.41)$$



Obr. 9.7: Topológia RC NN

kde $x_i^*(t)$ je spomínaný stav výstupného neurónu i pri dosiahnutí **rovnovážneho stavu** a $ev_i(t)$ je očakávaná hodnota neurónu v zmysle kontrolovaného učenia. Teda, ako u klasickej metódy BP deklarujeme chybovú funkciu v tvare

$$J(t) = \frac{1}{2} \sum_k e_k(t)^2 \quad (9.42)$$

kde k sa mení od 1 po počet neurónov vo výstupnej vrstve. Vychádzame zo všeobecného výrazu pre adaptačné pravidlo pre zmenu váhy medzi ľubo-volnými neurónmi „s“ a „r“:

$$\Delta w_{rs} = -\gamma \frac{\partial J}{\partial w_{rs}} \quad (9.43)$$

Odvodenie adaptačného pravidla

Postupujeme nasledovne:

1. môžeme vyjadriť, že

$$\frac{\partial J(t)}{\partial w_{rs}} = 0.5 \sum_k \frac{\partial (ev_k(t) - x_k^*(t))^2}{\partial w_{rs}} = -e_i \sum_k \frac{\partial x_k^*(t)}{\partial w_{rs}} \quad (9.44)$$

teda

$$\Delta w_{rs} = \gamma e_i \sum_k \frac{\partial x_k^*(t)}{\partial w_{rs}} \quad (9.45)$$

týmto sa náš problém zúžil na výpočet $\frac{\partial x_k^*}{\partial w_{rs}}$

2. vieme, že všeobecne môžeme napísat

$$x_i^*(t) = f(i_i(t)) \quad (9.46)$$

kde

$$i_i(t) = \sum_j w_{ij} x_j^*(t) + \theta_i \quad (9.47)$$

z toho ale vyplýva, že

$$\frac{\partial x_i^*(t)}{\partial w_{rs}} = \frac{\partial x_i^*(t)}{\partial i_i(t)} \frac{\partial i_i(t)}{\partial w_{rs}} \quad (9.48)$$

3. prvý člen rovnice (9.48) z pravej strany je zrejmý lebo

$$\frac{\partial x_i^*(t)}{\partial i_i(t)} = f'(i_i(t)) \quad (9.49)$$

4. druhý člen v (9.48), môžeme pomocou (9.47) upraviť:

$$\frac{\partial i_i(t)}{\partial w_{rs}} = \sum_j \left\{ \frac{\partial w_{ij}}{\partial w_{rs}} x_j^*(t) + w_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} \right\} \quad (9.50)$$

kde prvý člen pravej strany, vzhľadom na to, že SV sú od seba nezávislé, môžeme vyjadriť pomocou indexovej funkcie Kronekerovo-delta² v tvaru δ_{ij}

$$\frac{\partial w_{ij}}{\partial w_{rs}} = \delta_{ir} \delta_{js} \quad (9.51)$$

kde potom $\sum_j \delta_{js}$ pre $j = s$ je rovné 1 a pre všetky ostatné je 0. Vz»ah (9.50) môžeme prepísať do tvaru

$$\frac{\partial i_i(t)}{\partial w_{rs}} = \delta_{ir} x_s^*(t) + \sum_j w_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} \quad (9.52)$$

5. teraz môžeme pomocou indexovej funkcie δ_{ij} napísat

$$\frac{\partial x_i^*(t)}{\partial w_{rs}} = \sum_j \delta_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} \quad (9.53)$$

² $\delta_{ij} = \begin{cases} 1 & \text{pre } i = j \\ 0 & \text{ak } i \neq j \end{cases}$

Ked'že $x_i = f(i_i)$, tak môžeme pomocou predchádzajúceho vzorca napísat, že

$$\sum_j \delta_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} = \frac{\partial x_i^*(t)}{\partial i_i(t)} \frac{\partial i_i(t)}{\partial w_{rs}} \quad (9.54)$$

Pravú stranu tejto rovnice môžeme prepísať za pomoci (9.49) a (9.52) do tvaru

$$\sum_j \delta_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} = f'(i_i(t)) \left\{ \delta_{ir} x_s^* + \sum_j w_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} \right\} \quad (9.55)$$

po úprave tohto vzorca dostaneme nasledovný výraz

$$\delta_{ir} f'(i_i(t)) x_s^* = \sum_j \underbrace{\left(\delta_{ij} - f'(i_i(t)) w_{ij} \right)}_{L_{ij}} \frac{\partial x_j^*(t)}{\partial w_{rs}} \quad (9.56)$$

kde teda môžeme skrátene napísať

$$\sum_j L_{ij} \frac{\partial x_j^*(t)}{\partial w_{rs}} = \delta_{ir} f'(i_i(t)) x_s^*(t) \quad (9.57)$$

kde $j = 1, \dots, N$. Odtiaľ potom môžeme prejsť k vektorovému vyjáreniu pomocou vektorov \mathbf{L} a \mathbf{x}^* a to nasledovne

$$\mathbf{L} \frac{\partial \mathbf{x}^*(t)}{\partial w_{rs}} = \begin{bmatrix} \delta_{1r} f'(i_1) \\ \delta_{2r} f'(i_2) \\ \vdots \\ \vdots \\ \delta_{Nr} f'(i_N) \end{bmatrix} x_s^*(t) \quad (9.58)$$

z (9.58) môžeme vyjadriť $\frac{\partial \mathbf{x}^*(t)}{\partial w_{rs}}$:

$$\frac{\partial \mathbf{x}^*(t)}{\partial w_{rs}} = \mathbf{L}^{-1} \begin{bmatrix} \delta_{1r} f'(i_1) \\ \delta_{2r} f'(i_2) \\ \vdots \\ \vdots \\ \delta_{Nr} f'(i_N) \end{bmatrix} x_s^*(t) \quad (9.59)$$

Teraz keď sa vrátíme ku skalárnemu vyjadreniu pre "kty riadok dostaneme nasledovný výraz

$$\frac{\partial x_k^*(t)}{\partial w_{rs}} = \mathbf{L}_{kr}^{-1} f'(i_r(t)) x_s^*(t) \quad (9.60)$$

kde \mathbf{L}_{kr}^{-1} je kr -tý element v inverznej matici. Teda konečne môžeme napísť pomocou (9.45), že

$$\Delta w_{rs} = \gamma \sum_k e_i \mathbf{L}_{kr}^{-1} f'(i_r(t)) x_s^*(t) \quad (9.61)$$

Vzťah (9.61) môžeme prezentovať v obvyklom zápise pre BP v tvare³

$$\Delta w_{rs} = \gamma \delta_r x_s^*(t) \quad (9.62)$$

kde

$$\delta_r = f'(i_r) \sum_k e_i \mathbf{L}_{kr}^{-1} \quad (9.63)$$

Nevýhodou výrazu (9.62) je, že vyžaduje výpočet \mathbf{L}^{-1} , čo je globálna a nie lokálna operácia. Toto je možné vyriešiť nasledovným postupom

1. vo vzťahu (9.63) zavedieme substitúciu teda

$$\delta_r^* = f'(i_r(t)) y_r^*(t) \quad (9.64)$$

kde je zrejmé, že $y_r^*(t)$ je vlastne (z (9.41))

$$y_r^*(t) = \sum_k e_i \mathbf{L}_{kr}^{-1} \quad (9.65)$$

2. teraz keď vzorec (9.65) upravíme, resp. ho rozpíšeme a vyberieme z neho sadu rovníc po eliminácii (\mathbf{L}^{-1}), tak potom dostaneme e_i :

$$e_i(t) = \sum_r \mathbf{L}_{ri} y_r^*(t) \quad (9.66)$$

Ak zo vzťahu (9.56) vyberieme časť pre \mathbf{L} , tak potom

$$e_i(t) = \sum_r \left\{ \delta_{ri} - f'(i_r(t)) w_{ri} \right\} y_r^*(t) \quad (9.67)$$

³Pozor na rozdiel medzi indexovou funkciou δ_{ij} a chybovým signálom δ_r

3. ale vieme, že

$$\sum_r \delta_{ri} y_r^*(t) = y_i^*(t) \quad (9.68)$$

a teda vzťah (9.67) môžeme prepísať do tvaru

$$e_i = y_i^*(t) - \sum_r f'(i_r(t)) w_{ri} y_r^*(t) \quad (9.69)$$

Ak zameníme index "r" za "j", môžeme dostať po úprave tvar

$$0 = -y_i^*(t) + \sum_j f'(i_j^*(t)) w_{ji} y_j^*(t) + e_i(t) \quad (9.70)$$

4. ak predpokladáme, že skutočne $y_i^*(t)$ je bodom stability⁴ tak potom musí platiť $\frac{\partial y_i(t)}{\partial t} = 0$, resp. potom

$$\frac{\partial y_i(t)}{\partial t} = -y_i(t) + \sum_j f'(i_j^*) w_{ji} y_j(t) + e_i(t) \quad (9.71)$$

Rovnica (9.71) vyjadruje neurodynamiku systému, a preto tam je $y_i(t)$ a nie ako v rovnici (9.70) $y_i^*(t)$.

Teda ak rovnica (9.39) vyjadrovala formu dopredného šírenia cez synapsie w_{ij} tak vzťah (9.71) vyjadruje dynamiku spätného šírenia chyby po synapsii w_{ji} .

Ak porovnáme tieto vzťahy, tak zistíme, že w_{ij} bol nahradený vo vzťahu (9.71) výrazom $f'(i_j(t))w_{ji}$. Teda celkový postup môžeme zhrnúť do nasledovných bodov:

- pre vstup do NN je nutné určiť jednotlivé θ_i a potom stabilizovať RC NN v zmysle (9.39) výpočtom pevných bodov $\mathbf{x}^*(\mathbf{t})$
- vypočítame chybový signál e_i podľa (9.41)
- vypočítame zo vzťahu (9.71) pevný bod $\mathbf{y}^*(\mathbf{t})$ pre stabilné spätné šírenie chyby
- potom na základe (9.62) a (9.64) vypočítame nové hodnoty zmien SV Δw_{rs} .

Z hľadiska porovnania autor v [?] komentuje zvýšenie účinnosti učenia a skrátenie času pri použití RC NN. Na druhej strane BP na RC NN je citlivejší na zmenu parametrov samotného učenia γ .

⁴ $y_i^*(t)$ je odvodený od $x_i^*(t)$ cez vzťah (9.65)

9.2 Nekontrolované učenie na RC NN

Ďalšia skupina metód, ktorú charakterizuje nekontrolované učenie na RC NN, je skupina metód typu bf Adaptive Resonance Theory . (Ďalej ART)⁵.

Vo všeobecnosti ich môžeme charakterizovať ako metódy s cieľom zhľukovania dát. Tento prístup bol ovplyvnený samotným konkurenčným učením a impulzom vyplývajúcim zo sledovania biologických systémov. Problémom zostáva neexistencia dôkazu GS pri konkurenčnom učení, čo môže spôsobiť problémy v oblasti stability. Určitým východiskom je riešenie problému postupnou redukciou učiaceho parametra na $\rightarrow 0$. Potom určite nastane GS, ale na druhej strane NN nebude tak adaptívna⁶, teda nebude vedieť reagovať na nové typy dát. Tento problém prvýkrát zdôraznil Stefan Grossberg a nazval to **dilema stability-plasticity**.

A práve prístup ART, kde **ressonance** znamená súlad medzi vstupom a doteraz už identifikovanými zhľukmi, sa pokúša problém dejako vyriešiť. Niekoľko sa v literatúre u týchto metód stretávame s prílastkom **stabilné metódy** konkurenčného učenia. Bola vyvinutá metóda ART1 pre binárne vstupy, ďalej jej analógová verzia ART2, ako aj ďalšia verzia ARTMAP, ktorá predstavuje **kontrolované učenie** typu ART.

9.2.1 Teoretický popis metód Adaptive Resonance Theory (ART)

Výraznou črtou ľudskej pamäti je jej schopnosť učiť sa nové veci bez toho, aby zabúdala predtým nadobudnuté vedomosti. Bolo by veľmi výhodné, ak by sa podarilo vytvoriť umelé neurónové siete s podobnými vlastnosťami. Ale väčšina dosiaľ používaných neurónových sietí túto schopnosť nemá a skôr má sklon zabúdať staré znalosti v prípade, že sa pokúšame doplniť nejaké znalosti nové. Napríklad, ak sme siet typu FF NN s učiacim algoritmom backpropagation naučili 10 vecí a chceme doplniť jednu novú, siet musí byť trénovaná odznova použitím všetkých 11 vzoriek.

Tento problém je potrebné riešiť, v prípade, že chceme vytvoriť systém schopný autonómne sa prispôsobiť nečakaným zmenám vonkajšieho prostredia v reálnom čase. Ukázalo sa, že systém vyhovujúci pre daný problém by mal optimálne riešiť spomínanú dilemu stability a plasticity.

Vlastnosti stability a plasticity môžu byť definované takto:

⁵V NN literatúre sa nazývajú aj Cluster discovery networks - Siete pre vyhľadávanie zhľukov

⁶plastická

Plasticita: Systém v priebehu učenia generuje rozpoznávacie kódy ako odozvu na postupnosť vstupných vzoriek z okolitého prostredia. Pritom sa postupne vytvárajú takzvané vzorky kritických črt, čo znamená, že každý rozpoznávací kód si ponechá len tie črty, ktoré ho odlišujú od kódov ostatných tried.

Stabilita: V priebehu učenia sa vytvárajú stabilné stavy, ktoré zabezpečujú priamu odozvu na už známu vzorku, pričom sa vnútorme rozhoduje o tom, či sa pre prezentovanú vzorku vytvorí nová kategória, alebo či bude iba upravený kód niektornej už existujúcej kategórie.

Potom dilema stability a plasticity môže byť popísaná sériou otázok:

- Ako zabezpečiť, aby systém zostal dostatočne adaptívny (plastický) v odozve na výrazne novú vstupnú vzorku, a pritom bol nemenný (stabilný) v prípade, že vstup je nedôležitý?
- Ako má systém vedieť, kedy sa má prepniť do stabilného a kedy do plastického módu, tzn. ako dosiahnuť stabilitu bez neflexibilnosti a plasticitu bez vzniku chaosu v systéme?
- Ako zaistiť, aby si systém zachoval skôr naučené vedomosti a zároveň sa učil nové veci?

Názov ART súvisí so spôsobom, akým v sieti prebieha učenie a hľadanie víťaza. Vstupná informácia osciluje vo forme výstupných hodnôt medzi dvomi vrstvami neurónov, až dokiaľ sa nedosiahne rezonancia. V tom momente nastáva učenie, čiže adaptácia váh. Rezonancia môže nastáť dvoma rôznymi spôsobmi, a to v prípade, že siet už v minulosti spracovávala takú istú alebo veľmi podobnú vzorku, vtedy rezonancia nastáva okamžite; alebo v prípade, že vstupná vzorka je odlišná od všetkých predošlých, spúšťa sa proces prehľadávania naučených kódov a porovnáva sa ich podobnosť s prezentovanou vzorkou, pričom je definovaná určitá prahová hodnota, ktorá určuje minimálnu prípustnú podobnosť víťaznej triedy. Ak tento prah nespĺňa žiadna z už známych tried, systém vytvorí triedu novú, identickú s prezentovanou vzorkou. Takto sa zároveň dosahuje stabilita (siet rezonuje v prípade známeho vstupu) aj plasticita (zachovanie schopnosti učiť sa nové neznáme vzorky).

9.2.2 Siete triedy ART pre spracovanie binárnych dát

Teória sietí ART bola pôvodne vypracovaná iba pre binárne dátá. Prvá z tejto triedy sietí sa dnes nazýva ART1 a slúži pre nekontrolované učenie. Jej roz-

šírená modifikácia pre kontrolované učenie sa nazýva ARTMAP a umožňuje kontrolované učenie klasifikačných tried s možnosťou zovšeobecnenia v množine vzoriek ako aj v množine tried.

9.2.3 Štruktúra a dynamika sietí ART1

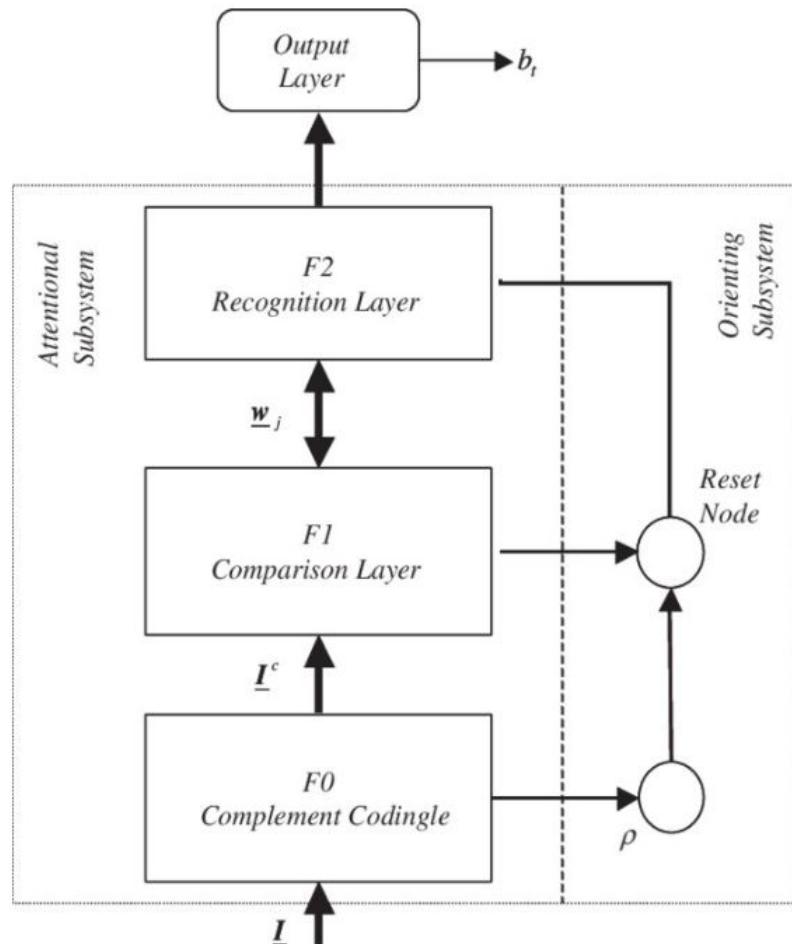
Základná architektúra sietí ART1 je tvorená troma skupinami neurónov (viď Obr. 9.9):

1. Vstupná vrstva (označovaná F_0) slúži iba na zaznamenanie hodnôt vstupnej vzorky.
2. Porovnávacia vrstva (označovaná F_1), v ktorej sa zistuje, či je víťazná kategória dostatočne podobná vstupnej vzorke.
3. Rozpoznávacia vrstva (označovaná F_2), aktivácia neurónov ktorej určuje víťaznú triedu v danom kroku hľadania.

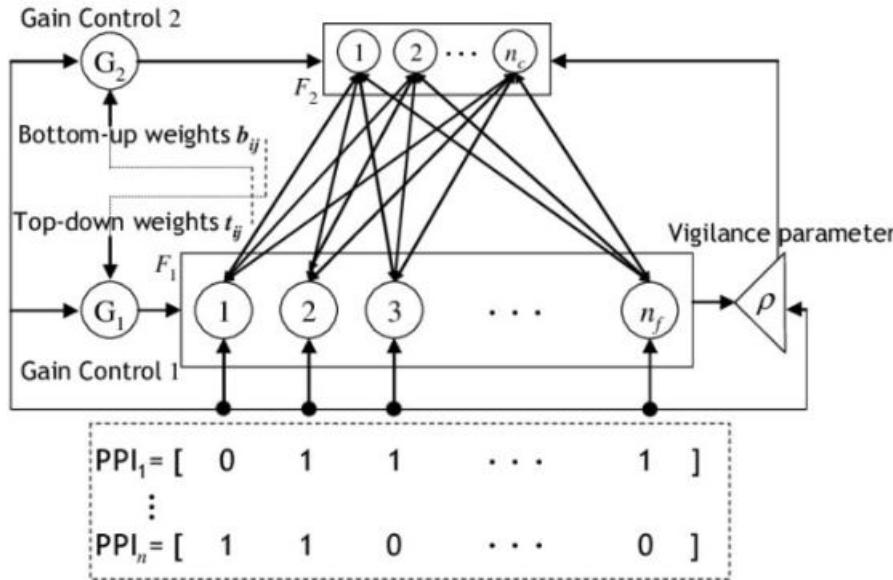
Vrstvy F_1 a F_2 tvoria tzv. krátkodobú pamäť (STM - short term memory), pretože sa v nich uchovávajú informácie týkajúce sa iba jednej prezentácie vzoriek, zatiaľ čo linky spájajúce tieto dve vrstvy úplným obojsmerným prepojením, tvoria dlhodobú pamäť (LTM - long term memory), keďže uchovávajú informácie o všetkých doteraz naučených kategóriách. Medzi vrstvami F_1 a F_2 oscilujú signály vyvolané prezentáciou vzorky na vstupe, a v prípade, že porovnávacia vrstva vyhodnotí, že odozva rozpoznávacej vrstvy reprezentuje prezentovanú vzorku dostatočne, nastáva rezonančný stav a následne adaptácia liniek medzi víťazným rozpoznávacím neurónom a všetkými neurónmi v komparačnej vrstve F_1 .

Okrem týchto troch vrstiev neurónov (F_0 , F_1 a F_2) sieť obsahuje niekoľko ďalších špeciálnych neurónov slúžiacich k riadeniu jej činnosti. Neuróny ρ a $gain$ slúžia výlučne k synchronizácii siete. Ich úlohou je rozhodovať, kedy sa môžu aktivovať vrstvy F_1 a F_2 . Riadiaci neurón *Reset* má úlohu definovanú priamo v algoritme učenia siete. Tento neurón rozhoduje na základe porovnania vstupnej a komparačnej vrstvy o tom, či nájdená víťazná kategória v rozpoznávacej vrstve splňa definované kritéria. V prípade, že kritérium podobnosti splnené nebolo, signál z tohto neurónu zmrazí aktuálneho víťaza vo vrstve F_2 na celý zvyšok prezentácie danej vzorky, čím umožní pokračovať v hľadaní lepšej triedy, prípadne vytvoriť triedu novú.

Základnou bázou, na ktorej sú siete ART postavené, je konkurenčné učenie (competitive learning), ktoré sa uskutočňuje medzi vrstvami F_1 a F_2 . Neuróny vrstvy F_2 majú medzi sebou okrem vstupov z vrstvy F_1 aj záporné prepojenia a kladnú spätnú väzbu na seba samých, čo spôsobí, že neurón,



Obr. 9.8: Architektúra neurónovej siete ART1



Obr. 9.9: Proces učenia v neurónovej sieti ART1

ktorý má najväčšiu hodnotu aktivácie, potláča aktiváciu ostatných, až kým sa po niekoľkých propagáciach aktivácia všetkých ostatných neurónov neustáli na nule. Linky vedúce k víťaznému neurónu sa potom nastavia na takú hodnotu, že pri opakovanej prezentácii danej vzorky bude aktivácia víťaza ešte väčšia. Takéto správanie sa siete sa nazýva tiež **vítaz berie všetko (winner takes all)**.

Učiaci cyklus sa začína spracovaním vstupnej vzorky vo vrstve F_0 . Neurón ρ v ďalšom kroku zabezpečí, že vzorka sa nezmenená dostane do porovnávacej vrstvy. Tento neurón má jednotkovú hodnotu na výstupe iba v prípade, že na vstupe je nenulová hodnota a zároveň nie je aktivovaný ani jeden neurón vo vrstve F_2 . A keďže neuróny vo vrstve F_1 sa aktivujú podľa tzv. *Pravidla 2/3*, ktoré vrávajú že neurón je aktívny práve vtedy, ak sú aktívne dva z jeho troch vstupov (ρ, F_0, F_2), za daného stavu (vrstva F_2 neaktívna, ρ aktívny, vrstva F_0 obsahuje vstup) sa vstup nezmenený skopíruje do vrstvy F_1 .

Vrstva F_1 je v smere zdola nahor plne prepojená s vrstvou F_2 . Týmito prepojmi je vzorka propagovaná. Ďalej k porovnávacím neurónom. A práve tu nastáva konkurenčné učenie, takže sa nájde práve jeden víťazný neurón, ktorého výstupná hodnota sa nastaví na 1, výstupné hodnoty ostatných sa vynulujú.

Týmto bola predbežne určená trieda, o ktorej sieť „predpokladá“, že by do nej mala byť zatriedená prezentovaná vzorka. čiže, ak máme vo vrstve F_2 M neurónov, neaktívnych bude práve $M - 1$ z nich, a aktívny bude iba jediný víťaz. Táto aktivačná vzorka je späť propagovaná cez linky spájajúce vrstvy F_2 a F_1 zhora nadol.

V tomto momente je už ale výstupná hodnota neurónu ρ nulová, keďže vo vrstve F_2 je už jeden neurón aktívny. Takže vo vrstve F_1 budú teraz aktivované iba tie neuróny, ktoré získali pozitívny signál zároveň od im zodpovedajúceho vstupného neurónu, ako aj od víťaza vo vrstve F_2 .

Na začiatku učenia sú váhy všetkých liniek inicializované na hodnotu 1. To znamená, že ak v rozpoznávacej vrstve vyhral predtým nepoužitý neurón, v porovnávacej vrstve sa opäť objaví nezmenená vstupná vzorka.

V prípade, že víťaz F_2 je už naučený na klasifikáciu nejakej inej vzorky, sú váhy niektorých od neho vedúcich liniek nulové, čo znamená, že zodpovedajúce neuróny vo vrstve F_1 sa neaktivujú, aj keď zo vstupu k nim prichádza jednotkový signál. Z toho vyplýva, že vektor reprezentovaný porovnávacou vrstvou tvorí vždy podmnožinu vstupného vektora.

Teraz musí sieť rozhodnúť o tom, či predpokladaná trieda skutočne klasifikuje prezentovanú vzorku. To sa deje porovnaním podobnosti vektorov F_0 a F_1 .

Ak sa zistí, že zhoda vektorov F_0 a F_1 je dostatočne veľká, nasleduje úprava váh medzi vrstvami F_1 a F_2 . Váhy liniek smerujúcich do F_2 sa nastavia tak, aby bola pri opakovanej prezentácii danej vzorky víťazná trieda zvolená priamo. Dole smerujúce linky sa nastavia na rovnaké hodnoty, aké majú aktivácie neurónov vo vrstve F_1 . To znamená, že do týchto váh sa uloží vzorka tvoriaca prienik medzi aktuálnou vzorkou a pôvodným klasifikačným kódom víťaznej triedy.

V prípade, že podobnosť vektorov F_0 a F_1 nie je dostatočná, aktivuje sa *Reset*, ktorý spôsobí zmrazenie aktuálneho víťaza na zvyšok prezentácie, čo znamená, že jeho výstupná hodnota ostane až do jej konca nulová. Tým sa sieť dostala do stavu zhodného s počiatočným a celý cyklus sa opakuje aktiváciou neurónu ρ , skopírovaním vstupu do porovnávacej vrstvy atď., až kým sa nenájde zodpovedajúca klasifikačná trieda, ktorá vygeneruje dostatočne podobný porovnávací vektor F_1 , alebo kým sa neodskúšajú všetky možné triedy, a ani jedna z nich nesplní kritérium podobnosti, čo znamená, že k dispozícii nie je dostatočné množstvo tried.

Stručne sa algoritmus učenia ART1 dá zhrnúť do týchto bodov:

1. prezentácia novej vzorky,
2. nájdenie víťaza vo vrstve F_2 ,

3. otestovanie navrhovaného víťaza vo vrstve F_1 ,
4. v prípade, že test zlyhal a vo vrstve F_2 sú ešte neotestované neuróny, zmrazenie víťaza a prechod na bod 2, inak koniec,
5. adaptácia váh medzi vrstvami F_1 a F_2 .

9.2.4 Teória sietí ART1

Zo slovného popisu dynamiky vyplývajú pre aktiváciu jednotlivých vrstiev neurónov tieto vzťahy:

- Neuróny vo vrstve F_1 sa správajú podľa *Pravidla 2/3*, čo sa dá zapísť:

$$x_i^1 = \begin{cases} 1 & \text{ak } u_i + x_1^g + \sum_{j=1}^M x_j^2 w_{ji} > 1 + \tilde{w}, \\ 0 & \text{inak,} \end{cases} \quad (9.72)$$

kde u_i je vstup $F_0 \rightarrow F_1$, x_1^g je výstupná hodnota neurónu ρ a $\sum x_j^2 w_{ji}$ je suma výstupných hodnôt x_j^2 vo vrstve F_2 vážená trénovateľnými váhami w_{ji} . \tilde{w} je konštanta z intervalu

$$0 < \tilde{w} < 1. \quad (9.73)$$

- Neurón ρ sa aktivuje podľa nasledovného pravidla:

$$x_1^g = \begin{cases} 1 & \text{ak } \sum_i u_i > 0 \text{ a } \sum_j x_j^2 \leq 0, \\ 0 & \text{inak,} \end{cases} \quad (9.74)$$

to znamená, že k tomu aby bolo x_1^g aktívne, musia byť všetky neuróny vo vrstve F_2 neaktívne a na vstupe musí byť nenulová hodnota.

Takže, ak je F_2 neaktívne, rovnica (9.72) sa redukuje na:

$$x_i^1 = \begin{cases} 1 & \text{ak } u_i = 1, \\ 0 & \text{inak,} \end{cases} \quad (9.75)$$

Inak je aktívny práve jeden neurón x_j^2 vo vrstve F_2 . Vtedy sa rovnica (9.72) zmení na:

$$x_i^1 = \begin{cases} 1 & \text{ak } u_i = 1 \text{ a } w_{ji} > \tilde{w}, \\ 0 & \text{inak.} \end{cases} \quad (9.76)$$

- Pre definíciu aktivácie neurónov vo vrstve F_2 sa zavádzajú premenná T_j , ktorá udáva vstup zo siete do j -tého neurónu, takto:

$$T_j = \frac{\sum_{i=1}^N x_i^1 w_{ij}}{\alpha + \sum_{i=1}^N w_{ij}}, \quad (9.77)$$

kde w_{ij} sú váhy $F_2 \rightarrow F_1$. Potom, za predpokladu, že existuje také j , že $T_j > 0$, ako víťazná bude zvolená trieda J , pre ktorú platí:

$$T_J = \max_{1 \leq j \leq M} \{T_j\}. \quad (9.78)$$

Takto by sa mohlo stať, že zároveň niekoľko neurónov vo vrstve F_2 nadobudne rovnakú maximálnu hodnotu:

$$T_{J_1} = T_{J_2} = \dots = T_{J_k} = \max_{1 \leq j \leq M} \{T_j\},$$

čo by znamenalo nejednoznačného víťaza a nasledujúci postup by sa značne skomplikoval. Preto sa vždy uvažuje, že víťaz je jednoznačný, a to ten s najmenším indexom.

- Neurón $Zisk2$ má funkciu analogickú s neurónom ρ , a to blokovať všetky neuróny vo vrstve F_2 . V priebehu učenia ART1 je dôležité, aby bola táto podmienka platná iba pri prezentácii novej vzorky, takže neurón $Zisk2$ sa aktivuje práve len v tomto momente.

Predtým, než budú popísané pravidlá pre učenie, si definujeme význam označenia, ktoré je Ďalej používané. Ak \mathbf{a} je binárny vektor s L zložkami, čiže $\mathbf{a} = (a_1, a_2, \dots, a_L)$, potom *norma* tohto vektora je definovaná nasledovne:

$$\|\mathbf{a}\| = \sum_{i=1}^L a_i, \quad (9.79)$$

čo značí, že $\|\mathbf{a}\|$ udáva počet jednotiek vo vektore \mathbf{a} . Ďalej si definujeme *prieknik dvoch vektorov* \mathbf{a} a \mathbf{b} ako vektor $\mathbf{c} \equiv \mathbf{a} \cap \mathbf{b}$, kde

$$c_i = 1 \Leftrightarrow a_i = 1 \text{ a } b_i = 1 \quad (9.80)$$

a na záver pojem *podmnoiny vektora* ako

$$\mathbf{a} \subset \mathbf{b} \Leftrightarrow \mathbf{a} \cap \mathbf{b} = \mathbf{a}. \quad (9.81)$$

Potom pre učenie platí:

- Váhy $F_2 \rightarrow F_1$ sa menia podľa pravidiel, ktoré boli uvedené už v predošej časti tejto kapitoly. Teraz ich rozšírime len o matematický popis. Pri inicializácii sa váhy všetkých týchto liniek nastavia na ich maximálnu hodnotu, tzn.:

$$\mathbf{w}_{ji}(0) = 1 \quad \forall 1 \leq j \leq M, 1 \leq i \leq N. \quad (9.82)$$

Úprava váh nastáva vždy až potom čo bol určený víťaz y_J vo vrstve F_2 a menia sa iba váhy liniek vedúcich z tohto neurónu. Formulovať to môžeme takto:

$$\frac{dw_{ij}}{dt} = x_j^2(x_i^1 - w_{ij}(t-1)), \quad (9.83)$$

z čoho vyplýva, že zmena sa deje prvýkrát iba ak daná linka vychádza z víťazného neurónu a druhýkrát iba ak existuje rozdiel medzi predtým nastavenou váhou linky a terajšou aktiváciou i -tého neurónu x_i . A keďže nemôže nastať prípad, že $x_i = 1$ a zároveň $w_{ji} = 0$, váha w_{ji} sa buď nemení, alebo sa mení z 1 na 0, čím sa zabezpečuje stabilita systému. Zmenu váhového vektora víťazného neurónu potom môžeme zapísat:

$$\mathbf{w}_J(t+1) = \begin{cases} \mathbf{u} \cap \mathbf{w}_J(t) & \text{ak } x_J^2 \text{ už niekedy vyhral,} \\ \mathbf{u} & \text{ak } x_J^2 \text{ vyhral prvýkrát.} \end{cases} \quad (9.84)$$

- Váhy $F_1 \rightarrow F_2$ majú dôležitejšiu úlohu, pretože sa musia starať aj o to, aby sa nové triedy vytvárali systematicky, t.j. aby sa nepreskakovali niektoré nepoužité neuróny, a tiež aby sa vždy najprv otestovali už použité neuróny a až potom, čo sa tieto ukážu ako nezodpovedajúce kritériám, použije sa nový neurón. Toto je zabezpečené inicializáciou váh, pre ktoré platí:

$$\mathbf{w}_{ij}(0) = \alpha_j \quad \forall 1 \leq j \leq M, 1 \leq i \leq N, \quad (9.85)$$

kde

$$0 < \alpha_M < \alpha_{M-1} < \dots < \alpha_1 \leq \frac{1}{\beta + N}. \quad (9.86)$$

Samotná úprava váh je totožná s úpravou váh $F_2 \rightarrow F_1$ až na to, že tu sú nové váhy ešte normované na hodnotu menšiu ako 1, čiže

$$\mathbf{w}_J(t+1) = \frac{\mathbf{u} \cap \mathbf{w}_J(t)}{\beta + |\mathbf{u} \cap \mathbf{w}_J(t)|}, \quad (9.87)$$

kde β je bezvýznamne malá hodnota.

- Ako posledný je popísaný test hypotézy, tzn. overenie či víťazná kategória určená rovnicami (9.72), (9.78) je uspokojivá z hľadiska parametra podobnosti ρ (vigilance parameter). Neurón *Reset* sa aktivuje, ak platí

$$\text{Reset} \Leftrightarrow \frac{|\mathbf{u} \cap \mathbf{w}_J|}{|\mathbf{u}|} \leq \rho. \quad (9.88)$$

V prípade, že podmienka neplatí, algoritmus sa dostáva do poslednej fázy-úpravy váh, popísanej rovnicami (9.84) a (9.87). Ak podmienka platí, vyšle sa signál reset, ktorý zmrazí aktuálneho víťaza x_J^2 , následkom čoho sa zase aktivuje neurón ρ a celý cyklus sa opakuje.

Z poslednej rovnice vyplýva dôležité pravidlo určujúce, aké hodnoty parametra ρ volíme. Ak dáme $\rho \leq 0$, reset nenastane nikdy, čiže všetky vzorky sú klasifikované do tej istej triedy. Na druhej strane pre $\rho = 1$ každej odlišnej vzorke bude vytvorená nová kategória. Ak by ρ nado-budlo hodnotu väčšiu ako jedna, reset nastane v každom prípade, čo spôsobí zmrazenie siete, a to je nekorektné správanie.

Kapitola 10

Modulárne neurónové siete

Je zrejme, že učenie je z hľadiska svojej dĺžky a presnosti zložitý proces, ktorý narastá so zložitosťou riešeného problému.

Preto ak je niekedy vhodné a možné rozdeliť celkovú úlohu na rad podúloh, tak je vhodné použiť tzv. **modulárne neurónové siete MNN**. Príkladom môže byť napr. problém aproximácie funkcie vyjadrenej vzťahom (10.1), pre ktorú je možné vytvoriť sieť s dvoma modulmi, ktoré sú naučené príslušným časťam tejto funkcie.

$$f(x) = \begin{cases} x & ak x > 0 \\ -x & ak x \leq 0 \end{cases} \quad (10.1)$$

Jedná z definícií MNN môže byť nasledovná:

Neurónové siete sa nazývajú modulárne, ak samotné výpočty v NN je možné dekomponovať do 2 alebo viacerých modulov (sub-systémov), ktoré môžu fungovať bez vzájomnej komunikácie. Výstupy z týchto modulov sú vstupnými signálmi do integrujúceho bránového modulu, ktorý môže mať nasledovné funkcie:

- rozhodne, ktorý z modulov sa bude učiť – ktorými trénovacími vzorkami
- rozhodne, ako budú kombinované vstupy na konečný výstup

Modularita je teda preto chápana ako proces spracovania v zmysle metódy

rozdeľuj a panuj!

Výhody MNN by sme mohli zhŕnúť do týchto bodov:

1. rýchlosť učenia na svojich dátach
2. variabilita učenia, teda každý subsystém môže mať svoje vlastné adaptačné pravidla
3. schopnosť riešiť komplikované problémy efektívnejšie ako jednoduché NN

Pri MNN poznáme dva základne subsystémy a to:

- **expertné siete – EN** (expert networks)
- **bránové siete – GN** (gating networks)

každá z týchto sietí má svoj vlastný adaptačný mechanizmus.

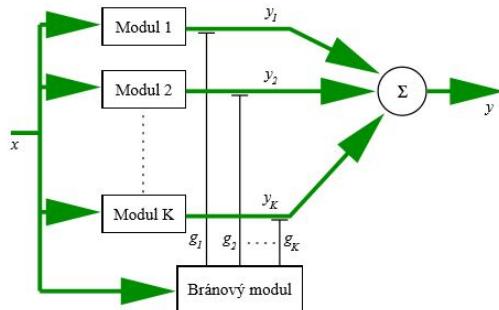
V modulárnej architektúre sa spájajú prvky kontrolovaného a nekontrolovaného učenia.

- *Kontrolované učenie* je charakterizované tým, že každá učiaca vzorka sa skladá z dvoch častí: vstupu a k nemu prislúchajúceho výstupu.
- *Nekontrolované učenie* je charakterizované modulmi, ktoré medzi sebou "súťažia" o možnosť produkovania správneho výstupu. O tom, ktorý modul sa bude učiť jednotlivé vstupné vzorky teda nerozhoduje externý učiteľ, ale schopnosť jednotlivých modulov produkovať výstup, ktorý je najbližšie k požadovanému.

Počas učenia sa uplatňuje efekt *kladnej spätej väzby*. Modul, ktorý "odpoval" najlepšie na daný vstup, prijíma najväčšie množstvo učiacej informácie v etape úpravy hodnôt váh. Moduly, ktoré produkovali horšie výsledky, prijímajú menšie množstvo učiacej informácie. Každý modul prijíma také množstvo učiacej informácie, ktoré je úmerné jeho schopnosti učenia. *Bránový modul* (gating module) plní úlohu kvalifikátora nad celým definičným oborom funkcie. *Expertné moduly* pracujú ako aproximátory čiastkových funkcií v jednotlivých častiach definičného oboru. Vzájomnou konkurenciou sa po skončení učenia dosiahne špecializácia jednotlivých expertných modulov na jednotlivé časti definičného oboru aproximovanej funkcie.

10.1 Štruktúra modulárnej siete

Bloková štruktúra základného modelu modulárnej neurónovej siete je zobrazená na Obr. 10.1. Sieť pozostáva z K expertných modulov a jedného bránového modulu. Všetky moduly sú zložené z jednej skrytej vrstvy. Medzi



Obr. 10.1: Základná bloková schéma modulárnej neurónovej siete pozostávajúcej z K expertných modulov a jedného bránového modulu.

vstupnou vrstvou a vrstvami jednotlivých modulov sú prepojenia typu „full connection“.

Nech sú vstupné vzorky reprezentované vstupným vektorom \mathbf{u} s rozmerom N a výstupným vektorom \mathbf{e} s rozmerom M . Expertné moduly obsahujú q neurónov, bránová sieť obsahuje K neurónov. Každému expertnému modulu prislúcha jeden neurón bránového modulu. Vstupný vektor \mathbf{u} je zo vstupu privedený do všetkých modulov naraz. Každý expertný modul produkuje výstupný vektor \mathbf{y}_i , ktorý má M prvkov. Bránový modul produkuje vektor \mathbf{g} s K prvkami. Nech g_i je výstupná hodnota i -teho neurónu bránového modulu. Celkový výstupný vektor \mathbf{y} sa vypočíta ako súčet výstupov jednotlivých expertných modulov \mathbf{y}_i násobených príslušnými hodnotami g_i .

$$\mathbf{y} = \sum_{i=1}^K g_i \mathbf{y}_i \quad (10.2)$$

Druh úlohy, pre ktorý je určené použitie siete určuje typ aktivačnej funkcie neurónov v expertných sieťach. Pre potreby *regresie* sú určené lineárne aktivačné funkcie, pre potreby *klasifikácie* sú určené nelineárne neuróny so sigmoidálnou aktivačnou funkciou. Zatiaľ bude výklad venovaný opisu štruktúry siete určenej pre regresné úlohy. Výstupy jednotlivých expertných modulov sú zhodné s aktivačnými hodnotami neurónov v týchto moduloch a ich hodnoty výsledkom násobenia vstupného vektora \mathbf{u} vektormi váh \mathbf{w}^i prislúchajúcich k jednotlivým modulom

$$\mathbf{y}_i = \mathbf{u}^T \mathbf{w}^i \quad (10.3)$$

Aktivačná funkcia neurónov bránového modulu je tiež lineárna. Vektor \mathbf{w}^b je vektor váh bránovej siete. Vektor aktivačných hodnôt \mathbf{x} sa vypočíta podľa vzťahu

$$\mathbf{x} = \mathbf{u}^T \mathbf{w}^b \quad (10.4)$$

Kedže pri tvorbe základného modelu bol zvolený *štatistický prístup*¹, výstupné hodnoty g_i neurónov bránovej siete sú potom upravené tak, aby nadobúdali hodnoty z intervalu $<0, 1>$ a ich súčet bol rovný jednej. Táto požiadavka je zabezpečená výstupou funkciou typu *softmax* tvaru

$$g_i = \frac{\exp(x_i)}{\sum_{j=1}^K \exp(x_j)} \quad (10.5)$$

Použitie funkcie softmax v tomto prípade dovoľuje interpretovať hodnoty g_i ako pravdepodobnosti. Hodnota g_i tak určuje apriórnu pravdepodobnosť, že i -ty modul generoval aktuálnu učiacu vzorku.

10.1.1 Pravdepodobnostná funkcia

Neurónovú sieť možno chápať aj ako prostriedok pre štatistické modelovanie a predikciu. Z tohto hľadiska je činnosť naučenej neurónovej siete úspešná, ak modeluje proces, pomocou ktorého boli učiace vzorky generované. Veľkosť chyby v procese učenia preto nemôže byť postačujúcim ukazovateľom kvality naučenia siete.

Najvšeobecnejší a kompletný popis vzoriek učiacej množiny je možný podľa funkcie rozdelenia ich pravdepodobnosti $p(\mathbf{u}, \mathbf{e})$ [?]. Cieľom učiaceho algoritmu aplikovaného na danú architektúru je modelovanie pravdepodobnostného rozdelenia množiny učiacich vzoriek $\{\mathbf{u}, \mathbf{e}\}$ alebo aj maximalizácia funkcie rozdelenia pravdepodobnosti $l = p(\mathbf{u}, \mathbf{e})$. Podľa [?] môže byť maximalizácia logaritmickej pravdepodobostnej funkcie l interpretovaná ako minimalizácia chybovej funkcie J doprednej siete

$$J = -\ln l \quad (10.6)$$

Ak $p(\mathbf{u})$ predstavuje nepodmienenú pravdepodobnosť vstupu a $p(\mathbf{e}|\mathbf{u})$ je podmienená pravdepodobnosť výstupu, ktorá je podmienená vstupným vektorom \mathbf{u} , potom sa funkcia rozdelenia pravdepodobnosti vypočíta ako súčin hodnôt týchto dvoch pravdepodobností

$$p(\mathbf{u}, \mathbf{e}) = \ln p(\mathbf{e}|\mathbf{u})p(\mathbf{u}) \quad (10.7)$$

¹Podobnou tématikou sa zaoberali Szymansk a Lemmon z informačno-teoretického hľadiska.

Pri ďalšom odvodzovaní učiaceho algoritmu je výhodnejšie pracovať s prirodzeným logaritmom výrazu (10.7). Je tak možné urobiť, pretože funkcia logaritmus je monotónne rastúca na celom definičnom obore.

Logaritmická pravdepodobnosť funkcia je definovaná nasledovne:

$$l(\mathbf{w}) = \ln p(\mathbf{e}|\mathbf{u})p(\mathbf{u}) \quad (10.8)$$

Hodnota pravdepodobnostnej funkcie závisí od množiny hodnôt voľných parametrov siete \mathbf{w} (vrátane \mathbf{w}^b). Typ podmienenej pravdepodobnosti $p(\mathbf{e}|\mathbf{u})$ závisí od typu úlohy, pre ktorý je modulárna sieť určená.

- *Regresia:* Učiace údaje sú spojité z oboru reálnych hodnôt. Rozdelenie pravdepodobnosti výstupných hodnôt je možné opísť pomocou Gaussovo rozdelenia pravdepodobnosti

$$p(\mathbf{e}|\mathbf{u}) = \frac{1}{\sqrt[2M]{2\pi} \sqrt{|\Lambda|}} \exp\left(-\frac{1}{2}(\mathbf{e} - \mathbf{y})^T \Lambda^{-1} (\mathbf{e} - \mathbf{y})\right) \quad (10.9)$$

\mathbf{e} - požadovaný výstupný vektor hodnôt

\mathbf{y} - výstupný vektor hodnôt expertnej siete

Λ - matica kovariancií

M - počet neurónov expertnej siete

- *Klasifikácia:* Výstupné hodnoty kvalifikátora nadobúdajú diskrétné hodnoty. Podľa [?] je v takom prípade potrebné pre ich opis použiť Bernoulliho rozdelenie pravdepodobnosti.

$$p(\mathbf{e}|\mathbf{u}) = \prod_{i=1}^M y_i^{e_i} (1 - y_i)^{1-e_i} \quad (10.10)$$

e_i - požadovaná výstupná hodnota i -teho neurónu expertnej siete

y_i - výstupná hodnota i -teho neurónu expertnej siete

M - počet neurónov expertnej siete

Za predpokladu, že kovariančná matica Λ vo výraze (10.9) je jednotková, je možné hodnotu argumentu funkcie \exp v Gaussovskom rozdelení pravdepodobnosti vypočítať ako Euklidovskú normu vektora.

$$p(\mathbf{e}|\mathbf{u}) = \frac{1}{\sqrt[2M]{2\pi}} \exp\left(-\frac{1}{2} \|\mathbf{e} - \mathbf{y}\|^2\right) \quad (10.11)$$

Dosadením výrazu (10.11) do (10.2) pre všetky K expertné siete sa získa vzorec pre výpočet celkovej podmienenej pravdepodobnosti.

$$p(\mathbf{e}|\mathbf{u}) = \frac{1}{\sqrt[|M|]{2\pi}} \sum_{i=1}^K g_i \exp\left(-\frac{1}{2}\|\mathbf{e} - \mathbf{y}_i\|^2\right) \quad (10.12)$$

Dosadením výrazu (10.12) do (10.8) prďzanedbaní konštanty $-\ln \sqrt[|M|]{2\pi}$ sa získa konečný tvar logaritmickej pravdepodobnostnej funkcie pre regresiu.

$$l_R(\mathbf{w}) = \ln \sum_{i=1}^K g_i \exp\left(-\frac{1}{2}\|\mathbf{e} - \mathbf{y}_i\|^2\right) \quad (10.13)$$

Toto rozdelenie pravdepodobnostdžie uvedené v [?] pod názvom *Gaussov zmesový model*. (Gaussian mixture model). Logaritmickú pravdepodobnostnú funkciu pre klasifikáciu možno vyjadriť podobne v tvare

$$l_K(\mathbf{w}) = \ln \sum_{i=1}^K g_i \prod_{j=1}^M y_{ij}^{e_{ij}} (1 - y_{ij})^{1-e_{ij}} \quad (10.14)$$

Pravdepodobostné funkcie (10.13) a (10.14) budú ďalej základom pre odvádzanie učiacich algoritmov pre regresiu a klasifikáciu.

10.2 Učiaci algoritmus pre regresiu

Odvodenie učiaceho algoritmu je založené na metóde najstrmšieho *vzostupu* hodnoty logaritmickej pravdepodobostnej funkcie $l_R(\mathbf{w})$. Cieľom učenia je maximalizácia hodnoty tejto funkcie.

Postup je analogický metóde najstrmšieho zostupu hodnoty chybovej funkcie $J(\mathbf{w})$ doprednej siete. Jej veľkosť závisí od matice \mathbf{w} , ktorá predstavuje množinu voľných parametrov doprednej siete (váh w a príp. aj prahov θ). V procese učenia sa upravujú hodnoty prvkov matice \mathbf{w} tak, aby sa hodnota chybovej funkcie $J(\mathbf{w})$ zmenšovala. Nové hodnoty prvkov matice \mathbf{w} sa v t -tom kroku učiaceho procesu vypočítajú podľa vzťahu

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \Delta \mathbf{w}(t) \quad (10.15)$$

Hodnota $\Delta \mathbf{w}$ sa v t -tom kroku vypočíta podľa vzťahu

$$\Delta \mathbf{w} = -\gamma \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} \quad (10.16)$$

V modulárnej architektúre opísanej v predošej časti predstavujú množinu voľných parametrov vektory váh \mathbf{w}^i expertných modulov a \mathbf{w}^b bránového modulu. Od ich hodnôt závisí veľkosť logaritmickej pravdepodobnostnej funkcie $l_R(\mathbf{w})$.

10.2.1 Úprava váh expertných modulov

Modulárna sieť podľa schémy na Obr. 10.1 pozostáva z K expertných modulov a jedného bránového modulu. Každý expertný modul obsahuje jednu vrstvu pozostávajúcu z M lineárnych neurónov. Ich výstupné hodnoty tvoria vektor výstupných hodnôt \mathbf{y}_i i -teho modulu ($i = 1 \dots K$). Hodnoty výstupného vektora určené podľa vzťahu (10.3) závisia od matice vektorov hodnôt váh \mathbf{w}^i . Prvok \mathbf{w}_j^i tejto matice predstavuje *vektor váh* prislúchajúci j -temu neurónu i -teho expertného modulu. Nové hodnoty tohto vektora sa v n -tom kroku učiaceho procesu vypočítajú analogicky ako vo vzťahoch (10.15) a (10.16).

$$\mathbf{w}_j^i(n+1) = \mathbf{w}_j^i(n) + \Delta \mathbf{w}_j^i(n) \quad (10.17)$$

$$\Delta \mathbf{w}_j^i = \gamma \frac{\partial l_R(\mathbf{w}_j^i)}{\partial \mathbf{w}_j^i} \quad (10.18)$$

Rozdiel oproti metóde najstrmšieho zostupu chybovej funkcie je v znamienku na pravej strane výrazu (10.18). Cieľom pri metóde najstrmšieho vzostupu pravdepodobnostnej funkcie je zväčšovanie jej hodnoty.

Deriváciu funkcie l_R podľa \mathbf{w}_j^i možno podľa pravidla zreťazenia prepísať do tvaru

$$\frac{\partial l_R}{\partial \mathbf{w}_j^i} = \frac{\partial l_R}{\partial y_j^i} \frac{\partial y_j^i}{\partial \mathbf{w}_j^i} \quad (10.19)$$

Výpočet zložitej derivácie sa rozdelil na výpočet dvoch jednoduchších. Derivácia l_R podľa y_j^i sa vypočíta nasledovne

$$\frac{\partial l_R}{\partial y_j^i} = h_i(e_j - y_j^i) \quad (10.20)$$

y_j^i - výstupná hodnota j -teho neurónu i -tej expertnej siete.

e_j - požadovaná hodnota j -teho prvku celkového výstupného vektora \mathbf{e}

h_i - aposteriórna pravdepodobnosť, že i -ty expertný modul generuje požadovaný výstupný vektor \mathbf{e} . Jej hodnota je daná vzťahom

$$h_i = \frac{g_i \exp(-\frac{1}{2}\|\mathbf{e} - \mathbf{y}^i\|^2)}{\sum_{j=1}^K g_j \exp(-\frac{1}{2}\|\mathbf{e} - \mathbf{y}^j\|^2)} \quad (10.21)$$

Zo vzťahu (10.21) vyplýva, že všetky $h_{1\dots K}$ späňajú podmienky

$$h_i \in <0, 1> \quad (10.22)$$

$$\sum_{i=1}^K h_i = 1$$

Kedže aktivačné aj výstupné funkcie neurónov v expertných moduloch sú lineárne, hodnota y_j^i sa podľa vzťahu (10.3) vypočíta nasledovne

$$y_j^i = \mathbf{u}^T \mathbf{w}_j^i \quad (10.23)$$

Deriváciou výrazu (10.23) podľa \mathbf{w}_j^i sa získa druhá časť výrazu (10.19).

$$\frac{\partial y_j^i}{\partial \mathbf{w}_j^i} = \mathbf{u} \quad (10.24)$$

Dosadením (10.20) a (10.24) do (10.19) a dosadením do (10.18) sa získa vzorec pre výpočet hodnoty $\Delta \mathbf{w}_j^i$

$$\Delta \mathbf{w}_j^i = \gamma h_i (e_j - y_j^i) \mathbf{u} \quad (10.25)$$

kde γ predstavuje veľkosť učiaceho parametra. Výsledný vzorec pre výpočet novej hodnoty vektora váh \mathbf{w}_j^i , ktorý prislúcha j -temu neurónu i -teho expertného modulu má tvar

$$\mathbf{w}_j^i(t+1) = \mathbf{w}_j^i(t) + \gamma h_i(t) (e_j(t) - y_j^i(t)) \mathbf{u} \quad (10.26)$$

10.2.2 Úprava váh bránového modulu

Bránový modul obsahuje K nelineárnych neurónov s výstupnou funkciu typu softmax. Výpočet hodnôt g_i je definovaný vzťahom (10.5), ktorého dosadením do (10.13) je určené vyjadrenie pravdepodobnostnej funkcie pomocou vektora váh bránového modulu \mathbf{w}^b

$$l_R(\theta) = \ln \sum_{i=1}^K \exp(x_i) \cdot \exp\left(-\frac{1}{2}\|\mathbf{e} - y_i\|^2\right) - \ln \sum_{j=1}^K \exp(x_j) \quad (10.27)$$

Hodnoty $x_{1\dots K}$ sú výstupnými hodnotami neurónov bránového modulu, ktoré sa vypočítajú podľa vzťahu (10.4).

ďalší postup odvodenia úpravy hodnôt vektora váh \mathbf{w}^b je analogický postupu odvodenia úpravy váh pre expertné moduly. Rozdiel je v počte a type

neurónov bránového modulu. Expertné moduly obsahovali M lineárnych neurónov. Bránový modul obsahuje K nelineárnych neurónov. Nové hodnoty vektora váh \mathbf{w}_i^b , ktorý prislúcha i -temu neurónu sa určia nasledovne

$$\mathbf{w}_i^b(n+1) = \mathbf{w}_i^b(n) + \Delta \mathbf{w}_i^b(n) \quad (10.28)$$

$$\Delta \mathbf{w}_i^b = \gamma \frac{\partial l_R(\mathbf{w}_i^b)}{\partial \mathbf{w}_i^b} \quad (10.29)$$

Učiaci parameter γ bránovej siete môže byť zhodný s učiacim parametrom pre expertné siete. Parciálnu deriváciu pravdepodobnostnej funkcie je možné znova rozpísat pomocou pravidla zreťazenia.

$$\frac{\partial l_R}{\partial \mathbf{w}_i^b} = \frac{\partial l_R}{\partial x_i} \frac{\partial x_i}{\partial \mathbf{w}_i^b} \quad (10.30)$$

Derivácia pravdepodobnostnej funkcie podľa x_i vyjadrená pre i -ty neurón bránovej siete má tvar

$$\frac{\partial l_R}{\partial x_i} = h_i - g_i \quad (10.31)$$

Derivácia x_i podľa vektora váh prislúchajúceho i -temu neurónu bránovej siete má tvar

$$\frac{\partial x_i}{\partial \mathbf{w}_i^b} = \mathbf{u} \quad (10.32)$$

Dosadením (10.31) a (10.32) do (10.30) a následným dosadením do (10.28) sa získá vzorec pre výpočet hodnoty $\Delta \mathbf{w}_i^b$

$$\Delta \mathbf{w}_i^b = \gamma(h_i - g_i)\mathbf{u} \quad (10.33)$$

Výsledný vzorec pre výpočet novej hodnoty vektora váh prislúchajúceho i -tému neurónu bránového modulu má tvar

$$\mathbf{w}_i^b(n+1) = \mathbf{w}_i^b(n) + \gamma(h_i(n) - g_i(n))\mathbf{u} \quad (10.34)$$

Zhrnutie učiaceho algoritmu pre regresiu:

1. *Incializácia.*

Hodnoty všetkých váh celej siete sa nastavia na náhodnú hodnotu z malého intervalu, napr. $\langle -1, 1 \rangle$.

2. *Úprava váh.*

Úprava váh sa uskutočňuje v T cykloch. V každom cykle sa priviedú na vstup a výstup siete všetky vzorky trénovacej množiny. Každá vzorka je reprezentovaná dvojicou $\{\mathbf{u}, \mathbf{e}\}$. Pre jednotlivé indexy platí:

$$t = 0, 1, 2, \dots, T$$

$$i = 1, 2, \dots, K$$

$$j = 1, 2, \dots, M$$

(a)

$$x_i(n) = \mathbf{u}^T \mathbf{w}_i^b(n)$$

$$g_i(n) = \frac{\exp(x_i(n))}{\sum_{j=1}^K \exp(x_j(n))}$$

(b)

$$y_j^i(n) = \mathbf{u}^T \mathbf{w}_j^i(n)$$

(c)

$$h_i(n) = \frac{g_i(n) \exp(-\frac{1}{2} \|\mathbf{e} - \mathbf{y}_i(n)\|^2)}{\sum_{j=1}^K g_j(n) \exp(-\frac{1}{2} \|\mathbf{e} - \mathbf{y}_j(n)\|^2)}$$

(d)

$$\mathbf{w}_j^i(n+1) = \mathbf{w}_j^i(n) + \gamma h_i(n)(d_j(n) - y_j^i(n))\mathbf{u}$$

(e)

$$\mathbf{w}_i^b(n+1) = \mathbf{w}_i^b(n) + \gamma(h_i(n) - g_i(n))\mathbf{u}$$

10.3 Učiaci algoritmus pre klasifikáciu

Výstupné hodnoty kvalifikátora nadobúdajú diskrétné hodnoty. Ich popis je možný pomocou Bernoulliho rozdelenia pravdepodobnosti. Pravdepodobnostnú funkciu $l_K(\theta)$ potom možno vyjadriť podľa vzťahu (10.14). Neuróny v expertných a bránovom module nie sú lineárne. Výstupná hodnota j -teho neurónu i -teho experného modulu je určená sigmoidálnou funkciou

$$y_j^i = \frac{1}{1 + \exp(-\mathbf{u}^T \mathbf{w}_j^i)} \quad (10.35)$$

Podobne je určená aj aktivačná hodnota i -teho neurónu bránového modulu

$$x_i = \frac{1}{1 + \exp(-\mathbf{u}^T \mathbf{w}_i^b)} \quad (10.36)$$

Výstupné hodnoty g_i neurónov bránového modulu sú určené nelinearitou typu softmax, ale bez funkcie \exp

$$g_i = \frac{x_i}{\sum_{j=1}^K x_j} \quad (10.37)$$

Hodnoty y_j^i a g_i späňajú nasledujúce podmienky:

$$y_j^i, g_i \in \langle 0, 1 \rangle \quad (10.38)$$

$$\prod_{j=1}^M y_j^i \in \langle 0, 1 \rangle$$

$$\sum_{i=1}^K g_i = 1$$

Podľa týchto ohraničení sa interpretuje činnosť bránového modulu ako klasifikácia nad celým definičným oborom aproximovanej funkcie. Takto sa vstupný priestor vzoriek rozdelí na viacej oblastí. činnosť expertných modulov sa interpretuje ako klasifikácia vo vnútri jednotlivých oblastí vstupného priestoru vzoriek. Na tomto mieste je potrebné podotknúť, že výstupné vektoru učiacich vzoriek sú kódované podľa pravidla 1-z-P. Ak aktuálna učiaca vzorka patrí do triedy p , výstupný vektor potom obsahuje jednu jednotku na pozícii p a zvyšné prvky vektora sú nulové. Hodnota g_i sa interpretuje ako pravdepodobnosť, že i -ty expertný modul generoval aktuálmu učiacu vzorku. Hodnota y_j^i sa interpretuje ako pravdepodobnosť, že i -ty modul klasifikuje aktuálmu vzorku do triedy j .

Ďalší postup odvodenia učiaceho algoritmu je analogický postupu pre odvodenie učiaceho algoritmu pre regresiu. Rozdiel je pri výpočte aposteriórnych pravdepodobností h_i pre potreby klasifikácie.²

$$h_i = \frac{g_i \prod_{k=1}^M y_{ik}^{e_{ik}} (1 - y_{ik})^{1-e_{ik}}}{\sum_{j=1}^K g_j \prod_{k=1}^M y_{jk}^{e_{jk}} (1 - y_{jk})^{1-e_{jk}}} \quad (10.39)$$

Derivácie y_j^i a x_i podľa vektorov váh, ktoré k nim prislúchajú, sú tvaru

$$\frac{\partial y_j^i}{\partial \mathbf{w}_j^i} = \frac{\exp(-\mathbf{u}^T \mathbf{w}_j^i)}{(1 + \exp(-\mathbf{u}^T \mathbf{w}_j^i))^2} \mathbf{u} \quad (10.40)$$

$$\frac{\partial x_i}{\partial \mathbf{w}_i^b} = \frac{\exp(-\mathbf{u}^T \mathbf{w}_i^b)}{(1 + \exp(-\mathbf{u}^T \mathbf{w}_i^b))^2} \mathbf{u} \quad (10.41)$$

²Pri programovej realizácii tohto algoritmu sa ukázalo, že pre výpočet aposteriórnej pravdepodobnosti expertného modulu je vhodnejšie použiť priemernú hodnotu ako o súčin hodnôt výstupných neurónov umocnených na požadovanú hodnotu.

Vzťahy pre úpravu hodnôt váh expertných a bránového modulu sú podobné vzťahom (10.26) a (10.34).

Zhrnutie učiaceho algoritmu pre klasifikáciu:

1. *Incializácia.*

Hodnoty všetkých váh celej siete sa nastavia na náhodnú hodnotu z malého intervalu napr. $\langle -1, 1 \rangle$.

2. *Úprava váh.*

Úprava váh sa uskutočňuje v T cykloch. V každom cykle sa privedú na vstup a výstup siete všetky vzorky trénovacej množiny. Každá vzorka je reprezentovaná dvojicou $\{\mathbf{u}, \mathbf{d}\}$.

(a)

$$x_i(n) = \frac{1}{1 + \exp(-\mathbf{u}^T \mathbf{w}_i^b(n))}$$

$$g_i(n) = \frac{x_i(n)}{\sum_{j=1}^K x_j(n)}$$

(b)

$$y_j^i(n) = \frac{1}{1 + \exp(-\mathbf{u}^T \mathbf{w}_j^i(n))}$$

(c)

$$h_i(n) = \frac{g_i(n) \prod_{k=1}^M y_{ik}^{e_{ik}} (1 - y_{ik})^{1-e_{ik}}(n)}{\sum_{j=1}^K g_j(n) \prod_{k=1}^M y_{jk}^{e_{jk}} (1 - y_{jk})^{1-e_{jk}}(n)}$$

(d)

$$\mathbf{w}_j^i(n+1) = \mathbf{w}_j^i(n) + \gamma h_i(n)(e_j(n) - y_j^i(n)) \frac{\exp(-\mathbf{u}^T \mathbf{w}_j^i)}{(1 + \exp(-\mathbf{u}^T \mathbf{w}_j^i))^2} \mathbf{u}$$

(e)

$$\mathbf{w}_i^b(n+1) = \mathbf{w}_i^b(n) + \gamma(h_i(n) - g_i(n)) \frac{\exp(-\mathbf{u}^T \mathbf{w}_i^b)}{(1 + \exp(-\mathbf{u}^T \mathbf{w}_i^b))^2} \mathbf{u}$$

10.4 EM algoritmus

Použitie modulárnej architektúry predpokladá zvládnutie problému rozdeľenia jednotlivých učiacich vzoriek medzi expertné moduly a sformovanie celkového výstupu. Algoritmus *Expectation Maximisation* - EM je alternatívnym riešením tohto problému k už uvedenému spôsobu metódou najstrmšieho vzostupu pravdepodobnostnej funkcie.

EM algoritmus vznikol v roku 1977 a bol pôvodne využívaný pre nekontrolované učenie v kontexte zhľukovania. Veľmi rozšírené je jeho použitie v oblasti matematiky. Použitie pre kontrolované učenie spolu s modulárnou architektúrou uvádzajú Jacobs a Jordan v [?]. Ide o úpravu voľných parametrov siete za účelom maximalizácie pravdepodobnostnej funkcie pomocou algoritmu EM.

Nech \mathcal{Z} je množina binárnych vektorov $\mathbf{z}_{1\dots P}$, pričom P je počet vzoriek trénovacej množiny \mathcal{X} . Každý vektor množiny \mathcal{Z} je K -prvkový, pričom K je počet expertných modulov. Prvok $z_{ij} = 1$ ak je i -ta vzorka generovaná j -tym expertným modulom. Pridaním množiny \mathcal{Z} k trénovacej množine \mathcal{X} vznikne nová množina \mathcal{Y} . Takýmto rozšírením trénovacej množiny vznikne *kompletnejšia* pravdepodobnostná funkcia $l_C(\theta; \mathcal{Y})$. Ak by bola množina Z náozaj známa, optimalizácia kompletnej pravdepodobnostnej funkcie by bola jednoduchšia. Keďže množina \mathcal{Z} je neznáma, EM algoritmus je rozdelený na dve časti. Najprv sa nájde očakávaná hodnota kompletnej pravdepodobnostnej funkcie. V druhom kroku sa hľadajú také hodnoty parametrov θ , ktoré zväčšujú jej hodnotu. *E-krok*:

$$Q(\theta, \theta_n) = E[l_C(\theta; \mathcal{Y}) | \mathcal{X}] \quad (10.42)$$

M-krok:

$$\theta_{n+1} = \arg \max_{\theta} Q(\theta, \theta_n) \quad (10.43)$$

Jacobs a Jordan v [?] uvádzajú, že zväčšením hodnoty kompletnej pravdepodobnostnej funkcie sa dosiahne aj zväčšenie hodnoty pôvodnej pravdepodobnostnej funkcie oproti predošlému kroku výpočtu. Každá iterácia tak zväčší hodnotu $l(\theta; \mathcal{X})$

$$l(\theta_{n+1} | \mathcal{X}) \geq l(\theta_n | \mathcal{X}) \quad (10.44)$$

Hodnota pravdepodobnostnej funkcie $l(\theta; \mathcal{X})$ sa monotónne zväčšuje spolu s postupnosťou odhadov parametrov θ , ktoré sú generované EM algoritmom. To je podľa [?] príčinou konvergencie do *lokálneho maxima*. Napriek výhodám oproti metóde najstrmšieho vzostupu nie je EM algoritmus odolný voči takejto nežiadúcej konvergencii.

Konkrétna implementácia pre úlohy klasifikácie je prevzatá z [?]. Architektúra siete je zhodná s architektúrou podľa Obr. 10.1. Všetky siete majú v tomto prípade výstupnú funkciu typu softmax. Výstupná hodnota j -teho neurónu i -tej expertnej siete je určená vzťahom

$$y_j^i = \frac{\exp(\mathbf{u}^T \mathbf{w}_j^i)}{\sum_{k=1}^M \exp(\mathbf{u}^T \mathbf{w}_k^i)} \quad (10.45)$$

Podobne sú určené výstupné hodnoty x_i neurónov bránovej siete.

$$a_i = \frac{\exp(\mathbf{u}^T \mathbf{w}_i^b)}{\sum_{j=1}^K \exp(\mathbf{u}^T \mathbf{w}_j^b)} \quad (10.46)$$

Pre všetky expertné moduly potom platí

$$\prod_{k=1}^M y_{ik} = 1 \quad (10.47)$$

Hodnota y_{it} sa môže interpretovať ako pravdepodobnosť, že i -ty modul klasifikuje aktuálnu učiacu vzorku do triedy p . Waterhouse v [?] uvádza, že ak aktuálna učiaca vzorka patrí do triedy p , potom platí

$$h_i = y_{it} \quad (10.48)$$

pričom h_i je aposteriórna pravdepodobnosť, že i -ty expertný modul klasifikuje aktuálnu učiacu vzorku do triedy p . Zanedbávajú sa tým hodnoty zvyšných neurónov expertného modulu. Je tak možné urobiť, ak sú výstupné vektory učiacich vzoriek kódované podľa pravidla 1-z- P . Pre úpravu hodnôt vektora váh, ktorý prislúcha j -tému neurónu i -teho expertného modulu platí

$$\mathbf{w}_j^i(n+1) = \mathbf{w}_j^i(n) + \gamma \frac{1}{\sum_{p=1}^P h_i(p)} \sum_{p=1}^P h_i(d_j - y_j^i) \mathbf{u}(p) \quad (10.49)$$

Pre úpravu hodnôt vektora váh, ktorý prislúcha i -tému neurónu bránového modulu platí

$$\mathbf{w}_i^b(n+1) = \mathbf{w}_i^b(n) + \gamma \frac{1}{\sum_{p=1}^P h_i(p)} \sum_{p=1}^P h_i(h_i - x_i) \mathbf{u}(p) \quad (10.50)$$

Z uvedených vzťahov vyplýva, že úprava hodnôt váh sa vykonáva iba raz na konci každého učiaceho cyklu. Pri metóde najstrmšieho vzostupu sa

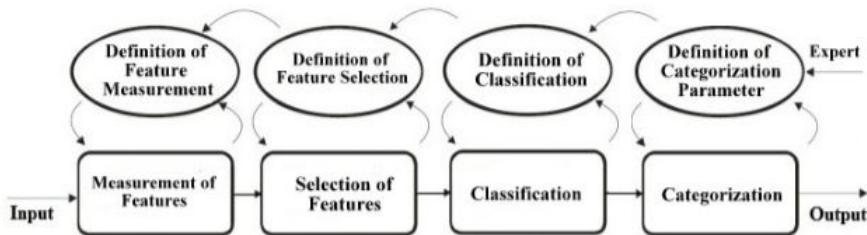
hodnoty váh upravujú pri prezentácii každej vzorky. Učenie pomocou EM algoritmu je preto z hľadiska behu v reálnom čase v porovnaní s touto metódou rýchlejšie. Podľa predpokladov by učenie pomocou EM algoritmu malo byť rýchlejšie aj z hľadiska počtu učiacich cyklov potrebných na dosiahnutie minimálnej chyby učenia.

Kapitola 11

Hlboké učenie neurónových sietí

Hlboké neuronové siete sú v súlade s základnou štruktúrou rozpoznávacieho procesu prostriedkom na

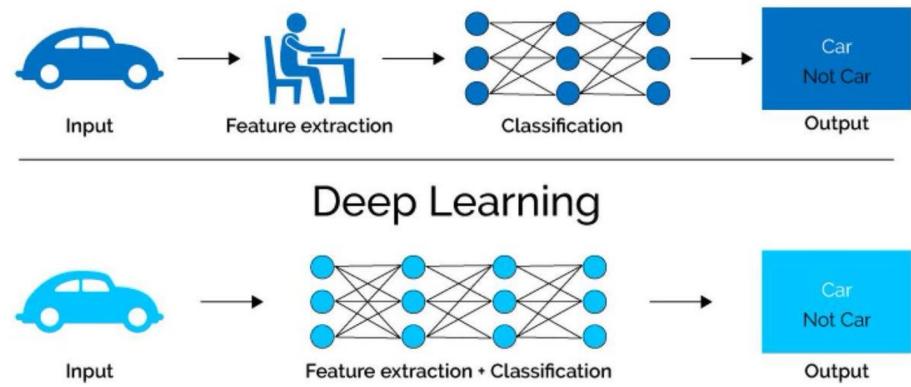
- meranie príznakov
- výber príznakov
- klasifikácia
- kategorizácia



Obr. 11.1: Základná schéma rozpoznávacieho procesu

Tieto 4 časti predstavujú základnú štruktúru rozpoznávania. Podčiarkujeme rozdiel medzi rozpoznávacím procesom a klasifikačným procesom resp.

kategorizáciou. Na nasledujúcom obrázku demonštrujeme porovnanie plytkých a hlbokých neurónových sietí.

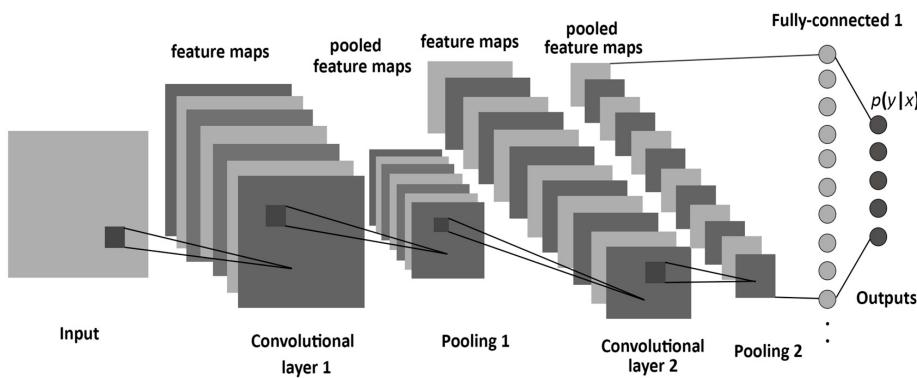


Obr. 11.2: Porovnanie plytkých a hlbokých neurónových sietí

Kapitola 12

Konvolučné neurónové siete

Z časti o plynkých neurónových sieťach vieme, že tieto siete v princípe splňujú tzv. **Univerzálnu aproximačnú teorému** pre approximáciu ľubovoľnej všade hladkej funkcie. V tejto časti sa budeme venovať problematike využitia hlbokých neurónových sietí v spracovaní obrazovej informácie. Typická topológiou takejto konvolučnej neurónovej sieti je nasledovnom obrázku



Obr. 12.1: Príklad typickej topológie konvolučnej neurónovej siete

12.1 Základné pojmy

V tejto časti si priblížime základné pojmy a taktiež základný matematický aparát pre pochopenie práce s konvolučnými neurónovými sieťami. Taktiež je treba pripomenúť že sa sústredíme na aplikáciu CNN na spracovanie digi-

tálneho obrazu.

12.1.1 Pojmy pri CNN

V práci s CNN narazíme na pojem hyperparametre a parametre CNN. Základný rozdiel medzi nimi je že

- **hyperparametre** sú definované užívateľom (ich voľba alebo nastavenie potrebujú určitý stupeň intuície a potrebujú skúsenosti s prácou na hlbokých neurónových sieťach)
- **parametre** - tieto sa adaptujú teda učia, v prípade kontrolovaného učenia podľa chybovej funkcie v rámci chybového priestoru tvoreného napr. W počtom parametrov a plus jeden rozmer chyba. Teda máme

$$(W + 1)$$

rozmerný chybový priestor, kde hľadáme globálne minimum pre $(W + 1)$. parameter práve hľadaním W optimálnych parametrov v celom chybovom priestore (vo väčšine prípadov gradientovou metódou, parciálnej derivácie chyby podľa jednotlivých parametrov).

Medzi hyperparametre patria nasledovne údaje zvolené užívateľom:

1. veľkosť filtra - veľká väčšina používa štvorcové filtre 3x3xch, 5x5xch, 7x7xch , kde ch je počet kanálov
2. počet filtrov p aplikovaných v jednej konvolučnej vrstve
3. počet konvolučných vrstiev
4. Výber aktivačnej funkcie konvolučnej vrstve
5. výber typu poolingovej vrstvy
6. výber typu vektorizácie a následnej integračnej vrstvy
7. definovanie aktivačnej funkcie na výstupe (pre klasifikáciu väčšinou softmax funkcia).

Klúčovým pojmom v CNN je pojem "konvolucia". Pojem "convolution" pochádza z obdobia 1690–1700 a latinského slova "convolūtus" alebo "convolū" podľa ?? znamená ako sloveso "formovať v rotovanom tvare" alebo ako prídavné meno vo význame

"skrútené kde jedna časť je nad druhou časťou".

Pôvod tohto latinského slova je v slove "volvere" čo znamená "otočit".

V matematike sa tento pojem začal používať okolo roku 1750, kde sa používal pri DAlembertovej derivácii Taylorovho radu ?.?. V princípe matematický pojem **konvolúcia** definuje funkciu ktorá predstavuje vzťah dvoch funkcií na vstupe a na výstupe je tretia funkcia, ktorá je "konvolúciou" vstupných funkcií napr. f a g . Operátor konvolúcie dvoch funkcií sa definuje ako $f * g$.

Tieto funkcie môžu byť ľubovoľné - vo všeobecnosti komplexné funkcie komplexnej premennej. V jednoduchom prípade reálne funkcie reálnej premennej.

Majme dve jednorozmerné funkcie závisle na čase $f(t)$ a $g(t)$. Potom pod konvolúciou budeme rozumieť výpočet funkcie $Conv_{f,g}(t)$

$$Conv_{f,g}(t) = f(t) * g(t) = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \quad (12.1)$$

alebo v pre diskrétnie prípady:

$$Conv_{f,g}(t) = f(t) * g(t) = \sum_{\tau=-\infty}^{\tau=\infty} f(t-\tau)g(\tau) = \sum_{\tau=-\infty}^{\tau=\infty} f(\tau)g(t-\tau) \quad (12.2)$$

Tento pojem "konvolúcia" je podobný pojmu "kross-korelácia", ktorý je definovaný pre jednorozmerný signál nasledovne (všimnite si podobnosť medzi vzorcami 12.1 a 12.3 a 12.2 s rovnicou 12.4).

Teda pod kross-Koreláciou budeme rozumieť funkciu $Cross_{f,g}(t)$ dvoch funkcií f a g . Operátor kross-korelácie dvoch funkcií sa definuje ako $f \star g$ a všimnite si rozdiel označenia medzi operátormi konvolúcie a kross-korelácie " $*$ " a " \star ".

$$Cross_{f,g}(t) = f(t) \star g(t) = \int_{-\infty}^{\infty} f^*(t+\tau)g(\tau)d\tau = \int_{-\infty}^{\infty} f^*(\tau)g(t+\tau)d\tau \quad (12.3)$$

alebo v pre diskrétny prípade:

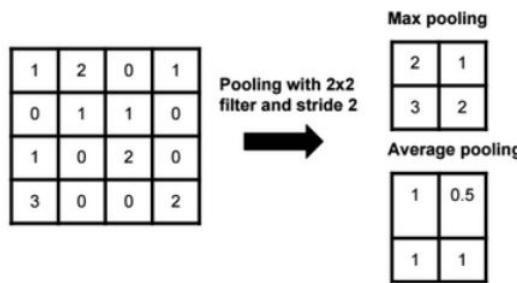
$$Cross_{f,g}(t) = f(t) \star g(t) = \sum_{\tau=-\infty}^{\tau=\infty} f^*(t+\tau)g(\tau) = \sum_{\tau=-\infty}^{\tau=\infty} f^*(\tau)g(t+\tau) \quad (12.4)$$

kde f^* je komplexne združená funkcia ku funkciu f . Ak je však táto funkcia reálna funkcia reálnej premennej tak $f^* = f$. V princípe Kross-korelácia dvoch funkcií popisuje "podobnosť - match" dvoch funkcií

$$f(t) \text{ a } g(t).$$

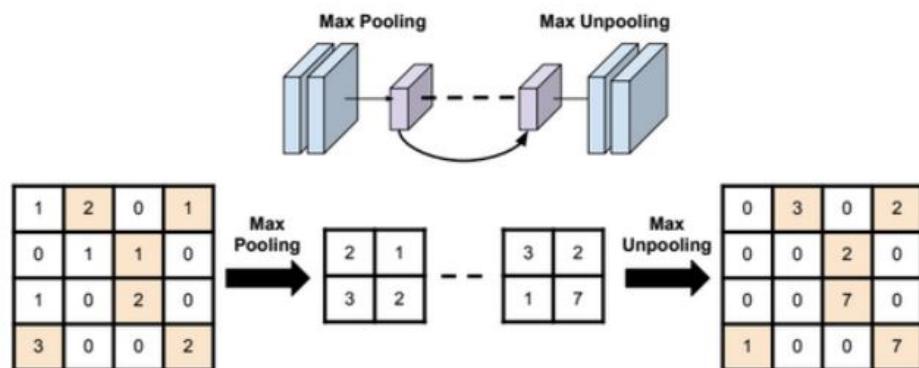
Medzi základné pojmy pri práci CNN narazíme na nasledovné pojmy

1. pojem vrstva konvolučná , operácia konvolucie a crosskoralacie
2. pojem vrstva pooling - jednoduchá matematická operácia z predchádzajúcej vrstvy (môže byť MAX pooling, AVERAGE pooling a iné

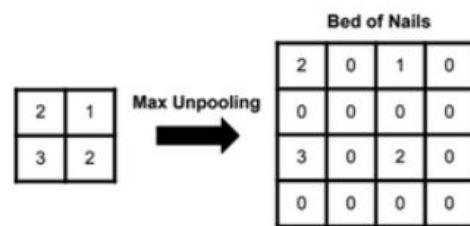


Obr. 12.2: Ilustračný obrázok pre operáciu pooling na digitálnom obrazze

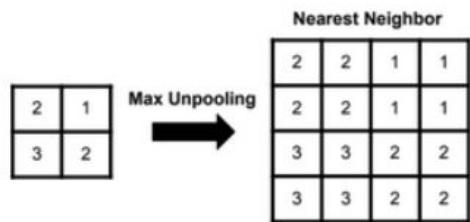
a súčasne formy inverzného poolingu tzv. unpoolingu sú napr. nasledovne



Bed of nails unpooling



Nearest neighbor unpooling



Obr. 12.3: Ilustračný obrázok operácie unpooling na digitálnom obraze

3. pojem Vectorizácia a Concatizácia - priestorové pre usporiadanie neurónov do inej geometrickej formácie

4. pojem "padding" - sposob aplikácie filtra na predošlú vrstvu. Ak padding je = 0 tak ak predošlá vrstva mala rozmer ($n * m * ch$) tak po aplikácii filtra o veľkosti ($f1 * f2 * ch$) bude výsledok konvolučnej vrstvy ($n + 2p - f1 + 1, m + 2p - f1 + 1, ch$) teda ak mame ($n = m = 28, 1$) a filter je ($u = v = 5, 1$) tak dostaneme na výstupe

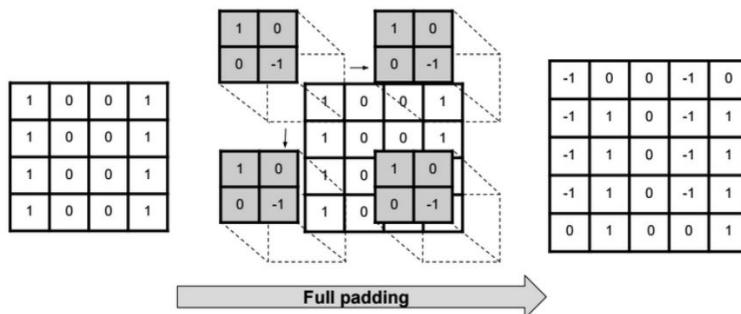
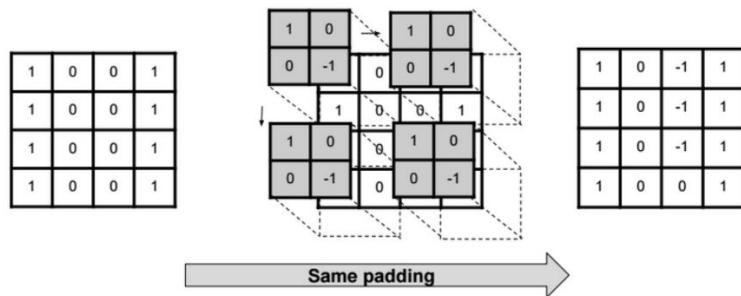
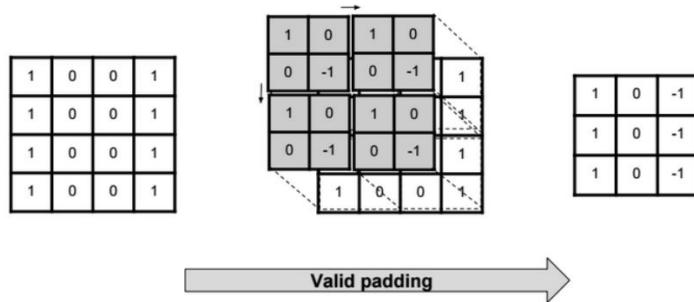
$$(28 + 2 * 0 - 5 + 1, 28 + 2 * 0 - 5 + 1, 1)$$

teda $24x24$. Ak je "padding = 0" (tzv. valid padding) tak postupne sa obrázok zmenšuje a preto sa niekedy využíva aj tzv. "same padding" teda obrázok sa nezmenšuje lebo pôvodný sa rozšíri a padding bude $padding = (f1 - 1)/2$ teda ak je filter $3x3$ tak pri padding=1 sa obrázok neredukuje, pri $5x5$ tak pri padding=2 sa obrázok taktiež neredukuje a podobne ďalej. Pri paddingu sa rozšírenie pôvodného obrázku realizuje číslom "0".

0	0	0	0	0	0
0	0	1	1	3	0
0	1	2	3	5	0
0	2	3	5	1	0
0	0	1	1	1	0
0	0	0	0	0	0

Obr. 12.4: Ilustračný obrázok operácie padding na digitálnom obrazze

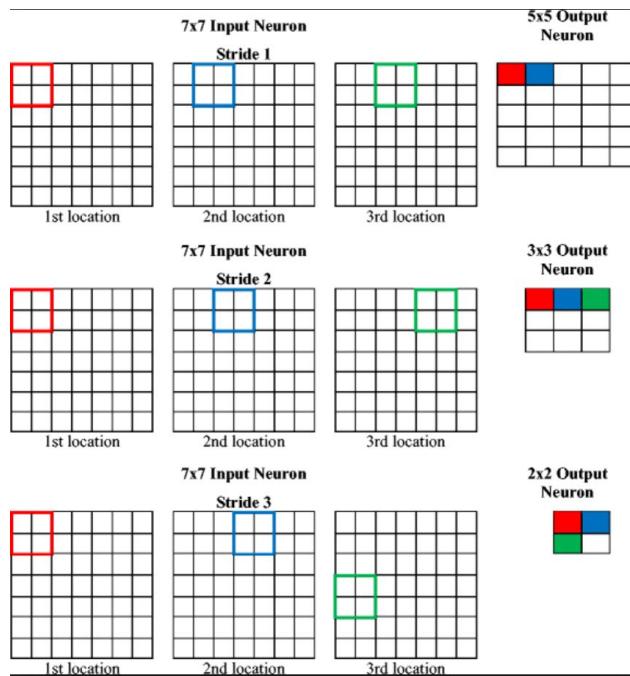
resp. jej rôzne formy

Full padding**Same padding (half padding)****Valid padding (no padding)**

Obr. 12.5: Ilustračný obrázok rôznych typov operácie padding na digitálnom obrazu

5. pojem *"strading"* hovorí o spôsobe posunu filtra na obrázku. Ak je $strading = 1$ filter sa kĺže po obrazu a stále sa vypočítava referenčne

nová hodnota centrálneho pixel, ak je $strading > 1$ filter sa aplikuje skokom po obraze nie kľzaním. Príklady na rôzne typy "strading" sú na nasledovnom obrázku

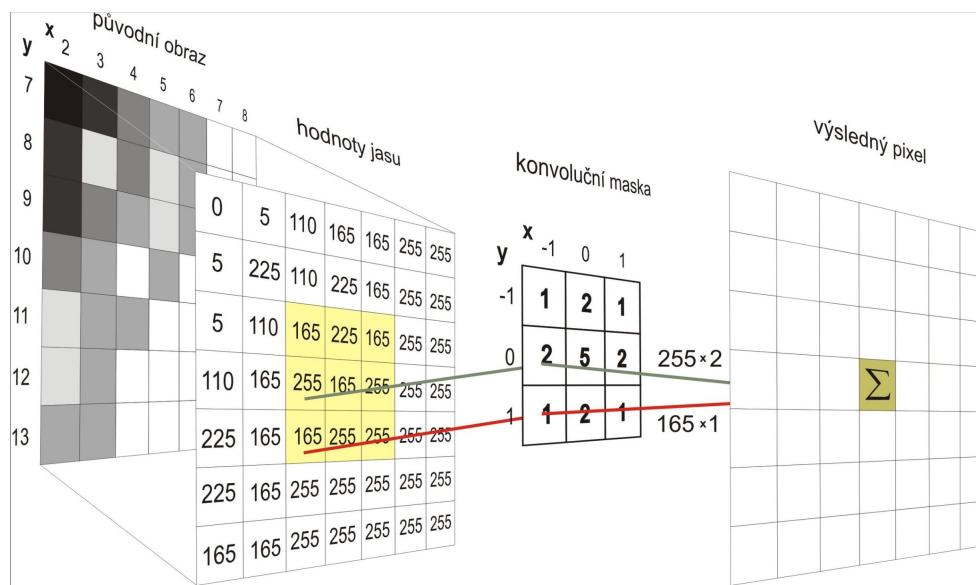


Obr. 12.6: Ilustračný obrázok rôznych typov stridingu na digitálnom obrazze

- pojem konvolúcia dvoch funkcií v matematike je definovaná pre diskrétny prípad ako skalár a to nasledovne ak f je funkcia obrazu a g je funkcia kernelu (masky aplikovanej na obraz).

$$(f * g)(x, y) = \sum_{i=-k}^{i=k} \sum_{j=-k}^{j=k} f(x - i, y - j) * g(i, j) \quad (12.5)$$

V princípe ide o lokálny operátor (kernel, filter) \mathbf{g} aplikovaný napr. na digitálny obraz \mathbf{f} . Výsledok tejto operácie bude hodnota pixela vo príznakovom poli na súradnici \mathbf{x}, \mathbf{y} . Dobre na tejto operácii je že sa dá vykonať rýchlo a pre celý obraz s vysokým stupňom paralelizácie. Ilustračne to zobrazuje nasledovný obrázok.



Obr. 12.7: Ilustračný obrázok konvolučnej operácie na digitálnom obraze

Súčasne je treba povedať, že parametre filtra môžeme považovať za považovať za synaptické váhy, ktoré budeme v ďalšom adaptovať.

12.2 Jednoduchá CNN na spracovanie obrazu

V nasledujúcom prípade si popíšme topológiu jednoduchej konvolučnej neurónovej siete - nazvime ju pracovne **CNNSIMPLE**.

Na vstupe siete budeme mať binárny obrázok 28x28 kde budeme môcť predkladať číslice z intervalu prirodzených čísel $< 0, 9 >$ a na výstupe sa nám vybudí jeden z 10 výstupov indikujúci číslicu na vstupe. Takže popis samotnej topológie a logiky siete je nasledovný :

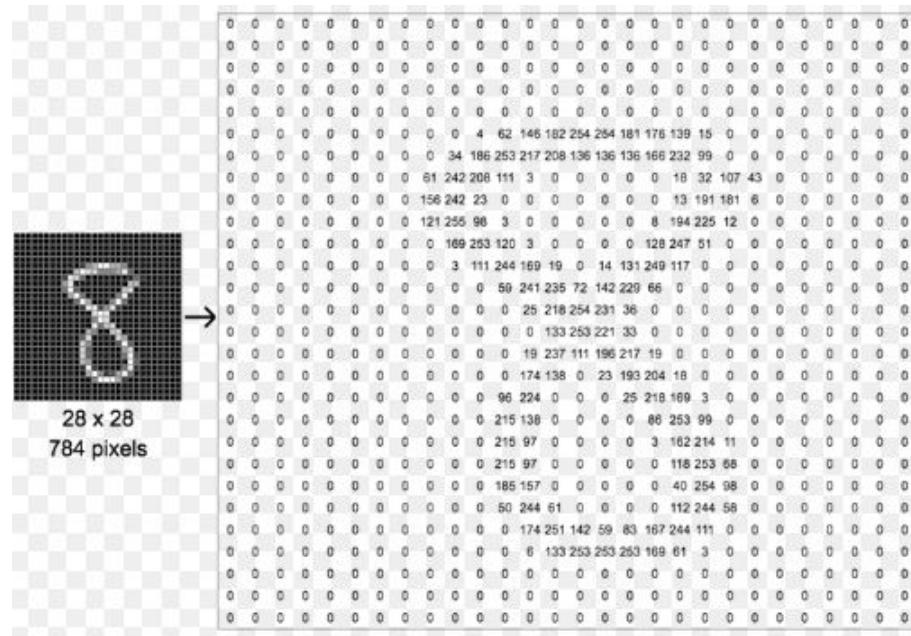
1. vstupom do siete je digitálny obraz o rozmeroch $n=28$ (šírka), $m=28$ (výška), $ch=1$ (počet kanálov) je to monochromatický obraz, kde hodnoty pixlov sú z intervalu prirodzených čísel $< 0, 255 >$. Na vstupnom obraze sú rôzne tvary čísel z intervalu prirodzených čísel $< 0, 9 >$. Príklady takýchto obrázkov sú na nasledujúcom obrázku 12.8. Každá jedna číslica na vstupnom obrázku má rozmer 28x28 pixlov.



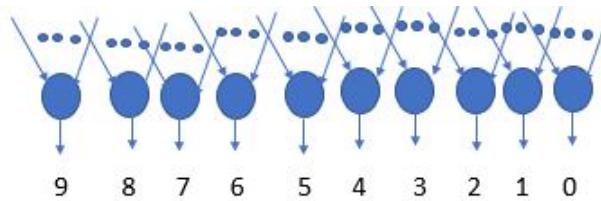
Obr. 12.8: Príklady vstupov do CNNSIMPLE neurónovej siete

teda každé jedno číslo na obrázku je reprezentované matici čísel 28x28x1 ako napríklad číslo 8 na obrázku 12.9

2. Výstupom zo CNNSIMPLE je vektor 10 hodnôt, ktorý prezentuje o ktorú číslicu sa jedna. Ten výstupný neurón bude mať najvyššiu hodnotu ako je to na nasledovnom obrázku 12.10 načrtnuté.



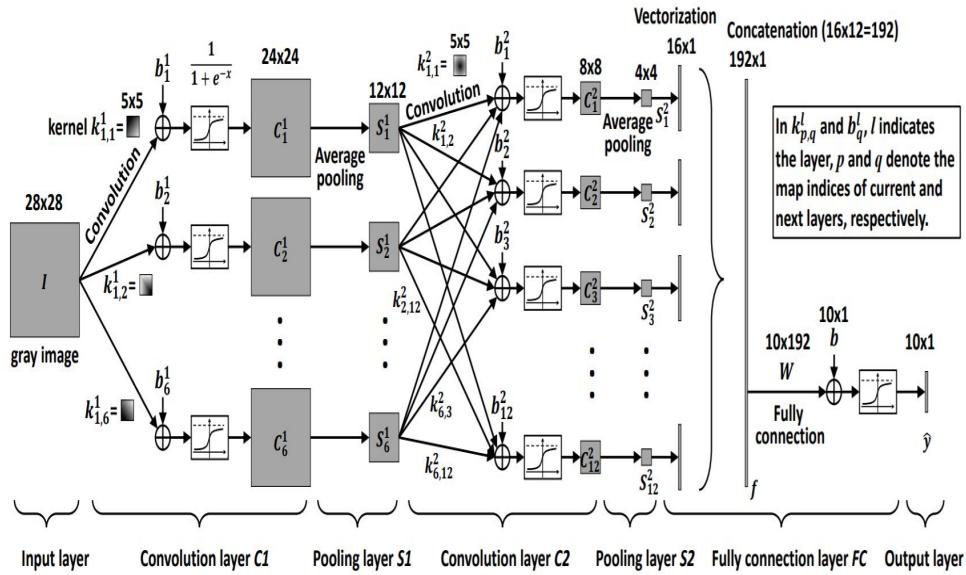
Obr. 12.9: Príklady vstupu čísla 8 ako matice čísel 28x28x1 do CNNSIMPLE neurónovej siete



Obr. 12.10: výstup z CNNSIMPLE (výstup z klasifikačnej časti CNN)

12.2.1 Topológia, parametre a dopredný prechod cez CNN-SIMPLE

V ďalšom si popíšeme topológiu CNNSIMPLE, jej topológiu a identifikujeme jej parametre resp. hyperparametre. Samotná neurónová sieť je na obrázku [12.11](#).



Obr. 12.11: Príklad jednoduchej konvolučnej neurónovej siete

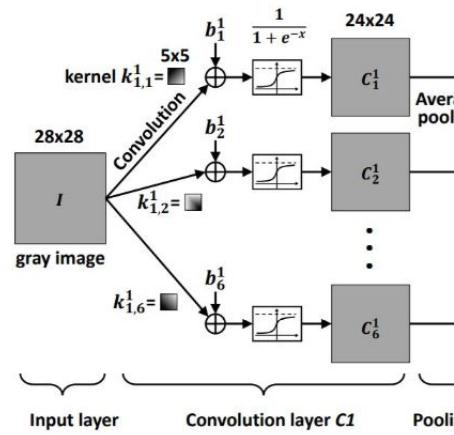
Ako je jasné z obrázku 12.11 CNNSIMPLE sa skladá z nasledovných 4 častí :

1. Vstup pozostáva z digitalného obrazu o rozmeroch 28×28 pixelov teda matice o rovnakých rozmeroch.
2. Konvolučná vrstva C1, ktorá sa skladá
 - (a) 6 filtrov $5 \times 5 \times 1$, ktoré a následne výsledok konvolúcie ide o aktivačnej funkcie sigmoidálnej. Po každej konvolúcii a výpočte sigmoidálnej funkcie sa na vytvorí hodnota vo všetkých 6 vrstvách a pripočítava baies. Teda výpočet hodnôt v pre C_1^1 teda prvá hodnota C^1 pre filter 1 bude nasledovná

$$C_1^1 = \sigma(I * k_{1,1}^1 + b_1^1) \quad (12.6)$$

kde σ je sigmoidálna funkcia

$$\sigma = \frac{1}{1 + \exp^{-x}} \quad (12.7)$$



Obr. 12.12: Vstupná časť CNNSIMPLE

a súčasne platí že

$$x = (I * k_{1,1}^1 + b_1^1)$$

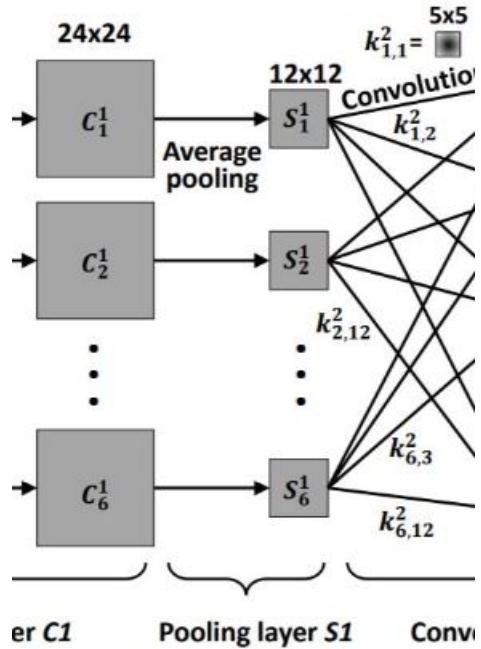
a teda celkový výsledok po konvolúcii je pre všetky filtre môžeme zapísť nasledovne

$$C_p^1(i, j) = \sigma \left(\sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} I(i-u, j-v) \cdot k_{1,p}^1(u, v) + b_p^1 \right) \quad (12.8)$$

kde

- I je vstupný obraz $28x28$
- $k_{i,p}^j$ je hodnota jadra (kernelu) filtra "p" na pozícii i, j
- b_p^1 je prah (bias) filtra "p" v našom prípade je stále iba jedna hodnota na filter
- padding je nulový
- ...

3. Vrstva S1 - Pooling vrstva



Obr. 12.13: Druhá časť CNNSIMPLE

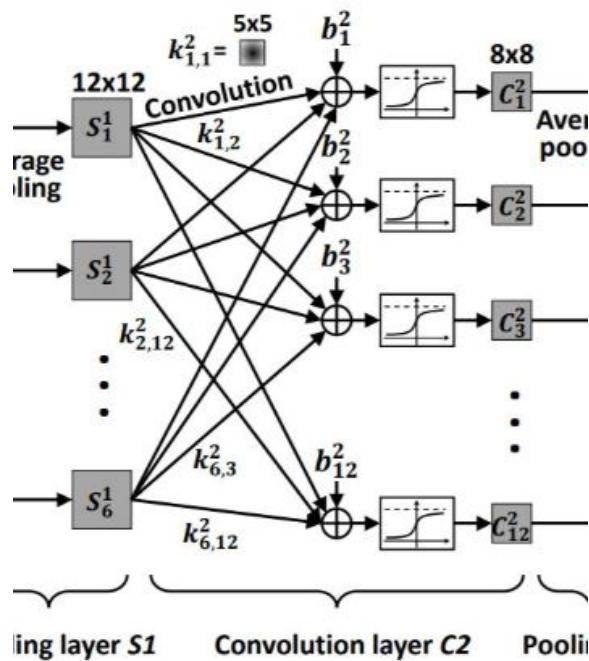
Ak z predchádzajúcej časti vieme vypočítať hodnoty $C_p^1(i, j)$ zo vzorca 12.8 sa tzv. Average Pooling vypočíta jednoducho nasledovne

$$S_p^1 = \frac{1}{4} \left(\sum_{u=0}^{v=1} \sum_{v=0}^{v=1} C_p^1(2i - u, 2j - v) \right) \quad (12.9)$$

kde $i, j = 1, 2, \dots, 12$ a $p = 1, \dots, 6$ lebo C_p^1 vrstva má rozmer 24×24 a priemerovaným poolingom dostaneme rozmer 12×12 .

4. Konvolučná vrstva $C2$

následne po vrstve $S1$ pozostáva z 12 filtrov $5x5$, s nulovým paddingom a jednotkovým stridom, aplikovaných na 6 obrazov po poolingu o rozmeroch $12x12$. Výsledok konvolúcie znova prechádza do sigmoidálnej aktivačnej funkcie σ_q a prahom b_q kde $q = 1, \dots, 12$. Za takýchto podmienok dostaneme na výstupe vrstvy $C2$ 12 obrazov C_q^2 o rozmeroch $8x8$ ako výsledok konvolúcie filtra $3x3$ na obraze $12x12$ s nulovým paddingom a stridom o hodnote 1.



Obr. 12.14: Tretia časť CNNSIMPLE

Teda znova ak

$$\sigma = \frac{1}{1 + \exp^{-x}} \quad (12.10)$$

a súčasne platí že

$$x = (S_p^1 * k_{p,q}^2 + b_q^2)$$

a teda celkový výsledok po konvolúcii je pre všetky filtre môžeme zápisť nasledovne

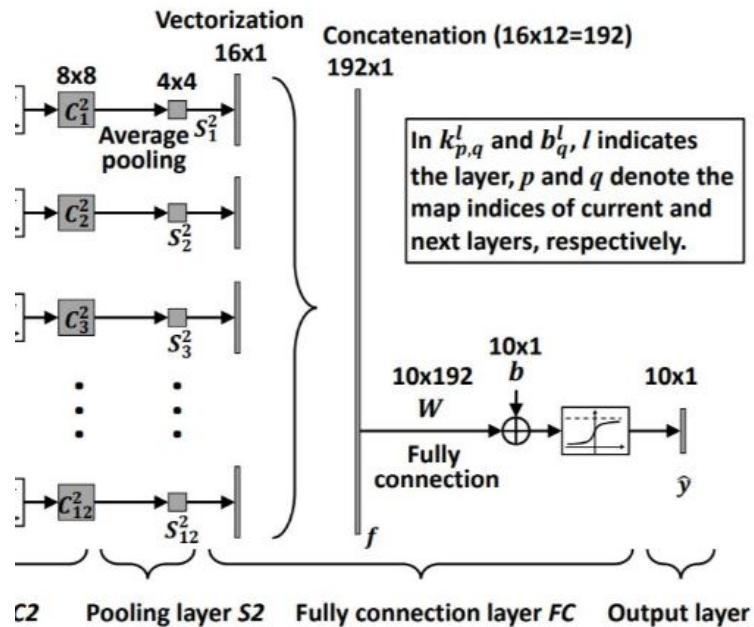
$$C_q^2(i, j) = \sigma \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.11)$$

teda v rovnice 12.11 je jasne že $q = 1, \dots, 12$ pretože máme 12 výstupov z vrstvy $C2$ o veľkosti $8x8$ (parametre i,j). Teda kym v vrstve $C1$ sme mali 6 príznakových polí $24x24$ po poolingu a vrstve $C2$ máme 12 príznakových polí o rozmere $8x8$.

5. vrstva $S2$ a výstupná vrstva

Táto časť CNNSIMPLE sa skladá z

- (a) jednoduchej poolingovej vrstvy $S2$, ktorá priemerovaním dostane z 12 príznakových polí $8x8$ následne 12 príznakových polí $4x4$
- (b) Plne prepojenej vrstvy FC, ktorá sa samotná skladá z
 - vektorizačnej časti, ktorá príznakové pole $4x4$ iba dá do vektoru $16x1$
 - integračnej časti (concatenation), ktorá z 12 vektorov $16x1$ vytvorí jeden vektor $196x1$, lebo $12 \times 16 = 192$
 - plne prepojenej časti, ktorá 192 neurónov plne prepojí na 10 výstupných neurónov
- (c) výstupnej vrstvy ktorá 10 neurónov má sigmodálnu funkciu a prah a je plne napojená na predchádzajúcu $192x1$ vrstvu neurónov



Obr. 12.15: Výstupná časť CNNSIMPLE

Teda samotné matematické vzťahy sú v celku jednoduché :

- vrstva priemerového poolingu S^2

$$S_q^2(i, j) = \frac{1}{4} \left(\sum_{u=0}^{v=1} \sum_{v=0}^{v=1} C_p^2(2i - u, 2j - v) \right) \quad (12.12)$$

kde $i, j = 1, \dots, 4$ a $q = 1, \dots, 12$ lebo C_p^2 vrstva má rozmer $8x8$ a priemerovaným poolingom dostaneme rozmer $4x4$ pre všetkých 12 príznakových polí.

- vektorizačná a integračná vrstva vlastne iba preorganizuje konfiguráciu siete z formácie $12x4x4$ zo vzorca 12.12 do jedného vektora neurónov o veľkosti $1x192$. tento proces označíme ako funkciu

$$f = F \left(\{S_q^2\}_{q=1, \dots, 12} \right) \quad (12.13)$$

kde f je vlastne vektor $f_{k=1, \dots, 192}$. Súčasne môžeme napísat pre tento jednoduchý vzťah 12.13 aj inverznú operáciu teda z $1x192$ späť na formáciu $12x4x4$ a to nasledovne

$$\{S_q^2\}_{q=1, \dots, 12} = F^{-1}(f) \quad (12.14)$$

Z toho teda vyplýva, že ak na vstupe sme mali obrázok $28x28$ to je **784 pixelov** (príznakov), po prvej konvolučnej vrstve $C1$ sme mali $6x24x24$ to jest **3456 hodnôt** príznakov po druhej konvolučnej vrstve $C2$ sme mali $12x8x8$ to jest **768 príznakov**. Všetko toto následne smerovalo k vrstve $S2$, ktorá mala **192 neurónov** (príznakov). Týmto bola ukončená časť hľadania príznakového priestoru a nasleduje výstup z konvolučnej siete, kde sa realizuje klasifikácia dát.

- výstupná vrstva o veľkosti $1x10$ neurónov, ktorá má prepojenie z predchádzajúcej vrstvy o veľkosti $1x192$ neurónov cez sigmoidálne aktivačné funkcie σ a prahy b teda vypočítaný výstup \hat{y} dostaneme ak

$$fc_l = \sum_{k=1}^{k=192} w_{l,k} * f_k + b_l \quad (12.15)$$

kde $l = 1,..10$ a $w_{l,k}$ je hodnota váhy medzi k-tym neurónom v integračnej vrstve (zdrojovým) a l-tým neurónom (cieľovým) vo výstupnej vrstve a fcl je suma súčinov pre jednotlivé výstupy pred aktivačnou funkciou výstupnej vrstvy. Následne teda

$$\sigma = \frac{1}{1 + \exp^{-fc}} \quad (12.16)$$

Vo vektorovom tvare to môžeme napísť ako

$$\hat{y} = \sigma \{(Wx^f) + b\} \quad (12.17)$$

Na záver ak predpokladáme vypočítanú hodnotu na výstupnej vrstve $\hat{y}(l)$ kde $l = 1,..10$ a **očakávanú hodnotu na výstupnej vrstve** označme $y(l)$ na l-tom neuróne, tak potom môžeme definovať našu **chybovú funkciu L (Lost function)** nasledovne cez všetky výstupné neuróny na výstupnej vrstve CNNSIMPLE

$$L = \frac{1}{2} \sum_{l=1}^{l=10} (\hat{y}(l) - y(l))^2 \quad (12.18)$$

teda $\hat{y}(l)$ je vypočítaná hodnota a $y(l)$ je očakávaná hodnota.

Na základe horeuvedeného teda môžeme uzavrieť **dopredný** prechod cez konvolučnú neurónovú sieť CNNSIMPLE.

12.2.2 Parametre CNNSIMPLE pre učenie

Veľmi dôležitá je otázka koľko parametrov budeme adaptovať resp. učiť na tejto konvolučnej sieti.

Teda ide o nasledovné parametre :

1. **parametre filtrov $k_{1,p}^1$** v konvolučnej vrstve $C1$ ide o $5 \times 5 + 1 = 26$ pre každý filter teda máme 6 filtrov **$26 \times 6 = 156$ parametrov** na $C1$. Tieto parametre je potrebné na úvod učenie inicializovať. Na základe teórie je vhodné inicializovať tieto parametre pomocou generátora rovnomenného rozdelenia nasledovnými parametrami

$$k_{1,p}^1 \sim U \left(\pm \sqrt{\frac{6}{(1+5)x5^2}} \right) \quad (12.19)$$

kde $p = 1, \dots, 6$ počet filtrov v $C1$. Táto inicializácia je dôležitá pre prvý výpočet vo vzorci 12.8. Súčasne b_p^1 sú nastavené na hodnotu nula.

2. **parametre filtrov $k_{p,q}^2$** lebo na 6 príznakových platní sa aplikuje konvolučná vrstva s filtrom 5×5 teda pre jednu konvolúciu na 6 príznakových platní máme $5 \times 5 \times 6$ parametrov a ešte prah b teda spolu je to $5 \times 5 \times 6 + 1$ násobené počtom filtrov v konvolučnej vrstve $C2$ teda 12 a výsledok je **$(5 \times 5 \times 6 + 1) \times 12 = 1812$ parametrov** v vrstve $C2$. Tieto parametre je potrebné na úvod učenie inicializovať. Na základe teórie je vhodné inicializovať tieto parametre pomocou generátora rovnomenného rozdelenia nasledovnými parametrami

$$k_{p,q}^2 \sim U \left(\pm \sqrt{\frac{6}{(6+12)x5^2}} \right) \quad (12.20)$$

kde $q = 1, \dots, 12$ počet filtrov v $C2$. Táto inicializácia je dôležitá pre prvý výpočet vo vzorci 12.11. Súčasne b_q^2 sú nastavené na hodnotu nula.

3. na výstupnej vrstve máme **synaptické váhy W** , teda **$192 \times 10 + 10 = 1930$ parametrov**. Tieto parametre je potrebné na úvod učenie inicializovať. Na základe teórie je vhodné inicializovať tieto parametre pomocou generátora rovnomenného rozdelenia nasledovnými parametrami

$$W \sim U \left(\pm \sqrt{\frac{6}{(192+10)}} \right) \quad (12.21)$$

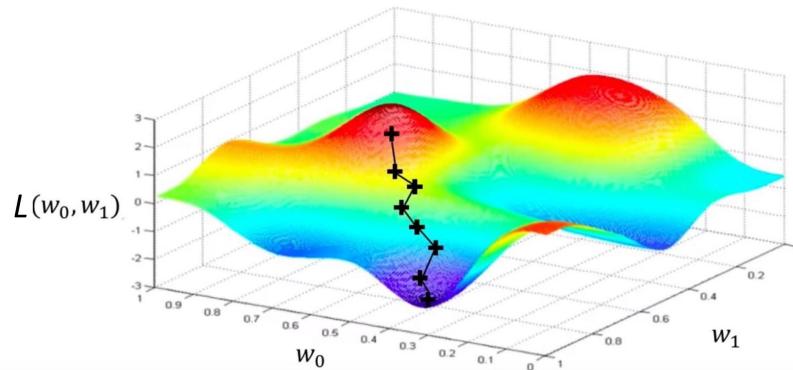
Táto inicializácia je dôležitá pre prvý výpočet vo vzorci 12.15. Súčasne b zo vzorca 12.17 sú nastavené na počiatku výpočtov na hodnotu nula.

Výsledok je teda nasledovný, konvolučná neurónová sieť CNNSIMPLE má spolu

$$\mathbf{156 + 1812 + 1930 = 3898} \quad (12.22)$$

parametrov, ktoré musíme nastaviť. Tu však je treba upozorniť že pôvodný obrázok $28 * 28$ pixelov bol pretrasponovaný do 192 rozmerného vektora. Teda vrstvy C1, S1, C2 a S2 hľadajú nové príznaky a v novom príznakovom priestore sa realizuje klasifikácia.

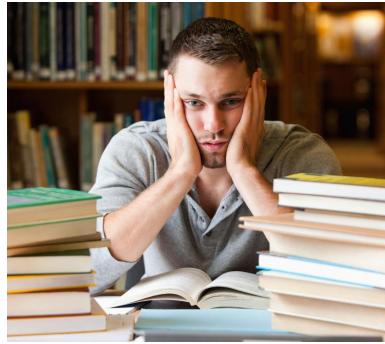
Globálne však my hľadáme aj transformáciu aj diskriminačné hyperplochy v novom príznakovom priestore **SÚBEŽNE** a teda sa **jedna o** $3898 + 1 = 3899$ **rozmerný chybový priestor**, ktorý bude predmetom nášho záujmu s cieľom nájdenia tých správnych resp. optimálnych parametrov, ktoré budú dávať najmenšiu hodnotu chybovej funkcie L definovanej vo vzorci 12.18.



Obr. 12.16: ilustračný obrázok 2 + 1 rozmerného chybového priestoru a výpočtu gradientu na chybovom povrchu s cieľom hľadania globálneho minima chyby L

12.2.3 Metóda spätného šírenia chyby v konvolučnej sieti

Ako je zrejmé z časti 12.2 budeme hľadať ako učiť resp. adaptovať parametre CNNSIMPLE aby sme dosiahli najlepšie výsledky v chybovej funkcií L definovanej v rovnici 12.18. Teraz si odvedieme učiace pravidlá pre jednotlivé parametre, ktoré budeme adaptovať vzhľadom na výstup.



Ak teda zhrnieme budeme nasledovným spôsobom adaptovať parametre učenia vzhľadom na minimalizáciu chybovej funkcie L

$$W(t+1) \leftarrow W(t) + \Delta W(t) \quad (12.23)$$

$$b(t+1) \leftarrow b(t) + \Delta b(t) \quad (12.24)$$

$$b_q^2(t+1) \leftarrow b_q^2(t) + \Delta b_q^2(t) \quad (12.25)$$

$$k_{p,q}^2(t+1) \leftarrow k_{p,q}^2(t) + \Delta k_{p,q}^2(t) \quad (12.26)$$

$$b_p^1(t+1) \leftarrow b_p^1(t) + \Delta b_p^1(t) \quad (12.27)$$

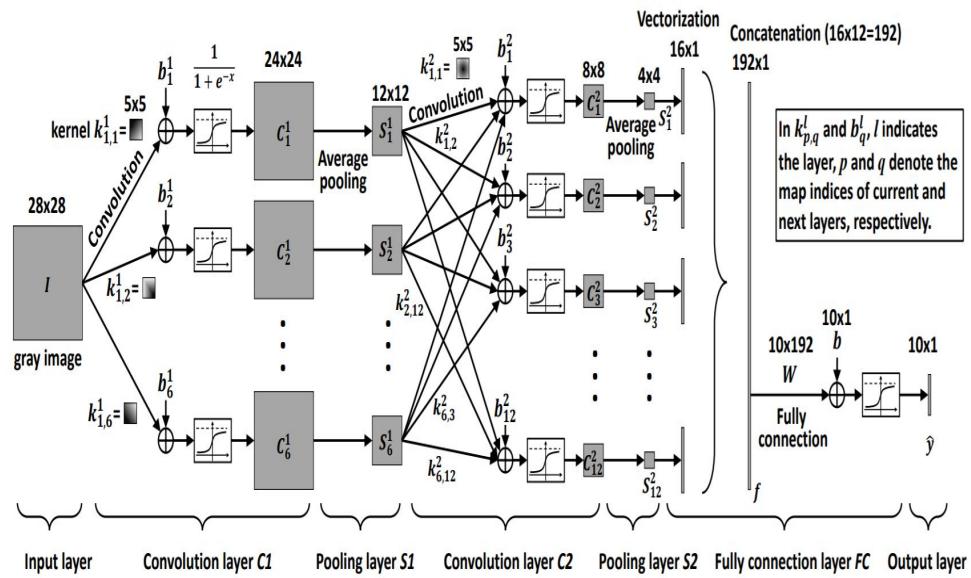
$$k_{1,p}^1(t+1) \leftarrow k_{1,p}^1(t) + \Delta k_{1,p}^1(t) \quad (12.28)$$

V nasledovných častiach si ukážeme matematické pozadie na výpočet

$\Delta \mathbf{W}, \Delta \mathbf{b}, \Delta \mathbf{b}_q^2, \Delta \mathbf{k}_{p,q}^2, \Delta \mathbf{b}_p^1, \Delta \mathbf{k}_{1,p}^1$

a tým aj následne nových hodnôt podľa horeuvedených vzorcov.

Pre pripomenutie si znova ukážeme konvolučnú neurónovú sieť, kde budeme tieto adaptácie počítať na základe výstupnej funkcie L 12.18.



Obr. 12.17: pripomenutie konvolučnej neurónovej siete

12.2.4 Učenie parametrov výstupnej časti siete

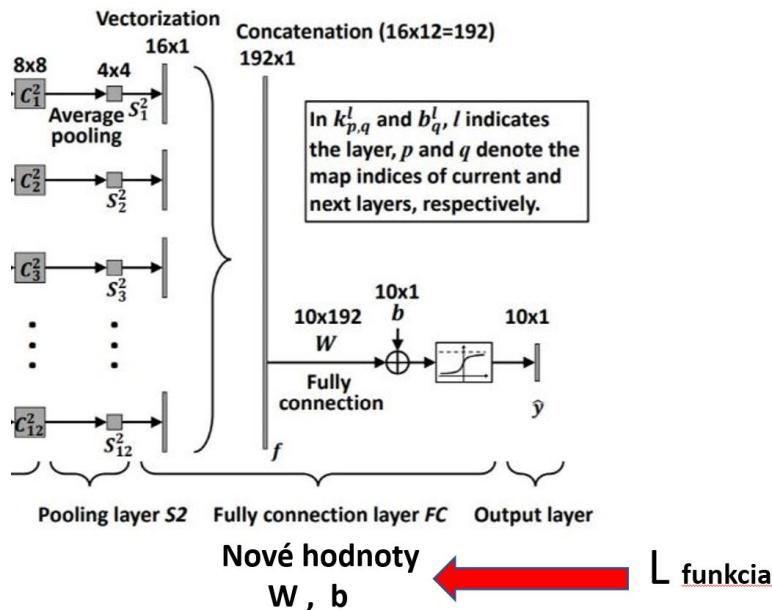
Vo výstupnej časti CNNSIMPLE ako je naznačené na obrázku 12.19 adaptujeme parameter \mathbf{W} (synaptické váhy, 192×10), a \mathbf{b} hodnoty biasu (prahu, 10×1) znázornené na obrázku 12.19.

Teda adaptačné rovnice budú nasledovné :

1. pre zmeny synaptickej váhy \mathbf{W} vo výstupnej časti

$\Delta \mathbf{W}$, pre 192×10 prepojení, l je počet neurónov na výstupe z intervalu $l \in <1, 10>$, k je počet neurónov na integračnej vrstve teda $k \in <1, 192>$

$$\Delta w_{l,k} = \frac{\partial L}{\partial w_{l,k}} \quad (12.29)$$



Obr. 12.18: prvá časť spätného šírenia chyby

$$\frac{\partial L}{\partial w_{l,k}} = \frac{\partial L}{\partial \hat{y}^l} \cdot \frac{\partial \hat{y}^l}{\partial w_{l,k}} \quad (12.30)$$

následne podľa vzorca 12.18 a vzorca 12.15 dostávame nasledovné matematické popisy predchádzajúceho vzorca. Skúsme odvodiť prvý člen pravej strany vzorca 12.30 najprv nahradíme L funkciu

$$\frac{\partial L}{\partial \hat{y}^l} = \frac{1}{2} \frac{\partial}{\partial \hat{y}^l} \sum_{l=1}^{l=10} (\hat{y}^l - y^l)^2 \quad (12.31)$$

teraz ju zderivujeme pod \hat{y}^l

$$\frac{1}{2} \frac{\partial}{\partial \hat{y}^l} \sum_{l=1}^{l=10} (\hat{y}^l - y^l)^2 = 2 \cdot \frac{1}{2} \cdot (\hat{y}^l - y^l) \cdot 1 \quad (12.32)$$

$$2 \cdot \frac{1}{2} \cdot (\hat{y}^l - y^l) \cdot 1 = (\hat{y}^l - y^l) \quad (12.33)$$

teda prvá časť pravej strany vzorca [12.30](#) má riešenie

$$\frac{\partial L}{\partial \hat{y}^l} = (\hat{y}^l - y^l) \quad (12.34)$$

teraz prejdeme na úpravu druhej časti pravej strany vzorca [12.30](#). Teda podľa vzorcov [12.17](#), [12.16](#) a [12.15](#) môžeme napísať nasledovné

$$\frac{\partial \hat{y}^l}{\partial w_{l,k}} = \frac{\partial}{\partial w_{l,k}} \sigma \left(\sum_{k=1}^{k=192} w_{l,k} \cdot f_k + b_l \right) \quad (12.35)$$

V horeuvedenom vzorci je σ sigmoidálna funkcia s argumentom.

V nasledovných riadkoch si ukážeme postup derivácie sigmoidálnej funkcie. Tento princíp budeme používať ak v ďalších častiach spätného šírenia chyby.

Kde σ je sigmoidalna funkcia definovaná v rovnici 12.16. Derivácia tejto funkcie má všeobecný tvar

$$\frac{\partial}{\partial x} \left\{ \frac{1}{1 + e^{-x}} \right\} = \frac{e^{-x}}{(1 + e^{-x})^2} \cdot \partial x \quad (12.36)$$

teda ak $x = fc$ zo vzorca 12.15 môžeme podľa vzorca 12.36 napísť nasledovnú matematickú úpravu derivácie (do menovateľa výsledku derivácie pripočítame 1 a následne odpočítame 1).

$$\begin{aligned} \frac{e^{-x}}{(1 + e^{-x})^2} \cdot \partial x &= \frac{1 + e^{-x} - 1}{(1 + e^{-x})^2} \cdot \partial x = \\ &= \left\{ \frac{1 + e^{-x}}{(1 + e^{-x})^2} - \frac{1}{(1 + e^{-x})^2} \right\} \cdot \partial x \end{aligned} \quad (12.37)$$

toto môžeme upraviť lebo

$$\frac{1 + e^{-x}}{(1 + e^{-x})^2} = \frac{1}{(1 + e^{-x})} \quad (12.38)$$

a teda dostávame

$$\begin{aligned} \left\{ \frac{1}{(1 + e^{-x})} - \frac{1}{(1 + e^{-x})^2} \right\} \cdot \partial x &= \\ &= \frac{1}{(1 + e^{-x})} \left\{ 1 - \frac{1}{(1 + e^{-x})} \right\} \cdot \partial x \end{aligned} \quad (12.39)$$

teda ak $x = fc$, kde fc je hodnota neurónu vo vrstve f (192 x 1) podľa vzorca 12.15 a súčasne podľa 12.17 môžeme prepísať vzorec 12.39 do tvaru

$$\frac{1}{(1 + e^{-fc})} \left\{ 1 - \frac{1}{(1 + e^{-fc})} \right\} \cdot \partial fc = \hat{y} \cdot (1 - \hat{y}) \cdot \partial fc \quad (12.40)$$

V poslednom vzorci už len vypočítame deriváciu

$$\frac{\partial fc}{\partial w_{l,k}}$$

teda ak fc je podľa vzorca 12.15 tak môžeme napísať

$$\frac{1}{\partial w_{l,k}} \cdot \partial \left(\sum_{kk=1}^{kk=192} w_{l,k} \cdot f_{kk} + b_l \right) = f_k \quad (12.41)$$

kde f_l je jeden neurón z vrstvy f. Teda pre výpočet druhej časti pravej strany vzorca 12.30 môžeme napísať

$$\frac{\partial \hat{y}^l}{\partial w_{l,k}} = \hat{y} \cdot (1 - \hat{y}) \cdot f_k \quad (12.42)$$

Teda výsledný vzorec podla 12.34, 12.42 nasledovne pre adaptáciu váhy z integračnej vrstvy k výstupným neurónom nasledovne :

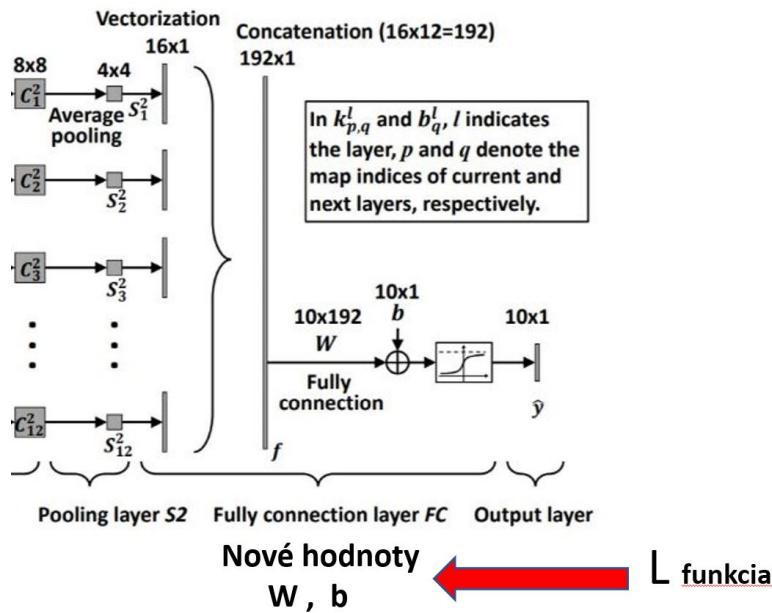
$$\frac{\partial L}{\partial w_{l,k}} = (\hat{y}^l - y^l) * \hat{y}^l \cdot (1 - \hat{y}^l) \cdot f_k \quad (12.43)$$

Teda pre synaptickú váhu idúcu od neurónu k v integrálnej vrstve k neurónu l v výstupnej vrstve vypočítame adaptáciu nasledovne

$$\Delta w_{l,k} = -\gamma \frac{\partial L}{\partial w_{l,k}} = -\gamma (\hat{y}^l - y^l) \cdot \hat{y}^l \cdot (1 - \hat{y}^l) \cdot f_k \quad (12.44)$$

a následne vypočítame novú hodnotu váhy $w_{l,k}$ nasledovne v zmysle vzorca 12.23

$$\boxed{\mathbf{w}_{l,k}(t+1) = \mathbf{w}_{l,k}(t) + \Delta \mathbf{w}_{l,k}(t)} \quad (12.45)$$



Obr. 12.19: prvá časť spätného šírenia chyby

- adaptačné pravidlo pre zmeny prahov (bias) b na výstupných neurónoch vo výstupnej časti

Teda zmenu prahu pre l -ty výstupný neurón vypočítame ako

$$\Delta b^l = \frac{\partial L}{\partial b^l} \quad (12.46)$$

teda

$$\frac{\partial L}{\partial b^l} = \frac{\partial L}{\partial y^l} \cdot \frac{\partial y^l}{\partial b^l} \quad (12.47)$$

teda prvá časť pravej strany vzorca 12.47 je známa z predchádzajúcich výpočtov podľa vzorca 12.34 a to nasledovne

$$\frac{\partial L}{\partial y^l} = (\hat{y}^l - y^l) \quad (12.48)$$

ostáva nám riešiť druhú časť vzorca 12.47 obdobne ako v pri synaptických váhach hlavne podľa vzorca 12.40 a súčasne vo vzorci 12.41

$$\frac{\partial \hat{y}^l}{\partial b^l} = \frac{\partial}{\partial b^l} \cdot \left(\sum_{kk=1}^{kk=192} w_{l,kk} \cdot f_{kk} + b^l \right) = 1 \quad (12.49)$$

$$\frac{\partial \hat{y}^l}{\partial b^l} = \hat{y} \cdot (1 - \hat{y}) \cdot 1 \quad (12.50)$$

teda výsledok má tvar

$$\frac{\partial L}{\partial b^l} = (\hat{y}^l - y^l) \cdot \hat{y}^l \cdot (1 - \hat{y}^l) \cdot 1 \quad (12.51)$$

z toho vyplýva že prah na l-tom výstupnom neuróne sa bude adaptovať nasledovne

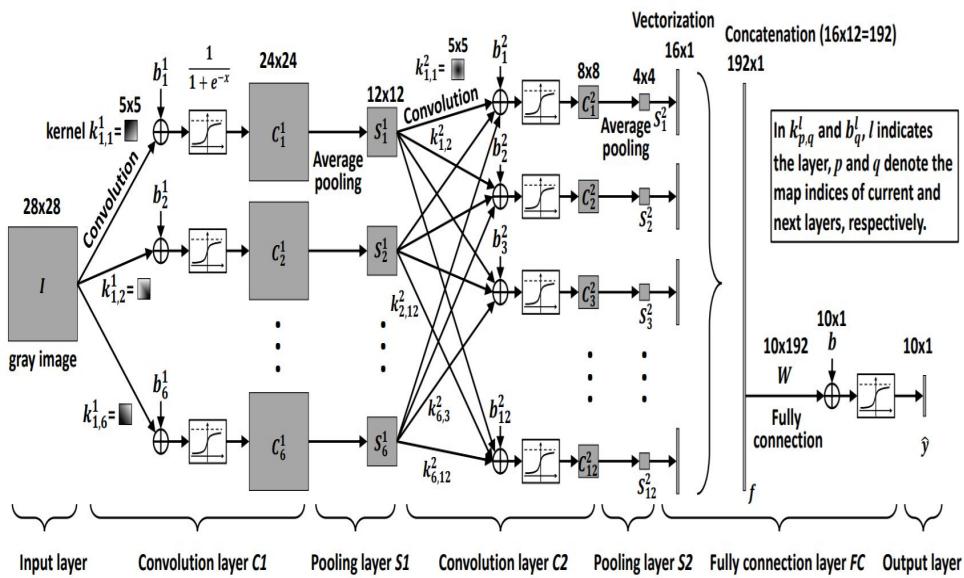
$$\Delta b^l = -\gamma \frac{\partial L}{\partial b^l} = -\gamma (\hat{y}^l - y^l) \cdot \hat{y}^l \cdot (1 - \hat{y}^l) \cdot 1 \quad (12.52)$$

a následne vypočítame novú hodnotu váhy b^l nasledovne v zmysle vzorca 12.52

$$\boxed{\mathbf{b}^l(\mathbf{t} + 1) = \mathbf{b}^l(\mathbf{t}) - \Delta \mathbf{b}^l(\mathbf{t})} \quad (12.53)$$

12.2.5 Inverzný výpočet zmien príznakových f , S^2 a C^2 pri spätnom šírení chyby (upsampling)

Pre pripomenutie si znova ukažeme konvolučnú neurónovú sieť, kde budeme tieho adaptácie počítať na základe výstupnej funkcie L .



Obr. 12.20: pripomenutie konvolučnej neurónovej siete

Ak sa pozriete na celkovú topológiu CNNSIMPLE na obrázku 12.11 na horeuvedenom obrázku tak máme na vstupe obraz I o rozmeroch $28 * 28$ následne máme príznakové pole $C^1_{p=1..6}$ o rozmeroch $24 * 24$, potom $S^1_{p=1..6}$ o rozmeroch $12 * 12$. Následne máme príznakové pole $C^2_{q=1..12}$ o rozmeroch $8 * 8$, potom ďalšie $S^2_{q=1..12}$ o rozmere $4 * 4$. A konečne máme posledné príznakové pole f o rozmere $192 * 1$. Teda ak ideme spočítať adaptáciu $k^2_{p,q}$ tak k tomu potrebujeme zmeny na príznakových poliach $f, S^2_{p,q}, C^2_{p,q}$ v závislosti od L – funkcie na výstupe z neurónovej siete.

Táto časť spätného šírenia neobsahuje výpočet adaptačných pravidiel ale iba **jednoduchý prepočet**. Teda je to len výpočet a nie odvodenie adaptačného pravidla. Teda opäť pre výpočet

$$\Delta f^k = -\gamma \frac{\partial L}{\partial f^k} \quad (12.54)$$

kde $k = 1, \dots, 192$ v integračnej vrstve a γ je parameter rýchlosťi

$$\frac{\partial L}{\partial f^k} = \sum_{l=1}^{l=10} \frac{\partial L}{\partial \hat{y}^l} \cdot \frac{\partial \hat{y}^l}{\partial f^k} \quad (12.55)$$

teraz prvý člen pravej strany má tvar

$$\frac{\partial L}{\partial \hat{y}^l} = (\hat{y}^l - y^l) \quad (12.56)$$

teraz druhý člen pravej strany má tvar

$$\frac{\partial \hat{y}^l}{\partial f^k} = \frac{\partial}{\partial f^k} \left(\sum_{kk=1}^{kk=192} w_{l,kk} \cdot f_{kk} + b^l \right) = w_{l,k} \quad (12.57)$$

$$\frac{\partial \hat{y}^l}{\partial b^l} = \hat{y}^l \cdot (1 - \hat{y}^l) \cdot w_{l,k} \quad (12.58)$$

následne druhý člen pravej strany

$$\frac{\partial L}{\partial f^k} = \sum_{l=1}^{l=10} (\hat{y}^l - y^l) \cdot \hat{y}^l \cdot (1 - \hat{y}^l) \cdot w_{l,k} \quad (12.59)$$

z toho vyplýva že prah na k-tom ($k = 1, \dots, 192$) neuróne sa bude výpočet nasledovný

$$\Delta f^k = -\alpha \frac{\partial L}{\partial f^k} = \sum_{l=1}^{l=10} (\hat{y}^l - y^l) \cdot \hat{y}^l \cdot (1 - \hat{y}^l) \cdot w_{l,k} \quad (12.60)$$

Tento výstup Δf^k o veľkosti ($k = 1, \dots, 192$) následne vstupuje do rovnice podľa už spomenutých prepočtov 12.13 resp. 12.14 pre konfiguráciu a vytvorenie príznakových formácií S_q^2 nasledovne

$$\{\Delta S_q^2\}_{q=1, \dots, 12} = F^{-1}(\Delta f) \quad (12.61)$$

kde vlastne funkcia F v predchádzajúcej rovnici symbolicky predstavuje spätnú reorganizáciu vrstvy f do 12 tich príznakových polí S_q^2 . Táto funkcia

F vznikala pri prvom doprednom prechode, keď sa z 12 tých príznakových polí o rozmere 4×4 teda $16 \times 1 \times 12 = 192$ vektorizovala vrstva f (pozri obrázok 12.21).

Následne urobíme inverznú operáciu k priemerovanému poolingu tzv. "upscale". Teda z príznakových formácií 4×4 vytvoríme príznakové formácie 8×8 teda hľadáme spätnú rovnicu k rovnici 12.12.

Súčasne si zavedieme špeciálnu funkciu $\alpha(i, j)$ ktorá bude prepočítavať spätné pozície pri upscalingu z S_q^2 príznakovnej formácie do C_q^2 príznakovnej formácie. Teda z 4×4 do 8×8 . Pre zmenu v príznakovnej formácii C_q^2 môžeme napísat nasledovné

$$\Delta C_q^2(i, j) = \frac{1}{4} \Delta S_q^2 \left(\alpha\left(\frac{i}{2}, \frac{j}{2}\right) \right), i, j = 1, 2, \dots, 8 \quad (12.62)$$

pre $q = 1, \dots, 12$. Aby sme pochopili vzorec 12.62 musíme indexy zaokruhlovať teda ak index i alebo j budu mať hodnoty ako napr.

$$\begin{array}{ll} \frac{1}{2} = 1 & \frac{2}{2} = 1 \\ \frac{3}{2} = 2 & \frac{4}{2} = 2 \\ \frac{5}{2} = 3 & \frac{6}{2} = 3 \\ \frac{7}{2} = 4 & \frac{8}{2} = 4 \end{array} \quad (12.63)$$

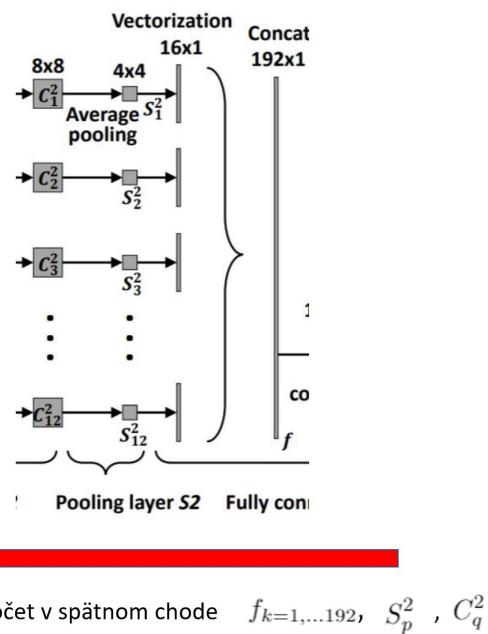
teda podľa tohto predpisu zaokrúhlňovania indexov dostaneme

$$\begin{aligned} \Delta C_q^2(1, 1) &= \frac{1}{4} \Delta S_q^2 \left(\alpha(1, 1) \right) & i = 1, j = 1 \\ \Delta C_q^2(1, 2) &= \frac{1}{4} \Delta S_q^2 \left(\alpha(1, 1) \right) & i = 1, j = 2 \\ \Delta C_q^2(2, 1) &= \frac{1}{4} \Delta S_q^2 \left(\alpha(1, 1) \right) & i = 2, j = 1 \\ \Delta C_q^2(2, 2) &= \frac{1}{4} \Delta S_q^2 \left(\alpha(1, 1) \right) & i = 2, j = 2 \end{aligned} \quad (12.64)$$

obdobne aj pre všetky indexy $i = 1 \dots 8$ resp. $j = 1 \dots 8$ realizujeme upscaling zo 4×4 na 8×8 podľa predpisu 12.62 a dostáva teda napr.

$$\begin{aligned}
 \Delta C_q^2(1, 3) &= \frac{1}{4} \Delta S_q^2 \left\{ \alpha(1, 2) \right\} \quad i = 1, j = 3 \\
 \Delta C_q^2(1, 4) &= \frac{1}{4} \Delta S_q^2 \left\{ \alpha(1, 2) \right\} \quad i = 1, j = 4 \\
 \Delta C_q^2(2, 3) &= \frac{1}{4} \Delta S_q^2 \left\{ \alpha(1, 2) \right\} \quad i = 2, j = 3 \\
 \Delta C_q^2(2, 4) &= \frac{1}{4} \Delta S_q^2 \left\{ \alpha(1, 2) \right\} \quad i = 2, j = 4
 \end{aligned} \tag{12.65}$$

Týmto máme vytvorené všetky predpoklady aby sme začali počítať zmeny pre konvolučnú vrstvu C^2 a to výpočtom zmien parametrov $\mathbf{k}_{\mathbf{p}, \mathbf{q}}^2$.



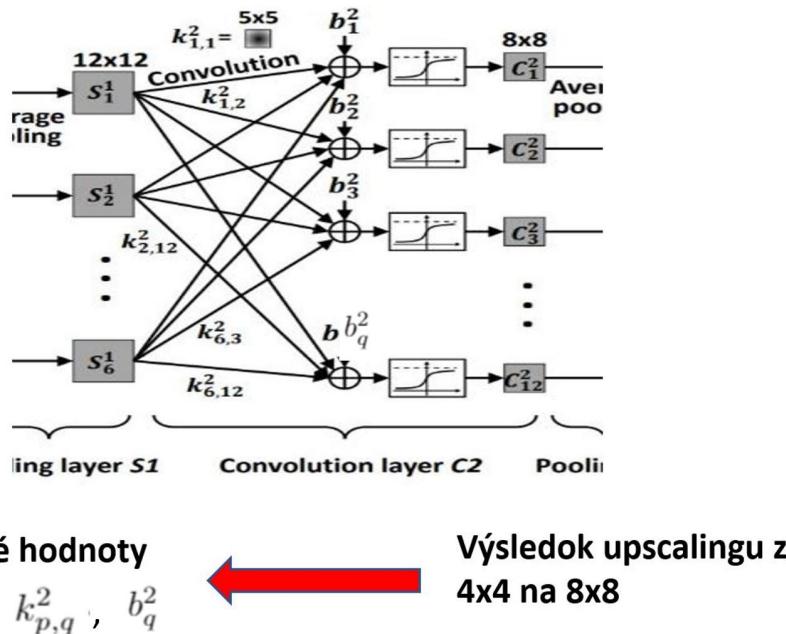
Obr. 12.21: Prepočet nových hodnôt pre f , $S2$ a $C2$ pri spätnom šírení chyby

12.2.6 Učenie (adaptácia) parametrov druhej konvolučnej vrstvy filtra $k_{p,q}^2$ pri spätnom šírení z C_q^2

Vychádzame zo vzorca

$$\Delta k_{p,q}^2(u, v) = -\gamma \frac{\partial L}{\partial k_{p,q}^2(u, v)} \quad (12.66)$$

kde $p = 1,..6$ a $q = 1,..12$ a taktiež $u = 1,..5$ a $v = 1..5$ (rozmery filtra) a samozrejme L je Loos funkcia na výstupe siete podľa 12.18 definovanej na strane 186.



Obr. 12.22: druhá časť spätného šírenia chyby

teda následne

$$\frac{\partial L}{\partial k_{p,q}^2(u, v)} = \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \frac{\partial L}{\partial C_q^2(i, j)} \frac{\partial C_q^2(i, j)}{\partial k_{p,q}^2(u, v)} \quad (12.67)$$

prvú časť pravej strany vzorca 12.67 označíme ako výsledok vzorca 12.62 teda

$$\frac{\partial L}{\partial C_q^2(i, j)} = \Delta C_q^2(i, j) \quad (12.68)$$

následne prepíšme druhú časť vzorca 12.67 pomocou vzorca 12.11 pre $C_q^2(i, j)$ do nasledovného tvaru :

$$\frac{\partial C_q^2(i, j)}{\partial k_{p,q}^2(u, v)} = \frac{\partial}{\partial k_{p,q}^2(u, v)} \sigma \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.69)$$

Tu treba pripomenúť že vo vzorci 12.69 je sigmoidálna funkcia σ definovaná o vzorci 12.10 a jej derivácia je vo vzorci 12.36 a argument sigmoidálnej funkcie je teda

$$x = \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.70)$$

a podľa 12.36 a podľa výpočtov pre $C_q^2(i, j)$ opakovane napíšeme vzorec 12.11

$$C_q^2(i, j) = \sigma \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.71)$$

tak potom môžeme napísat

$$\begin{aligned} \frac{\partial C_q^2(i, j)}{\partial k_{p,q}^2(u, v)} &= C_q^2(i, j)(1 - C_q^2(i, j)) \cdot \\ &\quad \frac{\partial}{\partial k_{p,q}^2(u, v)} \left\{ \sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right\} \end{aligned} \quad (12.72)$$

kde na pravej strane vzorca derivácia v druhej časti má tvar

$$\frac{\partial}{\partial k_{p,q}^2(u, v)} \left\{ \sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right\} = S_p^1(i-u, j-v) \quad (12.73)$$

teda môžeme napísať podľa výsledok pre vzorec 12.72

$$\frac{\partial C_q^2(i, j)}{\partial k_{p,q}^2(u, v)} = C_q^2(i, j)(1 - C_q^2(i, j))S_p^1(i - u, j - v) \quad (12.74)$$

čo je druhá časť pravej strany vzorca 12.67. Teda môžeme výsledok napísať pre adaptáciu $\mathbf{k}_{p,q}^2$ nasledovne

$$\Delta k_{p,q}^2(u, v) = -\gamma \frac{\partial L}{\partial k_{p,q}^2(u, v)} = -\gamma \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \Delta C_q^2(i, j)C_q^2(i, j)(1 - C_q^2(i, j))S_p^1(i - u, j - v) \quad (12.75)$$

teraz by sme mohli skončiť ale skúsme urobiť úpravy predošlého vzorca tak aby to bolo krajšie resp. výpočtovo výhodnejšie. Označme ako $C_{q,\beta}^2$ ako argument sigmoidalnej funkcie C_q^2 a tvrdíme, že môžeme napísať nasledovnú časť pravej strany predošej rovnice

$$\Delta C_{q,\beta}^2(i, j) = \Delta C_q^2(i, j)C_q^2(i, j)(1 - C_q^2(i, j)) \quad (12.76)$$

teraz si rozpíšme časti pravej strany predchádzajúceho vzorca za pomoci vzorca 12.11. Teda

$$C_q^2(i, j) = \sigma \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i - u, j - v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.77)$$

a taktiež podľa 12.62

$$\Delta C_q^2(i, j) = \frac{1}{4} \cdot \Delta S_q^2 \{ \alpha(i, j) \}, i, j = 1, 2, \dots 8 \quad (12.78)$$

Kde $\alpha(i, j)$ je upscaling funkcia - teda inverzná funkcia poolingu a ak teda zapíšeme príznakové pole S_p^1 tak že ho zrotujeme ale referenčne zmeníme indexy môžeme dostať

$$S_{p,rot180}^1(u - i, v - j) = S_p^1(i - u, j - v) \quad (12.79)$$

Následne teda môžeme prepísať výsledný vzorec 12.75 do tvaru

$$\Delta k_{p,q}^2(u, v) = \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \Delta C_{q,\beta}^2(i, j) \cdot S_{p,rot180}^1(u - i, v - j) \quad (12.80)$$

ktorý popisuje **konvolučný vzťah** medzi príznakovými platňami $C_{q,\beta}^2$ a $S_{p,rot180}^1$. Porovnajte vzťahy 12.75 a 12.80.

Teda záverečný iteračný resp. adaptačný vzorec v zmysle vzorca 12.26 je teda nasledovný

$$\boxed{k_{p,q}^2(u, v) \{t+1\} = k_{p,q}^2(u, v) \{t\} - \Delta k_{p,q}^2(u, v) \{t\}} \quad (12.81)$$

12.2.7 Výpočet adaptačného pravidla pre prah Δb_q^2 v konvolučnej vrstve C2

postup je veľmi podobný ako v predchádzajúcom prípade teda

$$\Delta b_q^2 = \gamma \frac{\partial L}{\partial b_q^2} \quad (12.82)$$

čo znamená následne

$$\frac{\partial L}{\partial b_q^2} = \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \frac{\partial L}{\partial C_q^2(i, j)} \cdot \frac{\partial C_q^2(i, j)}{\partial b_q^2} \quad (12.83)$$

a znova teda platí pre prvý člen pravej strany vzorec 12.68

$$\frac{\partial L}{\partial C_q^2(i, j)} = \Delta C_q^2(i, j) \quad (12.84)$$

následne prepíšme druhú časť vzorca 12.83 pomocou vzorca 12.11 pre $C_q^2(i, j)$ do nasledovného tvaru :

$$\frac{\partial C_q^2(i, j)}{\partial b_q^2} = \frac{\partial}{\partial b_q^2} \sigma \left(\sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right) \quad (12.85)$$

kde výsledkom tejto parciálnej derivácie vzhľadom na fakt že ide tu o deriváciu sigmoidálnej funkcie postupujeme obdobne ako odvádzaní adaptačného pravidla pre $\mathbf{k}_{p,q}^2$ podľa 12.72

$$\begin{aligned} \frac{\partial C_q^2(i, j)}{\partial b_q^2} &= C_q^2(i, j)(1 - C_q^2(i, j)) \cdot \\ &\quad \frac{\partial}{\partial b_q^2} \cdot \left\{ \sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right\} \end{aligned} \quad (12.86)$$

a následne môžeme napísť

$$\frac{\partial}{\partial b_p^2} \cdot \left\{ \sum_{p=1}^{p=6} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i-u, j-v) \cdot k_{p,q}^2(u, v) + b_p^2 \right\} = 1 \quad (12.87)$$

Teda na základe tohto môžeme napísť s podporou vzorcov ??, 12.84, 12.86 a konečne vzorca 12.87 nasledovný vzťah :

$$\Delta b_p^2 = \gamma \frac{\partial L}{\partial b_p^2} = \gamma \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \Delta C_q^2(i, j) \cdot C_q^2(i, j) (1 - C_q^2(i, j)) \cdot 1 \quad (12.88)$$

lenže tento posledný vzťah môžeme podľa vzorca 12.76 zapísať ako

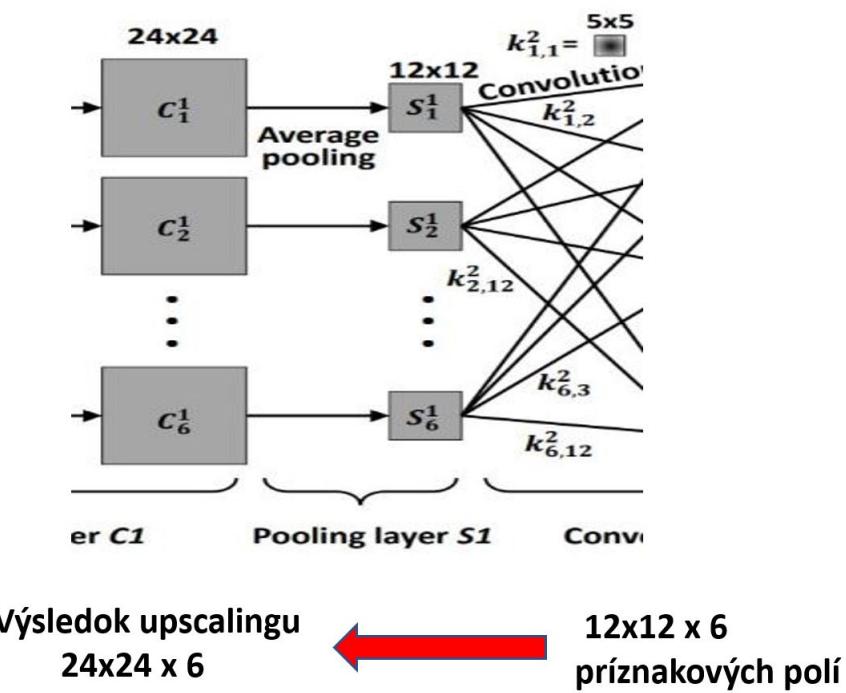
$$\Delta b_p^2 = -\gamma \frac{\partial L}{\partial b_p^2} = -\gamma \sum_{i=1}^{i=8} \sum_{j=1}^{j=8} \Delta C_{q,\beta}^2(i, j) \quad (12.89)$$

kde $C_{q,\beta}^2$ je argumentom sigmodalnej funkcie σ teda samotná konvolúcia. Následne samotný iteračný a adaptačný vzťah bude pre b_p^2 vyzerať následovne

$$\boxed{\mathbf{b}_p^2 \{t + 1\} = \mathbf{b}_p^2 \{t\} - \Delta \mathbf{b}_p^2 \{t\}} \quad (12.90)$$

12.2.8 Prepočet upscalingu v rámci spätného šírenia chyby

Táto vrstva vrámcí celej CNNSIPLE je medzi dvoma konvolučnými vrstvami a musíme znova vypočítať najprv hodnoty ΔS_p^1 následne ΔC_p^1 príznakových polí. Tieto budú následne použité v ďalšej časti na výpočet k_p^1 pre $p = 1, \dots, 6$.



Obr. 12.23: tretia časť spätného šírenia chyby

teda môžeme napísat

$$\begin{aligned}
\Delta S_p^1(i, j) &= \frac{\partial L}{\partial S_p^1(i, j)} \\
&= \sum_{p=1}^{p=12} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} \frac{\partial L}{\partial C_{q,\beta}^2(i+u, j+v)} \frac{\partial C_{q,\beta}^2(i+u, j+v)}{\partial S_p^1(i, j)} \\
&= \sum_{p=1}^{p=12} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} \Delta C_{q,\beta}^2(i+u, j+v). \\
&\cdot \frac{\partial}{\partial S_p^1(i, j)} \left\{ \sum_{p=1}^{p=12} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} S_p^1(i, j).k_{p,q}^2(u, v) + b_p^2 \right\} \\
&= \sum_{p=1}^{p=12} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} \Delta C_{q,\beta}^2(i+u, j+v).k_{p,q}^2(u, v) \quad (12.91)
\end{aligned}$$

ak teda plati že

$$k_{p,q,rot180}^2(-u, -v) = k_{p,q}^2(u, v) \quad (12.92)$$

potom môžeme napísat'

$$\Delta S_p^1(i, j) = \sum_{p=1}^{p=12} \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} \Delta C_{q,\beta}^2(i - (-u), j - (-v)).k_{p,q,rot180}^2(-u, -v) \quad (12.93)$$

Teda vo všeobecnom tvare môžeme napísat'

$$\Delta S_p^1 = \sum_{p=1}^{p=12} \Delta C_{q,\beta}^2 \cdot k_{p,q,rot180}^2 \quad (12.94)$$

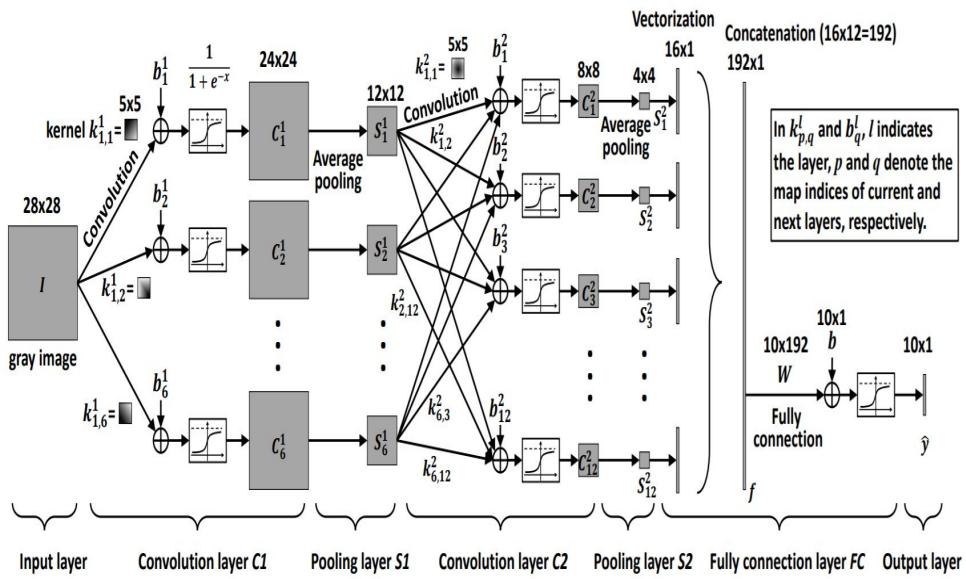
Na základe predošej rovnice môžeme teda dostať uscalingu vrstvu C_p^1 (obdobne ako vo vzorci 12.62 resp. pochopenie prepočtu indexov na strane 199) a to nasledovne

$$\Delta C_q^2(i, j) = \frac{1}{4} \Delta S_q^2 \left(\alpha \left(\frac{i}{2}, \frac{j}{2} \right) \right), i, j = 1, 2, \dots, 24 \quad (12.95)$$

Teda máme potrebné ΔC_q^1 a ΔS_p^1 aby sme mohli následne vypočítať $k_{1,p}^1$,

12.2.9 Učenie parametrov filtra $k_{1,p}^1$ z prvej konvolučnej vrstvy C_p^1

Pre pripomnenie si znova ukážeme konvolučnú neurónovú sieť, kde budeme tieto adaptácie počítať na základe výstupnej funkcie L .



Obr. 12.24: pripomnenie konvolučnej neurónovej siete

$$\Delta k_{1,p}^1(u, v) = \frac{\partial L}{\partial k_{1,p}^1(u, v)} \quad (12.96)$$

$$\frac{\partial L}{\partial k_{1,p}^1(u, v)} = \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \frac{\partial L}{\partial C_p^1(i, j)} \cdot \frac{\partial C_p^1(i, j)}{\partial k_{1,p}^1(u, v)} \quad (12.97)$$

prvú časť pravej strany vzorca 12.97 označíme ako výsledok vzorca 12.95 teda

$$\frac{\partial L}{\partial C_p^1(i, j)} = \Delta C_p^1(i, j) \quad (12.98)$$

následne prepíšme druhú časť vzorca 12.67 pomocou vzorca 12.11 pre $C_p^1(i, j)$ do nasledovného tvaru :

$$\begin{aligned}
& \frac{\partial C_p^1(i, j)}{\partial k_{1,p}^1(u, v)} = \\
& = \frac{\partial}{\partial k_{1,p}^1(u, v)} \sigma \left(\sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} I(i-u, j-v) \cdot k_{1,p}^1(u, v) (u, v) + b_p^1 \right) \\
& = C_p^1(i, j) (1 - C_p^1(i, j)) \cdot I(i-u, j-v) \quad (12.99)
\end{aligned}$$

Ked' porovnáte tretí riadok predošej rovnice 12.108 s rovnicou pri odvádzaní adaptácie $k_{p,q}^2(u, v)$ 12.75 tak je to podobný vzorec ale namiesto príznakov0ho panela S_p^1 tu vystupuje už vstupný obraz $I(i, j)$ ako je zrejmé z obrázku 12.25.

Teraz spojme obe časti pravej strany vzorca 12.96 resp. 12.97 do tvaru

$$\frac{\partial L}{\partial k_{1,p}^1(u, v)} = \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \Delta C_p^1(i, j) C_p^1(i, j) (1 - C_p^1(i, j)) \cdot I(i-u, j-v) \quad (12.100)$$

Ak označíme v predchádzajúcom vzorci

$$\Delta C_{p,\beta}^1(i, j) = \Delta C_p^1(i, j) \cdot C_p^1(i, j) (1 - C_p^1(i, j)) \quad (12.101)$$

ako argument sigmoidálnej funkcie σ tak potom môžeme napísat

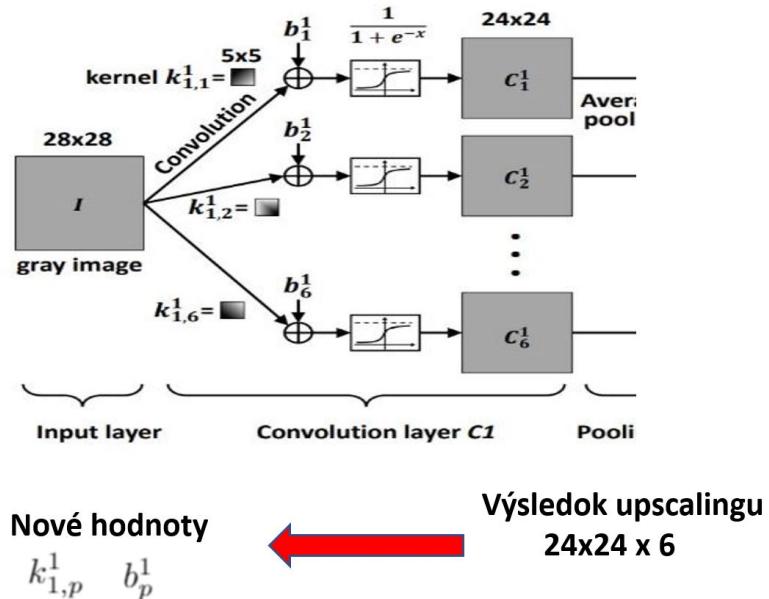
$$\frac{\partial L}{\partial k_{1,p}^1(u, v)} = \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \Delta C_{p,\beta}^1(i, j) \cdot I(i-u, j-v) \quad (12.102)$$

resp. podľa výhodnejšieho zápisu pri priznaní 180 stupňovej rotácie na obraze dostaneme

$$\Delta k_{1,p}^1(u, v) = -\gamma \frac{\partial L}{\partial k_{1,p}^1(u, v)} = -\gamma \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} I_{rot180}(u-i, v-j) \cdot \Delta C_{p,\beta}^1(i, j) \quad (12.103)$$

Teda dostávame záverečný vzorec adaptácie pre $k_{1,p}^1$ na prvej konvolučnej vrstve

$$\boxed{\mathbf{k}_{1,p}^1(\mathbf{u}, \mathbf{v}) \{t+1\} = \mathbf{k}_{1,p}^1(\mathbf{u}, \mathbf{v}) \{t\} - \Delta \mathbf{k}_{1,p}^1(\mathbf{u}, \mathbf{v}) \{t\}} \quad (12.104)$$



Obr. 12.25: posledná časť spätného šírenia chyby

12.2.10 Učenie parametrov b_p^1 prvej konvolučnej vrstvy

Tento postup je veľmi podobný ako v prípade b_q^2 druhej konvolučnej vrstvy a teda budeme postupovať nasledovne :

$$\Delta b_p^1 = -\gamma \frac{\partial L}{\partial b_p^1} \quad (12.105)$$

kde $p = 1 \dots 6$ a čo znamená následne

$$\frac{\partial L}{\partial b_p^1} = \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \frac{\partial L}{\partial C_p^1(i,j)} \cdot \frac{\partial C_p^1(i,j)}{\partial b_p^1} \quad (12.106)$$

a znova teda platí pre prvý člen pravej strany vzorec 12.68

$$\frac{\partial L}{\partial C_p^1(i,j)} = \Delta C_p^1(i,j) \quad (12.107)$$

následne prepíšme druhú časť vzorca 12.83 pomocou vzorca 12.11 pre $C_p^1(i,j)$ do nasledovného tvaru :

$$\frac{\partial C_p^1(i, j)}{\partial b_p^1} = \frac{\partial}{\partial b_p^1} \sigma \left(\sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} I(i-u, j-v) \cdot k_{1,p}^1(u, v) + b_p^1 \right) \quad (12.108)$$

kde výsledkom tejto parciálnej derivácie vzhladom na fakt že ide tu o deriváciu sigmoidálnej funkcie postupujeme obdobne ako odvádzaní adaptačného pravidla pre $\mathbf{k}_{p,q}^2$ podľa 12.72

$$\begin{aligned} \frac{\partial C_p^1(i, j)}{\partial b_p^1} &= C_p^1(i, j)(1 - C_p^1(i, j)) \cdot \\ &\quad \frac{\partial}{\partial b_p^1} \left\{ \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} I(i-u, j-v) \cdot k_{1,p}^1(u, v) + b_p^1 \right\} \end{aligned} \quad (12.109)$$

a následne môžeme napísť

$$\frac{\partial}{\partial b_p^1} \left\{ \sum_{u=-2}^{u=2} \sum_{v=-2}^{v=2} I(i-u, j-v) \cdot k_{1,p}^1(u, v) + b_p^1 \right\} = 1 \quad (12.110)$$

Teda na základe tohto môžeme napísť s podporou vzorcov 12.108, 12.84, 12.86 a konečne vzorca 12.87 nasledovný vzťah :

$$\Delta b_p^1 = -\gamma \frac{\partial L}{\partial b_p^1} = -\gamma \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \Delta C_p^1(i, j) \cdot C_p^1(i, j) (1 - C_p^1(i, j)) \cdot 1 \quad (12.111)$$

lenže tento posledný vzťah môžeme obdobne ako vo vzorci 12.76 zapísť nasledovne

$$\Delta b_p^1 = -\gamma \frac{\partial L}{\partial b_p^1} = -\gamma \sum_{i=1}^{i=24} \sum_{j=1}^{j=24} \Delta C_{p,\beta}^1(i, j) \quad (12.112)$$

kde $C_{p,\beta}^1$ je argumentom sigmodálnej funkcie σ teda samotná konvolúcia. Následne samotný iteračný a adaptačný vzťah bude pre b_p^2 vyzerať následovne

$$\boxed{\mathbf{b}_p^1 \{t+1\} = \mathbf{b}_p^1 \{t\} - \Delta \mathbf{b}_p^1 \{t\}} \quad (12.113)$$

Kapitola 13

Otázky k témam

13.1 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.A 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 napodobiteľný ?
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?

• **1.B. tutoriál**

1. Prečo je nutné hovoriť o stabilite NN a kedy ? Aká je kriterálna 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 lineárhou a nelineárhou 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 metódou najmenšej strednej kvadratickej chyby?
11. Aký je rozdiel medzi Adaline a perceptrónom ?

13.2 Repetitórium č. 2

- **2. tutoriál**

1. Aká je logika a cieľ Delta pravidla ?
2. Aky 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 urýchľovať učenie NN? čo sú heuristické pravidlá ?
6. Aky 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 hľadať 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?

13.3 Repetitórium č. 3

- **3. 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 **vítaz 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. Odvodte Ojove pravidlo!
11. Aký je rozdiel medzi nekontrolovaným BP učením a učením Counterpropagation?

13.4 Repetitórium č. 4

Tematické okruhy pre štúdium

- **4. tutoriál**

1. Vysvetlite základnu topológiu Hopfieldovej siete.
2. Aké sú aplikačné možnosti Hopfieldovej siete?
3. Stručne charakterizujte TSP problém.
4. Vysvetlite význam RC-modelu neurónu.
5. Vysvetlite základny princíp BP na RC sieti, význam $x(\infty)$ a $y(\infty)$.
6. čo predstavuje indexová funkcia δ_{ij} a čo δ_i .
7. Charakterizujte postup činnosti BP na RC NN.
8. Stručne popíšte topológiu ART, je to kontrolované učenie na RC ?
9. Aké sú základné kroky jej činnosti?
10. Aké existujú iné varianty ART-učenia?
11. Popíšte základné charakteristiky učenia podľa stavu systému **re-inforcement learning**.
12. Aké základné časti má táto NN? Aké majú funkcie?
13. Aké sú adaptačné pravidlá pre túto NN?
14. Charakterizujte modulárne NN, akú stratégiu používajú?
15. Aké sú základné typy časti modulárnych NN?
16. Charakterizujte jednoduchšiu a zložitejšiu MNN.

13.5 Repetitórium č. 5

Tematické okruhy pre štúdium v oblasti hlbokého učenia

- **5. tutoriál**

1. Vysvetlite základný princíp schémy rozpoznávania
2. Vysvetlite základný princíp schémy hlbokého učenia a porovnajte rozdiel medzi hlbokými a plynkými neurónovými sieťami
3. Vysvetlite pojmy hyperparametre a parametre pri HU
4. Vysvetlite čo je to konvolúcia dvoch matematických funkcií a aký je rozdiel medzi konvolúciou a kros-koreláciou
5. čo je to chybový priestor a vysvetlite ho na CNNSIMPLE neurónovej sieti. Aký je rozdiel medzi Loss funkciami a optimizérmi pri učení neurónových sietí.
6. Ako pristupujeme k budovaniu topológie CNN ?
7. Komentujte do prednú časť jednoduchej CNN popíšte výstupy z jednotlivých vrstiev takejto siete v zmysle jednotlivých rovníc.
8. komentujte prvú časť adaptácie pri CNNSIMPLE teda $\Delta\mathbf{W}$, $\Delta\mathbf{b}$
9. komentujte adaptácie $\Delta\mathbf{b}_q^2$, $\Delta\mathbf{k}_{p,q}^2$ podľa skript pri CNNSIMPLE
10. komentujte adaptácie $\Delta\mathbf{b}_p^1$, $\Delta\mathbf{k}_{1,p}^1$ pri CNNSIMPLE
11. Čo je úlohou Normalizačná vrstvy pri CNN aké typy normalizácií poznáme
12. Akú úlohy zohráva vrstva Dropout pri učení neurónových sietí
13. Akú úlohu zohráva Reziduálny block pri DL ?
14. Akú úlohu zohrávajú incepčné bloky pri DL ?
15. Aké grafické karty poznáte od jakých výrobcov ?
16. Čo sú to AI počítače a čo sú Edge-počítače pre AI a aké majú výkony ?