INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
TENTO PROJEKT JE SPOLUFINANCOVÁN EVROPSKÝM SOCIÁLNÍM FONDEM A STÁTNÍM ROZPOČTEM ČESKÉ REPUBLIKY.

NEURONOVÉ SÍTĚ

EVA VOLNÁ

PODPORA TERCIÁRNÍHO VZDĚLÁVÁNÍ
STUDENTŮ SE SPECIFICKÝMI
VZDĚLÁVACÍMI POTŘEBAMI
NA OSTRAVSKÉ UNIVERZITĚ V OSTRAVĚ

CZ.1.07/2.2.00/29.0006

OSTRAVA, červen 2013

Studijní opora je jedním z výstupu projektu ESF OP VK.

Číslo Prioritní osy:	7.2
Oblast podpory:	7.2.2 – Vysokoškolské vzdělávání
Příjemce:	Ostravská univerzita v Ostravě
Název projektu:	Podpora terciárního vzdělávání studentů se specifickými vzdělávacími potřebami na Ostravské univerzitě v Ostravě
Registrační číslo projektu:	CZ.1.07/2.2.00/29.0006
Délka realizace:	6.2.2012 – 31.1.2015
Řešitel:	PhDr. Mgr. Martin Kaleja, Ph.D.

Tento projekt je spolufinancován Evropským sociálním fondem a státním rozpočtem České republiky.

Název: Neuronové sítě
Autor: doc. RNDr. PaedDr. Eva Volná, PhD.

Studijní opora k inovovanému předmětu: Neuronové sítě (KIP/NESI1)

Jazyková korektura nebyla provedena, za jazykovou stránku odpovídá autor.

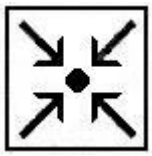
Recenzent: doc. RNDr. PaedDr. Hashim Habiballa, PhD, Ph.D.
Ostravská univerzita v Ostravě

ISBN 978-80-7464-329-3

OBSAH:

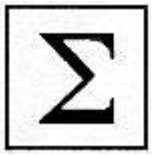
Vysvětlivky k používaným symbolům

Průvodce studiem – vstup autora do textu, specifický způsob kterým se studentem komunikuje, povzbuzuje jej, doplňuje text o další informace.



Příklad – objasnění nebo konkretizování problematiky na příkladu ze života, z praxe, ze společenské reality apod.

K zapamatování



Shrnutí – shrnutí předcházející látky, shrnutí kapitoly.

Literatura – použitá ve studijním materiálu, pro doplnění a rozšíření poznatků.

Kontrolní otázky a úkoly – prověřují, do jaké míry studující text a problematiku pochopil, zapamatoval si podstatné a důležité informace a zda je dokáže aplikovat při řešení problémů.

Úkoly k textu – je potřeba je splnit neprodleně, neboť pomáhají k dobrému zvládnutí následující látky.

Korespondenční úkoly – při jejich plnění postupuje studující podle pokynů s notnou dávkou vlastní iniciativy. Úkoly se průběžně evidují a hodnotí v průběhu celého kurzu.

Otázky k zamyšlení

Část pro zájemce – přináší látku a úkoly rozšiřující úroveň základního kurzu. Pasáže i úkoly jsou dobrovolné.

Úvod

Tento text má sloužit pro potřeby výuky výběrového předmětu neuronové sítě na katedře informatiky a počítačů. Nepředpokládá se žádná předchozí znalost problematiky, pouze základy matematické analýzy, především diferenciální počet a maticový počet. Předmět má rysy kurzu, ve kterém student získá ucelenější pohled na problematiku umělých neuronových sítí.

Po prostudování textu budete znát:

Tyto učební texty jsou určeny studentům informatiky pro předmět neuronové sítě. Jsou v nich vysvětleny základní pojmy z teorie umělých neuronových sítí. V jednotlivých kapitolách jsou postupně podle obtížnosti uvedeny základní modely neuronových sítí (tj. perceptron, adaline, madaline, dopředná vícevrstvá neuronová síť s adaptační metodou backpropagation, neuronové sítě pracující na principu samoorganizace a diskrétní Hopfieldova síť.), a to jejich architektura, aktivní dynamika a adaptivní dynamika. Závěrečné kapitoly jsou věnovány problémům klasifikace, postavení neuronových sítí v informatice a jejich použití v teoretické informatice a umělé inteligenci.

Po prostudování textu získáte:

Po prostudování této studijní opory získáte ucelený pohled na základní principy teorie umělých neuronových sítí. Důraz je zde kladen nejen na teoretické základy, ale i na schopnost je aplikovat při řešení příkladů.

Korespondenční úkoly.

Vyberte si a vypracujte JEDEN korespondenční úkol z kapitoly 5. - 11. této studijní opory. Dále vypracujte korespondenční úkol z kapitoly 12. Řešení obou korespondenčních úkolů zašlete do konce semestru (nejpozději před zkouškou) na adresu eva.volna@osu.cz .

Pokud máte jakékoliv věcné nebo formální připomínky k textu, kontaktujte autora (eva.volna@osu.cz).

1 Základní pojmy neuronových sítí

V této kapitole se dozvíte:

- Jaké jsou základní pojmy z oblasti umělých neuronových sítí.
- Rozdíl mezi klasickým počítačem a neuronovou sítí.

Po jejím prostudování byste měli být schopni:

- vysvětlit základní pojmy z teorie umělých neuronových sítí,
- objasnit jaký je rozdíl mezi klasickým počítačem a neuronovou sítí,
- rozdělit neuronové sítě podle různých kritérií.

Klíčová slova kapitoly: Neuronová síť, algoritmus učení, přenosová funkce neuronu.

Průvodce studiem

V této úvodní kapitole se stručně seznámíte se základními pojmy teorie neuronových sítí. Ukážeme si, jaký je rozdíl mezi klasickým počítačem a neuronovou sít a podle jakých kritérií lze dělit neuronové sítě. V závěru si vysvětlíme, proč se síť učí a jak síť funguje.

Na zvládnutí této kapitoly budete potřebovat asi 3 hodiny.

Neuronová síť je jeden z výpočetních modelů používaných v umělé inteligenci. Jejím vzorem je chování odpovídajících biologických struktur. Umělá neuronová síť je struktura určená pro distribuované paralelní zpracování dat. Skládá se z umělých (nebo také formálních) neuronů, jejichž vzorem je biologický neuron. Neurony jsou vzájemně propojeny a navzájem si předávají signály a transformují je pomocí tzv. přenosových funkcí. Neuron má libovolný počet vstupů, ale pouze jeden výstup.

1.1 Rozdíl mezi klasickým počítačem a neuronovou sítí

Rozdíl mezi klasickým počítačem a neuronovou sítí je ten, že při používání klasického PC musíme vytvořit program, který řeší daný problém. Do tohoto programu obvykle zahrneme ve formě podmínek a rozhodovacích instrukcí veškeré dostupné informace. Co se však stane, jestliže je náš program postaven před problém, který patří do třídy známých problémů, ale je dost odlišný? Obvykle je buď ignorován, nebo v lepším případě je obsluha jen upozorněna, že se vyskytl neznámý případ, který byl odložen bokem. V takovém případě musí opět nastoupit programátor a program upravit.
Co se však stane, použije-li se neuronová síť? Nemusí se vymýšlet žádný algoritmus a v případě vhodné konfigurace a dobrého učení daná neuronová síť zareaguje správným způsobem a novou informaci zařadí s velkou pravděpodobností do správné třídy. Není potřeba žádné úpravy sítě. Samozřejmě, že nic není neměnné, takže jak vzrůstá počet nových informací, je pravděpodobné, že nám vzniknou i nové třídy informací, na které se daná síť musí doučit a případně pozměnit svou konfiguraci konfiguraci. Ale to se dá obejít bez přítomnosti člověka.

Jako názorný příklad lze uvést dva programátory, kteří se neznají a oba dva dostanou stejný úkol. Musí udělat algoritmus, který má umět rozlišit podle vstupních informací, do jaké třídy patří daný dopravní prostředek. Kolo do třídy kol, auto do třídy aut, letadlo do třídy letadel, atd. Klasický programátor to bude asi řešit tak, že vytvoří kriteriální filtry typu „jestliže objekt má dvě kola, řetězový převod na zadní kolo, řídítka, sedátko pro jednu osobu, atd. pak jej zařaď do třídy kol" a totéž udělá pro všechny možné třídy. Samozřejmě, že chytřejší programátor bude popis dělat v proporcích a ne v absolutních rozměrech z jednoduchého důvodu - i malé dětské kolo je kolo. Co se však stane, jestliže má jeho program vyhodnotit kolo z minulého století (obří přední kolo s maličkým vzadu a navíc bez převodu - pedály byly přímo na předním kole), nebo zařadit trojkolku či nový futuristický model kola? Pravděpodobně ho vyřadí a v lepším případě dá vědět obsluze, že došlo k vyřazení neznámého případu - programátor musí udělat úpravu. To stojí čas a peníze. To však

neplatí v případě neuronové sítě. Ta by s velkou pravděpodobností provedla správné vyhodnocení - zařadila by nový objekt (kolo) do správné třídy. Několik důležitých rozdílů mezi klasickým PC a neuronovou sítí je uvedeno v následující tabulce 1 (Zelinka, 1999).

Neuronová síť	**Počítač**
Je učena nastavováním vah, prahů a struktury	Je programován instrukcemi, (if, then, go to,...)
Paměťové a výkonné prvky jsou uspořádány spolu	Proces a paměť pro něj jsou separovány
Paralelismus	Sekvenčnost
Tolerují odchylky od originálních informací	Netolerují odchylky
Samoorganizace během učení	Neměnnost programu

Tabulka 1: Rozdíly mezi neuronovou sítí a PC

1.2 Typy umělých neuronových sítí

Všechny typy neuronových sítí jsou složeny ze stejných stavebních jednotek - neuronů. Mohou obsahovat různé přenosové funkce, spojení mezi sebou a adaptivní - učící algoritmus. To vše pak určuje, o jakou síť se jedná.

Umělé neuronové sítě se dělí podle několika kritérií (Zelinka, 1999).

Podle počtu vrstev:

- s jednou vrstvou (Hopfieldova síť, Kohonenova síť, ...)
- s více vrstvami. (ART síť, Perceptron, klasická vícevrstvá síť s algoritmem Backpropagation)

Dělení podle počtu vrstev znamená, že rozlišujeme z kolika vrstev se daná síť skládá. Existují sítě s jednou vrstvou, se dvěma, třemi a více vrstvami. Sítě s jednou či dvěma vrstvami bývají většinou speciální sítě jako např. Hopfieldova, Kohonenova či ART síť, které mají svůj speciální učící algoritmus a topologii, zatímco pro sítě se třemi a více vrstvami se obvykle používá topologie klasické vícevrstvé sítě a adaptační algoritmus Backpropagation. Pro topologii sítě obvykle platí pravidlo, že každý neuron

bývá spojen s každým neuronem ve vyšší vrstvě, obrázek 29. Zvláštností je např. Hopfieldova síť, ve které je spojen každý neuron se všemi ostatními. Každé spojení mezi neurony je ohodnoceno váhami, které mohou nabývat různých hodnot a vyjadřují, jaký význam tento spoj má pro daný neuron. To ovšem neznamená, že spoje s malou vahou lze zanedbat, protože nikdy není jasné, jaký vliv má tento vstup na celkovou činnost sítě.

Podle typu algoritmu učení:

- s učitelem (síť s algoritmem Backpropagation,...)
- bez učitele (Hopfieldova síť,...)

Učení s učitelem znamená, že se síť snaží přizpůsobit svou odezvu na vstupní informace tak, aby se její momentální výstup co nejvíce podobal požadovanému originálu. Je to stejné, jako když učíme dítě číst. Ukážeme mu písmeno (vstup) a vyslovíme ho např. „áááá" (požadovaný výstup). Pokud je dítě pozorné, pak se nás snaží napodobit - učí se. Učitelem zde není myšlen onen dotyčný dospělý, ale princip předložení fonetického vzoru a vyžadování jeho zopakování. Dotyčný dospělý pouze zajišťuje vykonávání těchto operací. Učení bez učitele je proces, ve kterém síť vychází z informací, které jsou obsaženy ve vstupních vektorech. Je to stejné, jako když vám někdo vysype na jednu hromadu různé tvary jako krychličky, pyramidky, kuličky, tyčinky a jejich zdeformované variace a dá vám za úkol tuto hromadu roztřídit na předem neurčený počet hromádek podle tvaru – zde nebyly stanoveny „podmínky" učitele (tj. kolik hromádek jakých typů máte udělat) a tudíž se jedná u učení bez učitele. Průměrně inteligentní člověk začne třídit hromadu na hromádky podle typu (podobnosti) tvaru, tzn. hromádku krychliček, kuliček, atd. Pokud se vyskytne zdeformovaný tvar (např. krychlička se zdeformovanými stranami či useknutým rohem) tak jej pravděpodobně zařadí mezi krychličky. Pokud bude takhle zdeformovaných krychliček vice, určitě vytvoří třídu „zdeformované krychličky". V případě že budou deformace příliš velké a tudíž nebude možné rozpoznat, co je to za tvar, bude určitě vytvořena třída „nerozpoznané". Pokud se čtenáři nezdá příklad s člověkem vhodný, opak je pravdou. Mozek je velmi složitý neurosystém, který pomocí zraku, sluchu, hmatu a dalších vstupů zpracovává informace takto získané - v podstatě se takto adaptuje na své okolí.

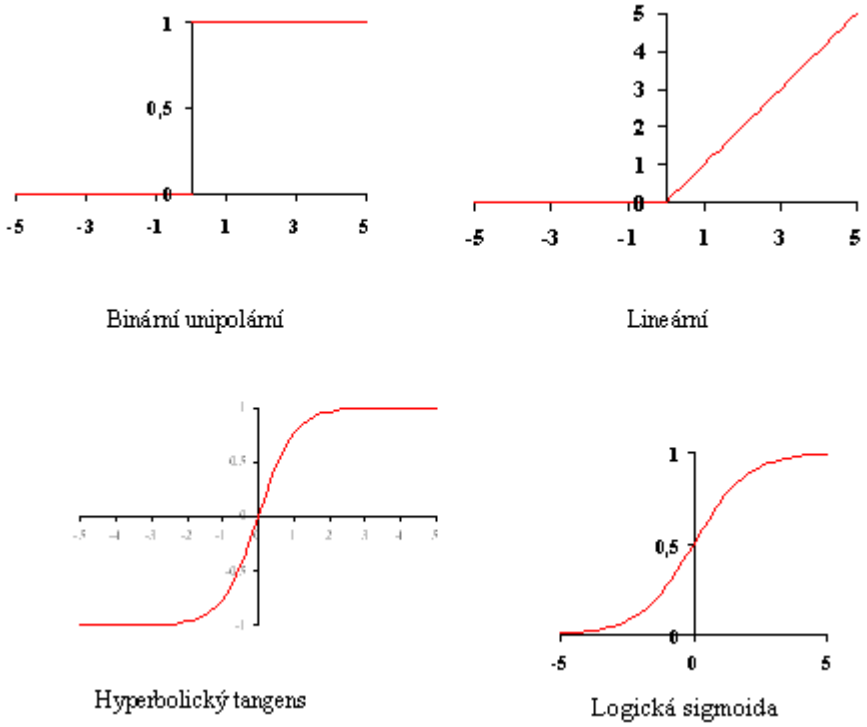
Podle stylu učení na sítě s učením:

- deterministickým (např. algoritmus Backpropagation)
- stochastickým (náhodné nastavování vah)

Styl učení v podstatě znamená, jak se přistupuje k nastavení vah sítě. V případě, že se jedná o nastavení vah výpočtem, pak mluvíme o deterministickém učení. Jestliže jsou však váhy získávány pomocí generátoru náhodných čísel, pak mluvíme o stochastickém stylu učení. Tento způsob získání vah sítě se obvykle používá jen při startu sítě (inicializaci sítě).

Přenosová funkce neuronu:

Přenosová funkce neuronu je funkce, která transformuje vstupní signál na signál výstupní. Tato funkce může být skoková, či spojitá, ale musí být monotónní - tzn. že přiřazení odezev výstupu na vstup je jednoznačné.



Obrázek 1: Přenosové funkce a neuronů

Pro správný chod neuronu a neuronových sítí je důležité, jakou přenosovou funkci zvolíme. Přenosová funkce udává, jaká bude odezva výstupních neuronů na vstupní podnět. Jsou různé druhy funkcí, u kterých obecně platí, že

jejich výstupní hodnota musí být z oboru hodnot dané funkce (sigmoida, hyperbolický tangens, binární funkce apod.). Nejpoužívanější přenosové funkce jsou uvedeny na obrázku 1 (Zelinka, 1999).
Volba funkce závisí na problému, který chceme řešit. Např. pokud chceme klasifikovat, zda je výrobek dobrý či špatný, pak nám stačí binární funkce. Pokud bychom použili funkci spojitou, pak musíme rozhodnout, jaká její hodnota (0.7, 0.8,...) znamená dobrý a jaká špatný výrobek.

1.3 Obecné schéma činnosti neuronové sítě

Pod pojmem topologie (někdy také struktura) sítě, rozumíme způsob, jakým jsou mezi sebou spojeny jednotlivé neurony, vrstvy a kolik vstupů a výstupů síť má. Do topologie sítě lze zahrnout i typ přenosové funkce neuronů. Mezi parametry neuronové sítě dále patří parametr učení, momentum, atd.

V technickém slova smyslu si lze představit neuronovou síť jako síť složenou z jedné či několika vrstev, přičemž každá vrstva se skládá z „libovolného“ počtu neuronů. Libovolného samozřejmě jen teoreticky, protože jsme limitováni technickými podmínkami. Vlastní funkce neuronové sítě je v podstatě transformační funkce, která přiřazuje jistému vstupnímu obrazu obraz výstupní. Důkaz matematického charakteru o takovéto funkci nebyl dlouho k dispozici. Kolmogororův teorém o řešení třináctého Hilbertova problému (Zelinka, 1999) aplikovaného na neuronové sítě vedl k poznatku, že k aproximaci libovolné funkce neuronovou sítí stačí, aby měla minimálně tři vrstvy s odpovídajícím počtem neuronů v každé z nich. Bohužel tento důkaz již nespecifikuje počet neuronů v těchto vrstvách, jež by byl z hlediska řešení daného problému optimální.

Proč se síť učí:
Učící schopnost neuronových sítí spočívá právě v možnosti měnit všechny váhy v síti podle vhodných algoritmů na rozdíl od sítí biologických, kde je schopnost se učit založena na možnosti tvorby nových spojů mezi neurony. Fyzicky jsou tedy obě schopnosti se učit založeny na rozdílných principech, nicméně z hlediska logiky ne. V případě vzniku nového spoje - vstupu u biologického neuronu je to stejné, jako když v technické síti je spoj mezi

dvěma neurony ohodnocen vahou s hodnotou 0, a tudíž jako vstup pro neuron, do kterého vstupuje, neexistuje. V okamžiku, kdy se váha změní z 0 na libovolné číslo, tak se daný spoj zviditelní - vznikne.

Jak síť funguje:

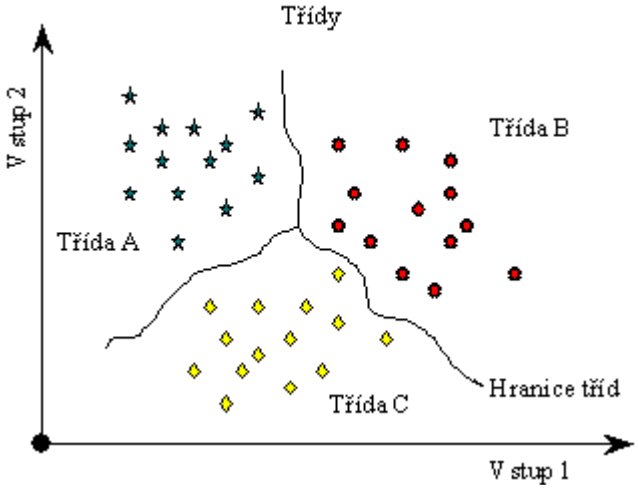
Jeden ze základních předpokladů pro funkci sítě je její naučení - adaptace na daný problém, během které se nastavují váhy sítě. Adaptační - učící proces se skládá ze dvou fází - adaptační a aktivační.
Nově vytvořená, ale také jakákoliv nenaučená neuronová síť neumí řešit žádný problém. Aby se mohla síť používat, musí být naučena stejně, jako kterýkoliv živý jedinec (samozřejmě, že zde se množství informací a délka učení nedá porovnávat). Z tohoto důvodu byly vyvinuty algoritmy, pomocí kterých se příslušná síť dokáže naučit na danou množinu informací. Algoritmus se obvykle dělí na dvě fáze, a to na fázi aktivační (vybavovací) a adaptační (učící), které ke své činnosti potřebují trénovací množinu. Trénovací množina je skupina vektorů obsahujících informace o daném problému pro učení. Pokud učíme síť s učitelem, pak jsou to dvojice vektorů vstup - výstup. Jestliže učíme síť bez učitele, pak trénovací množina obsahuje jen vstupní vektory. Pokud používáme jen fázi aktivační, pak mluvíme o vybavování. Tuto fázi používáme samostatně jen tehdy, když je síť naučena. Cyklické střídání obou fází je vlastní učení.

Aktivační fáze je proces, při kterém se předložený vektor informací na vstup sítě přepočítá přes všechny spoje včetně jejich ohodnocení vahami až na výstup, kde se objeví odezva sítě na tento vektor ve formě výstupního vektoru. Při učení se tento vektor porovná s vektorem originálním (požadovaným, výstupním) a rozdíl (lokální odchylka - chyba) se uloží do paměťové proměnné.

Adaptační fáze je proces, při kterém je minimalizována lokální chyba sítě tak, že se přepočítávají váhy jednotlivých spojů směrem z výstupu na vstup za účelem co největší podobnosti výstupní odezvy s originálním vektorem. Poté se opět opakuje aktivační fáze. Další získaný rozdíl (lokální odchylka) se přičte k předchozímu, atd. Pokud se tímto postupem projde celá trénovací množina, je

dokončena jedna epocha. Celé sumě odchylek za jednu epochu se říká globální odchylka - chyba. Pokud je globální odchylka menší než námi požadovaná chyba, pak proces učení skončí.

Z výše popsaného je vidět, že proces učení není nic jiného, než přelévání informací ze vstupu na výstup a naopak. Při učení se v tzv. vstupním prostoru vytvářejí shluky bodů, které představují jednotlivé členy tříd, přičemž každý shluk představuje jednu třídu (obrázek 2) (Zelinka, 1999).



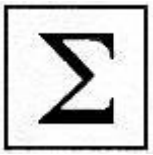
Obrázek 2: Třídy a jejich hranice

Například, mějme několik tříd: třída kol, třída aut, třída lodí, atd. Z každé z nich se vybere množina reprezentativních zástupců (vzorů pro učení) a všechny se popíší vhodným číselným způsobem ve formě vektorů. Pro každou množinu vektorů jedné třídy se vytvoří vzorový vektor, který bude zastupovat mateřskou třídu z reálného světa. V tomto okamžiku jsou vytvořeny skupiny vektorů popisující jednotlivé členy a jim příslušející představitele tříd ve formě vektorů. Například máme vektory, které popisují vybrané členy z třídy kol a vektor, který říká „já jsem třída kol“. V tomto případě učení znamená to, že se učící algoritmus snaží najít takovou kombinaci vah, která umožní přiřazení vektoru třídy jejím členům. Jinak řečeno, hledá se taková kombinace vah, že pokud položíme ve vybavovací fázi na vstup vektor popisující objekt např. kolo, pak by se na výstupu měl objevit vektor, který říká „já jsem třída kol“. Mějme tedy například 5 různých tříd a pro každou třídu 20 reprezentativních členů, dohromady tedy 100 vstupních vektorů. Při učení se snažíme najít

takové váhy, aby odezva sítě pro daný vstupní vektor - člen byla co nejvíce podobná vektoru jeho třídy. Vzhledem k tomu, že se neuronová síť učí vždy s nějakou chybou, nebude odezva odpovídat vždy přesně originálu a tím pádem dostaneme shluk bodů, okolo pozice originální třídy (obrázek 2).

To, zda se naše síť naučí správným odezvám na dané podněty, závisí na více okolnostech, a to na množství vektorů a jejich velikosti, topologii sítě, odlišnosti charakteristických vlastností jednotlivých tříd, přípravě trénovací množiny, a jiných. Schopnost sítě přiřazovat jednotlivé vstupní členy daným třídám je dána tím, že síť v podstatě počítá vzdálenost daného členu od členů již přiřazených a na základě toho usuzuje, do jaké třídy daný vektor patří. To samozřejmě může někdy znamenat problém, který se dá odstranit tzv. pravděpodobnostní sítí (Barnsley, 1993).

Shrnutí kapitoly



Kapitola vás stručně seznámila se základními pojmy teorie neuronových sítí. Ukázali jsme si, jaký je rozdíl mezi klasickým počítačem a neuronovou sít a podle jakých kritérií lze dělit neuronové sítě. V závěru jsme si vysvětlili, proč se síť učí a jak síť funguje.

Kontrolní otázky a úkoly:

1. Jaký je rozdíl mezi klasickým počítačem a neuronovou sítí?
2. Podle jakých kritérií lze dělit neuronové sítě?
3. Jaké je obecné schéma činnosti neuronové sítě?

Citovaná a doporučená literatura

Zelinka, I. (1999) Aplikovaná informatika. Učební texty VUT v Brně.
Barnsley M.F. (1993) Fractals Everywhere. Academic Press Professional 1993.

2 Logické neurony

V této kapitole se dozvíte:

- Co je to logický neuron.
- Jaké Boolovské funkce mohou simulovat sítě složené z logických neuronů.

Po jejím prostudování byste měli být schopni:

- vysvětlit, jaké Boolovské funkce mohou simulovat logické neurony,
- objasnit princip činnosti logických neuronů,

Klíčová slova kapitoly: Boolovské funkce, logický neuron.

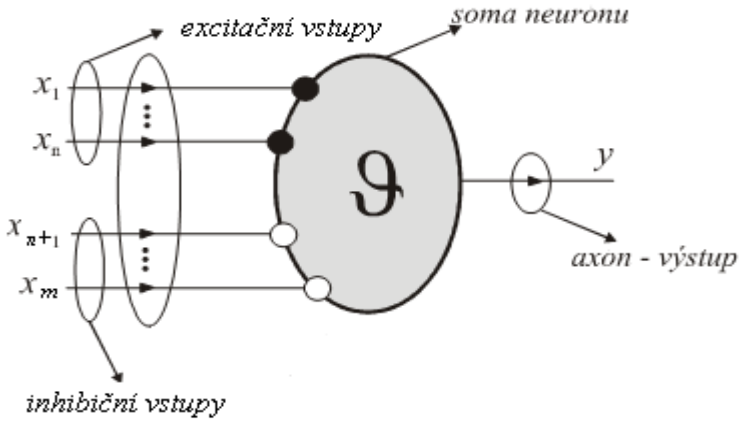
Průvodce studiem

V této kapitole si ukážeme, logické neurony jsou schopny simulovat logické spojky, které jsou charakterizovány jako lineární oddělitelné (např. disjunkce, konjunkci, implikaci a negaci).

Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny.

V publikaci Warrena McCullocha a Waltera Pittsa (McCulloch, 1943) „A logical calculus of the ideas immanent to nervous activity" z r. 1943 bylo poprvé uvedeno, že neuronové sítě jsou mocným modelovým prostředkem, např. sítě složené z logických neuronů mohou simulovat Boolovské funkce.

Elementární jednotkou McCullochových a Pittsových neuronových sítí je logický neuron (výpočetní jednotka), přičemž stav neuronu je binární (tj. má dva stavy, 1 nebo 0). Takový logický neuron lze interpretovat jako jednoduché elektrické zařízení - relé. Předpokládejme, že dendritický systém logického neuronu obsahuje jak excitační vstupy (popsáno binárními proměnnými x_1, x_2, ..., x_n, které zesilují odezvu), tak inhibiční vstupy (popsáno binárními proměnnými x_{n+1}, x_{n+2}, ..., x_m, které zeslabují odezvu), viz obrázek 3 (Kvasnička, 1997).



Obrázek 3: Znázornění McCullochova a Pittsova neuronu, který obsahuje dendritický systém pro vstupní (excitační nebo inhibiční) aktivity, axon pro výstup neuronové aktivity. Soma (tělo neuronu) je charakterizována prahovým koeficientem ϑ.

Aktivita logického neuronu je jednotková, pokud vnitřní potenciál neuronu definovaný jako rozdíl mezi sumou excitačních vstupních aktivit a inhibičních vstupních aktivit je větší nebo roven prahu - ϑ, v opačném případě je nulová

$$y = \begin{cases} 1 & \left(x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m \geq -\vartheta\right) \\ 0 & \left(x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m < -\vartheta\right) \end{cases}$$

Pomocí skokové funkce s můžeme aktivitu vyjádřit takto:

$$y = s\left(\underbrace{x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m}_{\xi} + \vartheta\right)$$

Tento vztah pro aktivitu logického neuronu můžeme rovněž interpretovat tak, že excitační aktivity vstupují do neuronu přes spoje, které jsou ohodnoceny jednotkovým váhovým koeficientem (w=1), zatímco inhibiční aktivity vstupují do neuronu přes spoje se záporným jednotkovým váhovým koeficientem (w=-1). Potom aktivitu logického neuronu můžeme vyjádřit takto:

$$y = s\left(\underbrace{w_1 x_1 + \ldots + w_m x_m}_{\xi} + \vartheta\right) = s\left(\sum_{i=1}^{m} w_i x_i + \vartheta\right)$$

Jednoduchá implementace elementárních Booleovských funkcí disjunkce, konjunkce, implikace a negace je znázorněna na obrázku 4.

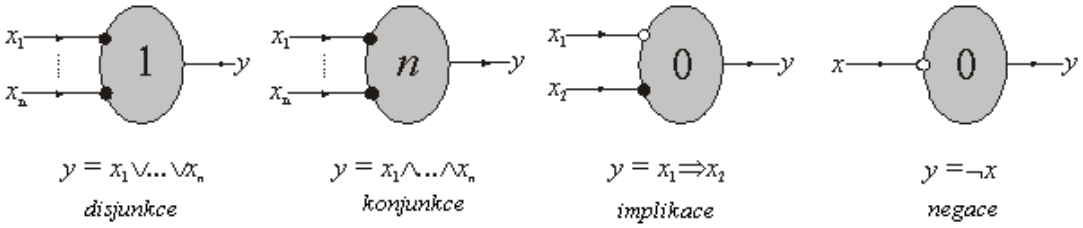
Jako ilustrační příklad uvedeme funkci disjunkce pro n = 2, použitím vzorců z definice logického neuronu dostaneme (Kvasnička, 1997):

$$y_{OR}(x_1, x_2) = s(x_1 + x_2 - 1)$$

#	x_1	x_2	$y_{OR}(x_1,x_2)$	$x_1 \vee x_2$
1	0	0	s(-1)	0
2	0	1	s(0)	1
3	1	0	s(0)	1
4	1	1	s(1)	1

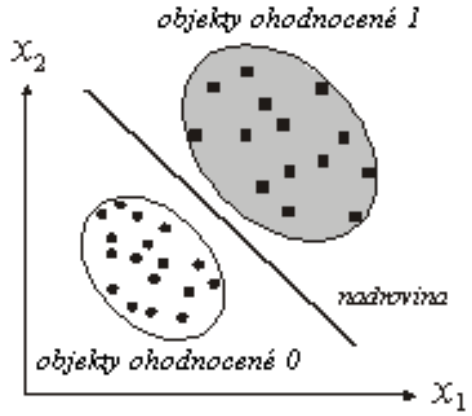
Tabulka 2: Binární Booleova funkce disjunkcie

Z tabulky 2 vyplývá, že Booleova funkce y_{OR} simuluje Booleovu funkci disjunkcie.



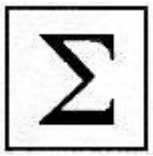
Obrázek 4: Logické neurony pro implementaci Booleovských funkcí disjunkce, konjunkce, implikace a negace. Excitační spoje jsou znázorněny plným kroužkem, inhibiční prázdným kroužkem.

Můžeme si položit otázku, jaké Boolovské funkce je schopen logický neuron vyjádřit? Tato otázka se dá poměrně jednoduše vyřešit pomocí geometrické interpretace výpočtu probíhajícího v logickém neuronu. Výpočetní funkce logického neuronu rozděluje prostor vstupů na dva poloprostory pomocí roviny $w_1x_1 + w_2x_2 + ... + w_nx_n = \vartheta$ pro koeficienty $w_i \in \{0, +1, -1\}$. Říkáme, že Booleova funkce $f(x_1, x_2, ..., x_n)$ je lineárně separovatelná, pokud existuje taková rovina $w_1x_1 + w_2x_2 + ... + w_nx_n = \vartheta$, která separuje prostor vstupních aktivit tak, že v jedné části prostoru jsou vrcholy ohodnoceny 0, zatímco v druhé části prostoru jsou vrcholy ohodnoceny 1 (viz obrázek 5).



Obrázek 5: Schematické znázornění pojmu lineární separovatelnost, kde kulaté a čtvercové objekty jsou separované nadrovinou tak, že v jednom poloprostoru jsou objekty jednoho druhu, zatímco ve druhém poloprostoru jsou objekty druhého druhu.

Logický neuron je schopen simulovat jen takové Boolovské funkce, které jsou lineární separovatelné.



Shrnutí kapitoly

V této kapitole jsme si ukázali, že logické neurony jsou schopny simulovat logické spojky, které jsou charakterizovány jako lineární oddělitelné (např. disjunkce, konjunkci, implikaci a negaci).

Kontrolní otázky a úkoly:

1. Jaké Boolovské funkce mohou simulovat logické neurony?
2. Jaká je topologie logického neuronu?
3. Jaké Boolovské funkce je schopen simulovat logický neuron?

Korespondenční úkoly

Uveďte logické neurony pro implementaci Booleovských funkcí disjunkce, konjunkce, implikace a negace. Excitační spoje znázorněte plným kroužkem, inhibiční prázdným kroužkem. Řešte numericky i graficky. Uveďte definice logického neuronu pro implementaci uvedených Booleovských funkcí.

Citovaná a doporučená literatura

McCulloch, W. S., Pitts, W. H. (1943) A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, **5** (1943), 115-133.

Kvasnička, V., Beňušková, L., Pospíchal, J., Farkaš, I., Tiňo, P., Kráľ, A. (1997) Úvod do teórie neurónových sietí. IRIS, Bratislava.

3 Matematický model neuronu

V této kapitole se dozvíte:

- Jaká je struktura biologického neuronu.
- Jak modelujeme biologický neuron.

Po jejím prostudování byste měli být schopni:

- vysvětlit model formálního neuronu s biasem,
- objasnit co je to vnitřní potenciál.

Klíčová slova kapitoly: Biologický neuron, synapse, formální neuron, vnitřní potenciál, bias.

Průvodce studiem

V této úvodní kapitole se stručně seznámíte se základním matematickým modelem biologického neuronu, tj. formálním neuronem. Z tohoto modelu budeme dále vycházet, a proto je nutné, abyste jeho pochopení věnovali zvýšenou pozornost.

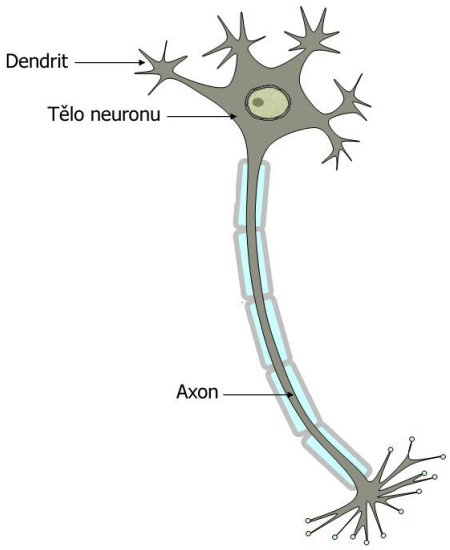
Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny.

3.1 Biologický neuron

Základní jednotkou mozkové tkáně je specializovaná buňka označovaná jako neuron. Neuronů je v mozku asi kolem 100 miliard (100 000 000 000). Zjednodušeně řečeno, každý neuron se skládá ze dvou částí – z buněčného těla a výběžku (axonu). Výběžek slouží k předávání signálu dalšímu neuronu. Na těle se nachází speciální výběžky, označované jako dendrity, které naopak slouží k příjmu signálů od jiných neuronů.Struktura neuronu je schematicky znázorněna na obrázku 6.

Neuron tedy přijímá signály prostřednictvím dendritů, zpracuje je a pomocí axonu je předává dál. Počet dendritů se liší, jeden neuron může mít desítky

dendritů, až stovky tisíc dendritů. Při výše uvedeném počtu neuronů tak vzniká nepředstavitelně složitá síť.



Obrázek 6: Model neuronu (převzato z http://www.upsychiatra.cz/srozumitelne/neurony-neurotransmitery-a-synapse)

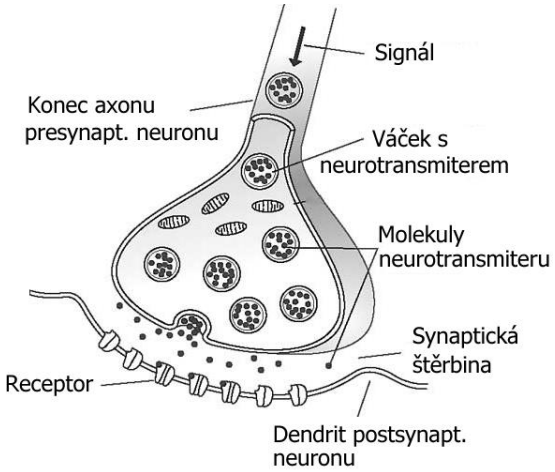
Samotné šíření signálu prostřednictvím neuronů se děje na základě změn eletrického napětí na jejich membránách spojených s přečerpáváním elektricky nabitých částic (Polách, 2012). Tyto procesy probíhají takřka bleskovou rychlostí. Přenos signálu mezi neurony (tedy přenos v oblasti axonu jednoho neuronu ku dendritu druhého neuronu) se děje jinou cestou, protože neurony nejsou ve skutečnosti vzájemně spojené, ale jsou mezi nimi velmi malé mezery – ty znemožňují vedení elektrického náboje. Signál se přenáší jiným způsobem, a to chemickými látkami, zvanými neurotransmitery.

Místa, kde dochází k přenosu signálů mezi neurony, se označují jako synapse. Synapse se skládá ze tří částí – z presynaptické membrány (“pre” = lat. před), synaptické šterbiny a postsynaptické membrány (“post” = lat. po).

Presynaptická membrána je koncem předcházejícího neuronu a postsynaptická membrána je začátkem následujícího neuronu.

Jak je vidět na obrázku 7, blízko presynaptické membrány se nachází uvnitř neuronu váčky. Tyto váčky obsahují neurotransmiter. Jakmile signál doputuje z těla neuronu až na konec axonu, způsobí zde splynutí váčků s membránou a “vylití” jejich obsahu do štěrbiny. Molekuly neurotransmiteru překonají velmi rychle prostor štěrbiny a naváží se na receptory na postsynaptické membráně. Receptory jsou útvary na membráně, které jsou po navázání neurotransmiterů

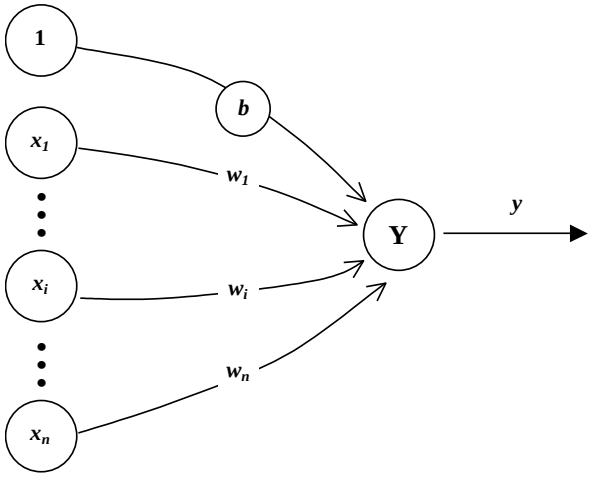
schopné šířit signál do dál neuronu, ke kterému patří. Důležité je, že neurotransmitery mají přesnou strukturu a díky ní se mohou navázat pouze na odpovídající receptory.



Obrázek 7: Princip synapse (převzato z http://www.upsychiatra.cz/srozumitelne/neurony-neurotransmitery-a-synapse)

3.2 Formální neuron

Základem matematického modelu neuronové sítě je formální neuron. Jeho struktura je schematicky zobrazena (Fausett, 1994) na obrázku 8. Formální neuron Y (dále jen neuron) má obecně n reálných vstupů, které modelují dendrity a určují vstupní vektor $x = (x_1, ... ,x_n)$. Tyto vstupy jsou ohodnoceny reálnými synaptickými váhami tvořícími vektor $w = (w_1, ... , w_n)$. Ve shodě s neurofyziologickou motivací mohou být synaptické váhy i záporné, čímž se vyjadřuje jejich inhibiční charakter.



Obrázek 8: Formální neuron s biasem

Vážená suma vstupních hodnot y_in představuje vnitřní potenciál neuronu Y:

$$y\_in = \sum_{i=1}^{n} w_i x_i$$

Bias může být do vztahu včleněn přidáním komponent $x_0 = 1$ k vektoru **x**, tj. $\mathbf{x} = (1, x_1, x_2, ..., x_n)$. Bias je dále zpracováván jako jakákoliv jiná váha, tj. $w_0 = b$. Vstup do neuronu Y je pak dán následujícím vztahem:

$$y\_in = \sum_{i=0}^{n} w_i x_i = w_0 + \sum_{i=1}^{n} w_i x_i = b + \sum_{i=1}^{n} w_i x_i$$

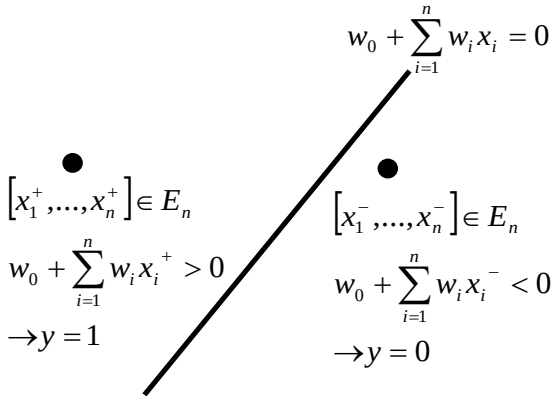
Hodnota vnitřního potenciálu y_in po dosažení hodnoty b indukuje výstup (stav) y neuronu Y, který modeluje elektrický impuls axonu. Nelineární nárůst výstupní hodnoty y = f(y_in) při dosažení hodnoty potenciálu b je dán aktivační (přenosovou) funkcí f. Nejjednodušším typem přenosové funkce je ostrá nelinearita, která má pro neuron Y tvar:

$$f(y\_in) = \begin{cases} 1 \text{ pokud } y\_in \geq 0 \\ 0 \text{ pokud } y\_in < 0 \end{cases}$$

Příklad:

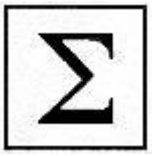
K lepšímu pochopení funkce jednoho neuronu nám pomůže geometrická představa načrtnutá na obrázku 9. Vstupy neuronu chápeme jako souřadnice bodu v n - rozměrném Euklidovském vstupním prostoru E_n. V tomto prostoru má rovnice nadroviny (v E_2 přímka, v E_3 rovina) tvar: $w_0 + \sum_{i=1}^{n} w_i x_i = 0$. Tato nadrovina dělí vstupní prostor na dva poloprostory. Souřadnice bodů $[x_1^+,...,x_n^+]$, které leží v jednom poloprostoru, splňují následující nerovnost: $w_0 + \sum_{i=1}^{n} w_i x_i^+ > 0$. Body $[x_1^-,...,x_n^-]$ z druhého poloprostoru pak vyhovují relaci s opačným relačním znaménkem: $w_0 + \sum_{i=1}^{n} w_i x_i^- < 0$. Synaptické váhy $\mathbf{w} = (w_0,..., w_n)$ chápeme jako koeficienty této nadroviny. Neuron Y tedy klasifikuje, ve kterém z obou poloprostorů určených nadrovinou leží bod, jehož

souřadnice jsou na vstupu, tj. realizuje dichotomii vstupního prostoru. Neuron Y je aktivní, je-li jeho stav y = 1 a pasivní, pokud je jeho stav y = 0.



Obrázek 9: Geometrická interpretace funkce neuronu

Shrnutí kapitoly



V této kapitole jste se stručně seznámili s principem činnosti biologického neuronu a se základním matematickým modelem biologického neuronu, tj. formálním neuronem.

Kontrolní otázky a úkoly:

1. Jaký je principem činnosti biologického neuronu?
2. Vysvětlete princip formálního neuronu.
3. Jaká je geometrická interpretace funkce neuronu?

Otázky k zamyšlení:

Vytvořte geometrickou interpretaci funkce jednoho neuronu pro Booleovské funkci OR ve 2-rozměrném Euklidovském prostoru. Vstupy neuronu jsou souřadnice bodu.

Citovaná a doporučená literatura

Polách, L. (2012) Neurony, neurotransmitery a synapse. [on line http://www.upsychiatra.cz/srozumitelne/neurony-neurotransmitery-a-synapse/] (Citováno 21.6.2013)

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

4 Hebbovo adaptační pravidlo

V této kapitole se dozvíte:

- Jaké jsou principy Hebbova učení.
- Jak probíhá adaptace podle Hebbova adaptačního pravidla.

Po jejím prostudování byste měli být schopni:

- vysvětlit princip Hebbova adaptačního pravidla,
- objasnit průběh adaptace podle Hebbova adaptačního pravidla.

Klíčová slova kapitoly: Hebbovo adaptační pravidlo, tréninkový vzor, váhový přírustek.

Průvodce studiem

V této kapitole si objasníme princip Hebbova adaptačního pravidla. Dříve než se pustíte do studia této kapitoly, důkladně se seznamte s problematikou formálního neuronu.

Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny, tak se pohodlně usaďte a nenechte se nikým a ničím rušit.

Hebbovo učení je založeno na myšlence, že váhové hodnoty na spojení mezi dvěma neurony, které jsou současně ve stavu „*on*“, budou narůstat a naopak, tj. váhové hodnoty na spojení mezi dvěma neurony, které jsou současně ve stavu „*off*“, se budou zmenšovat. Změna synaptické váhy spoje mezi dvěma neurony je úměrná jejich souhlasné aktivitě, tj. součinu jejich stavů. Donald Hebb tímto způsobem vysvětloval vznik podmíněných reflexů. Uvažujme jednovrstvou neuronovou síť, ve které jsou všechny vstupní neurony propojeny s jediným výstupní neuronem Y (viz. obr. 2), ale ne již navzájem mezi sebou. Pokud jsou složku vstupního vektoru $\boldsymbol{x} = (x_1, \ldots, x_n)$ reprezentovány v bipolární formě, lze složky příslušného váhového vektoru $\boldsymbol{w} = (w_1, \ldots, w_n)$ aktualizovat následovně:

$$w_i(new) = w_i(old) + x_i y,$$

kde *y* je výstup z neuronu *Y*.

Hebbovo adaptační pravidlo (Fausett, 1994):

Krok 0. Inicializace všech vah:

$w_i = 0$ *(i = 1* až *n)*

Krok 1. Pro každý vzor - tréninkový pár, tj. vstupní vektor (**s**) a příslušný výstup (**t**), opakovat následující kroky (2 až 4).

Krok 2. Aktivovat vstupní neurony:

$x_i = s_i$ *(i = 1* až *n).*

Krok 3. Aktivovat výstupní neuron:

$y = t$.

Krok 4. Aktualizovat váhy podle

$w_i(new) = w_i(old) + x_i y$ *(i = 1* až *n).*

Aktualizovat biasy podle

$b(new) = b(old) + y$.

Bias lze zapsat také jako váhovou hodnotu přiřazenou výstupu z neuronu, jehož aktivace má vždy hodnotu 1. Aktualizace váhových hodnot může být také vyjádřena ve vektorové formě jako

$$\mathbf{w}(new) = \mathbf{w}(old) + \mathbf{x}y.$$

Výše uvedený algoritmus je pouze jedním z mnoha způsobu implementace Hebbova pravidla učení. Tento algoritmus vyžaduje jen jeden průchod tréninkovou množinou. Existují však i jiné ekvivalentní metody nalezení vhodných váhových hodnot, které jsou popsány dále.

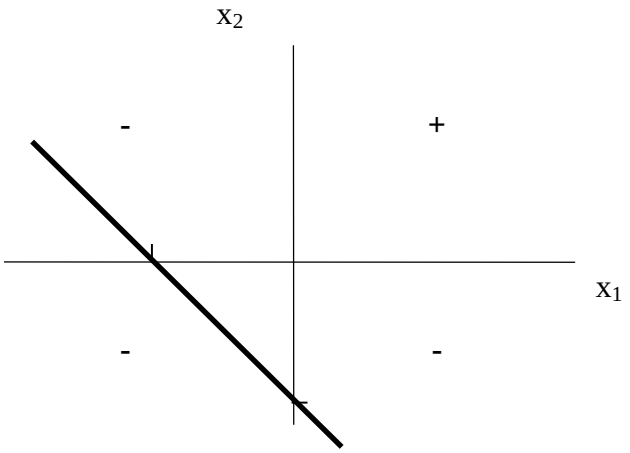
Příklad:

Hebbovovo pravidlo učení pro logickou funkci AND v bipolární reprezentaci můžeme zapsat následovně, viz tabulka 3:

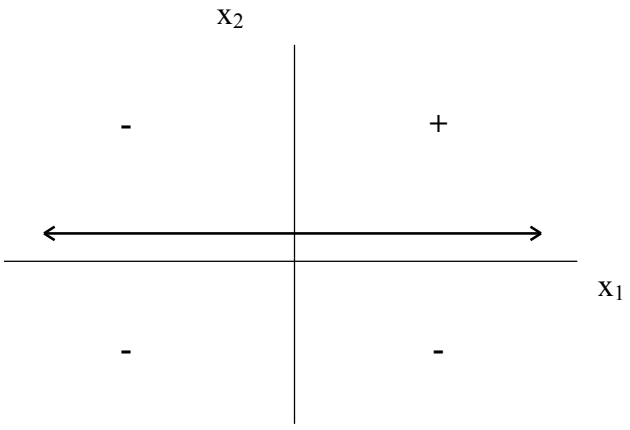
čas	**VSTUP**		**POŽADOVANÝ VÝSTUP**	**PŘÍRUSTKY VAH**			**VÁHOVÉ HODNOTY**		
	x_1	x_2	t	Δw_1	Δw_2	Δb	w_1	w_2	b
0							0	0	0
1	1	1	1	1	1	1	1	1	1
2	1	-1	-1	-1	1	-1	0	2	0
3	-1	1	-1	1	-1	-1	1	1	-1
4	-1	-1	-1	1	1	-1	2	2	-2

Tabulka 3: Hebbovovo pravidlo učení pro logickou funkci AND v bipolární reprezentaci

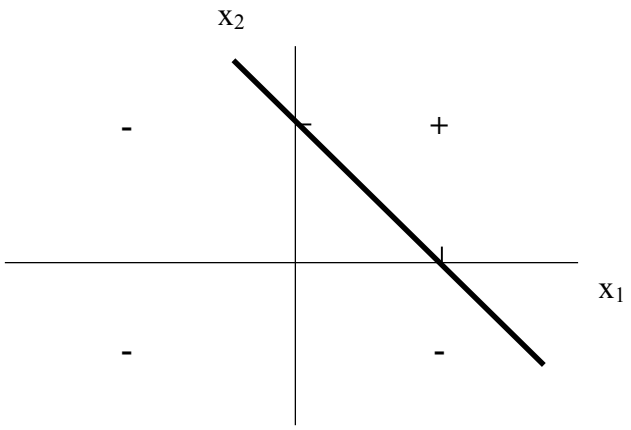
Grafický postup řešení je uveden na obrázku 10.



Rovnice přímky pro první tréninkový vzor: $x_2 = -x_1 - 1$

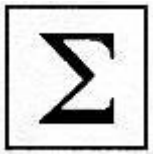


Rovnice přímky pro druhý tréninkový vzor: $x_2 = 0$.



Rovnice přímky pro 3. a 4. tréninkový vzor: $x_2 = -x_1 + 1$

Obrázek 10: Hebbovovo pravidlo učení pro logickou funkci AND v bipolární reprezentaci.



Shrnutí kapitoly

V této kapitole jsme si objesnili princip Hebbova adaptačního pravidla a na praktickém příkladu jsme si uvedli průběh adaptace prodle tohoto pravidla.

Kontrolní otázky a úkoly:

1. Jaký je princip Hebbova adaptačního pravidla.
2. Jaký je průběh adaptace podle Hebbova adaptačního pravidla.

Úkoly k textu

Realizujte Hebbovo pravidlo učení pro logickou funkci OR v bipolární reprezentaci.

Citovaná a doporučená literatura

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

5 Perceptron

V této kapitole se dozvíte:

- Jaký je princip adaptačního algoritmu perceptronu.
- Jak pracuje perceptron.

Po jejím prostudování byste měli být schopni:

- vysvětlit princip adaptačního algoritmu perceptronu,
- charakterizovat nejjednodušší neuronovou síť s jedním pracovním neuronem.

Klíčová slova kapitoly: Perceptron, učení s učitelem.

Průvodce studiem

V této kapitole je představen perceptron, který lze považovat za nejjednodušší neuronovou síť s jedním pracovním neuronem. Na jeho adaptačním algoritmu si vysvětlíme proces učení s učitelem.

Na zvládnutí této kapitoly budete potřebovat asi 3 hodiny.

Autorem této nejjednodušší neuronové sítě je Frank Rosenblatt (r. 1957). Za typický perceptron je považována jednoduchá neuronová síť s n vstupy (x_1, x_2, ..., x_n) a jedním pracovním neuronem spojeným se všemi svými vstupy. Každému takovému spojení je přiřazena váhová hodnota (w_1, w_2, ..., w_n). Signál přenášený vstupními neurony je buď binární (tj. má hodnotu 0 nebo 1), nebo bipolární (tj. má hodnotu -1, 0 nebo 1). Výstupem z perceptronu je pak $y = f(y\_in)$. Aktivační funkce f má tvar:

$$f(y\_in) = \begin{cases} 1 & pokud\ y\_in > \theta \\ 0 & pokud\ -\theta \le y\_in \le \theta \\ -1 & pokud\ y\_in < -\theta \end{cases},$$

kde θ je libovolný, ale pevný práh aktivační funkce f.

Váhové hodnoty jsou adaptovány podle adaptačního pravidla perceptronu tak, aby diference mezi skutečným a požadovaným výstupem byla co nejmenší.

Adaptační pravidlo perceptronu je mnohem silnější než Hebbovo adaptační pravidlo.

Adaptační algoritmus perceptronu (Fausett, 1994)

Krok 0. Inicializace vah w_i *(i = 1* až *n)* a biasu b *malými náhodnými čísly.*

Přiřazení inicializační hodnoty koeficientu učení α *(0 < $\alpha \leq$ 1).*

Krok 1. Dokud není splněna podmínka ukončení výpočtu, opakovat kroky (2 až 6).

Krok 2. Pro každý tréninkový pár **s:**t (tj. vstupní vektor **s** a příslušný výstup t**)**, provádět kroky (3 až 5).

Krok 3. Aktivuj vstupní neurony:

$x_i = s_i$.

Krok 4 Vypočítej skutečnou hodnotu na výstupu:

$$y\_in = b + \sum_i x_i w_i ;.$$

$$y = \begin{cases} 1 & pokud\ y\_in > \theta \\ 0 & pokud - \theta \leq y\_in \leq \theta \\ -1 & pokud\ y\_in < -\theta \end{cases}$$

Krok 5 Aktualizuj váhové hodnoty a bias pro daný vzor

jestliže $y \neq t$,

$w_i(new) = w_i(old) + \alpha\, t\, x_i$ *(i = 1* až *n).*

$b(new) = b(old) + \alpha\, t.$

jinak

$w_i(new) = w_i(old)$

$b(new) = b(old)$

Krok 6. Podmínka ukončení:

jestliže ve *2. kroku* již nenastává žádná změna váhových hodnot, stop; jinak, pokračovat.

Aktualizaci podléhají pouze ty váhové hodnoty, které neprodukují požadovaný výstup *y*. To znamená, že čím více tréninkových vzorů má korektní výstupy, tím méně je potřeba času k jejich tréninku. Práh aktivační funkce je pevná nezáporná hodnota. Tvar aktivační funkce pracovního neuronu je takový, že umožňuje vznik pásu pevné šířky (určené hodnotou prahu) oddělujícího oblast pozitivní odezvy od oblasti negativní odezvy na vstupní signál.

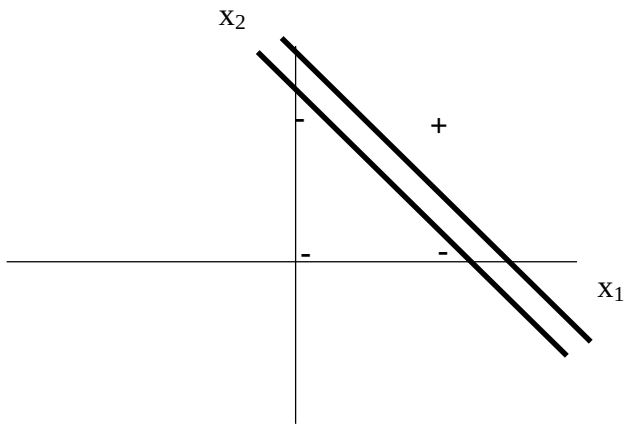
Příklad:

Adaptační algoritmus perceptronu pro logickou funkci AND: binární vstupní hodnoty, bipolární výstupní hodnoty. Pro jednoduchost předpokládejme, že práh $\theta = 0{,}2$ a koeficient učení $\alpha = 1$. V tabulce 4 je uvedeno řešení.

čas	**VSTUP**		**VÝSTUP**			**PŘÍRUSTKY VAH**			**VÁHOVÉ HODNOTY**		
	x_1	x_2	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
0									0	0	0
1	1	1	0	0	1	1	1	1	1	1	1
2	1	0	2	1	-1	-1	0	-1	0	1	0
3	0	1	1	1	-1	0	-1	-1	0	0	-1
4	0	0	-1	-1	-1	1	0	0	0	0	-1
. . .											
37	1	1	1	1	1	0	0	0	2	3	-4
38	1	0	-2	-1	-1	0	0	0	2	3	-4
39	0	1	-1	-1	-1	0	0	0	2	3	-4
40	0	0	-4	-1	-1	0	0	0	2	3	-4

Tabulka č. 4 Algoritmus perceptronu pro logickou funkci AND pro binární vstupní hodnoty a bipolární výstupní hodnoty.

Hraniční pás pro logickou funkci AND po adaptaci algoritmem perceptronu je uvedeno na obrázku 11.



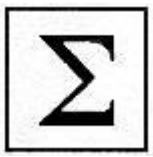
Obrázek 11: Hraniční pás pro logickou funkci AND po adaptaci algoritmem perceptronu.

Oblast kladné odezvy je dána body, pro které platí: $2x_1 + 3x_2 - 4 > 0{,}2$

a hraniční přímka této oblasti má tvar: $x_2 = -\frac{2}{3}x_1 + \frac{7}{5}$.

Oblast záporné odezvy je dána body, pro které platí: $2x_1 + 3x_2 - 4 < -0{,}2$.

a hraniční přímka této oblasti má tvar: $x_2 = -\frac{2}{3}x_1 + \frac{19}{15}$.



Shrnutí kapitoly

V této kapitole jsme byl stručně představen perceptron, který lze považovat za nejjednodušší neuronovou síť s jedním pracovním neuronem. Na jeho adaptačním algoritmu jsme si vysvětlili proces učení s učitelem.

Kontrolní otázky a úkoly:

1. Jaký je princip adaptačního algoritmu perceptronu?
2. Jak pracuje perceptron?

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu perceptronu.

Citovaná a doporučená literatura

Rosenblatt, F. (1958). The perceptron. Psych. Rev, 65(6), 386-408.

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

6 Adaline

V této kapitole se dozvíte:

- Jaký je princip adaptačního algoritmu neuronu Adaline.
- Budete schopni porovnat adaptační algoritmus neuronu Adaline s adaptačním algoritmem percetronu

Po jejím prostudování byste měli být schopni:

- vysvětlit princip adaptačního algoritmu neuronu Adaline,
- srovnat adaptační algoritmus neuronu Adaline s adaptačním algoritmem percetronu.

Klíčová slova kapitoly: Adaline, geometrická interpretace funkce neuronu Adaline.

Průvodce studiem

V této kapitole si vysvětlíme princip adaptačního algoritmu neuronu Adaline. Tento adaptační algoritmus je zde srovnán s adaptačním algoritmem percetronu.

Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny.

Adaline, tj. Adaptive Linear Neuron. Pro své vstupy obvykle používá bipolární aktivaci (1 nebo -1), výstupní hodnota je nejčastěji také bipolární. Adaline má rovněž bias chovající se jako regulovatelná váha (w_0) přiřazená spojení, které vychází z neuronu, jehož aktivace je vždy 1.

Adaptační algoritmus pro Adaline (Fausett, 1994)**:**

Krok 0. Inicializace vah malými náhodnými hodnotami.
Přiřazení inicializační hodnoty koeficientu učení α
(viz poznámky za algoritmem).

Krok 1. Dokud není splněna podmínka ukončení výpočtu, opakovat kroky (2 až 6).

Krok 2. Pro každý bipolární tréninkový pár **s:**t (tj. vstupní vektor **s** a příslušný výstup t**)**, provádět kroky (3 až 5).

Krok 3. Aktivovat vstupní neurony:

$x_i = s_i$.

Krok 4 Vypočítat skutečnou hodnotu na výstupu:

$y\_in = b + \sum_i x_i w_i$;.

$y = y\_in$.

Krok 5 Aktualizovat váhové hodnoty a

i = 1, ..., n:

$w_i(new) = w_i(old) + \alpha\,(t - y\_in)\, x_i$.

$b(new) = b(old) + \alpha\,(t - y\_in)$.

Krok 6. Podmínka ukončení:
jestliže největší změna váhových hodnot, která se vyskytuje v kroku 2 je menší než maximální povolená chyba, stop; jinak, pokračovat.

Geometrický význam funkce Adaline se nepatrně liší od perceptronu. Uvažujme vstup $\boldsymbol{x}=(x_1, ..., x_n)$, tj. bod $[x_1, ..., x_n]$ v *n*-rozměrném vstupním prostoru. Nadrovina s koeficienty ***w*** pro daný neuron Adaline určená rovnicí

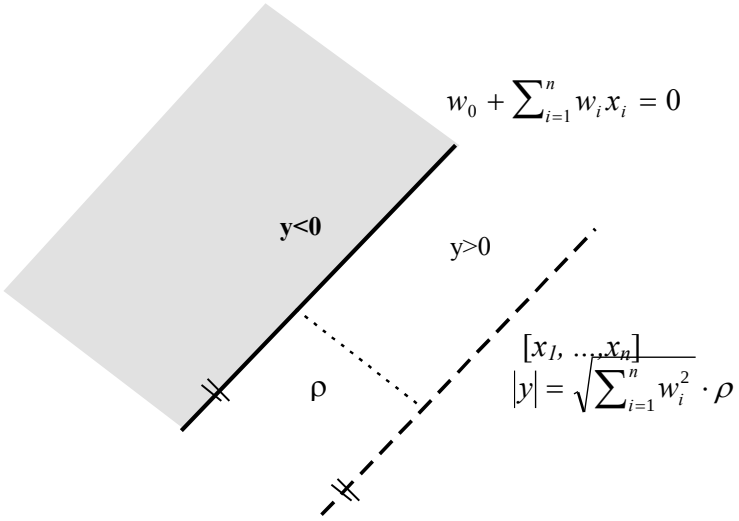
$$w_0 + \sum_{i=1}^{n} w_i x_i = 0$$

rozděluje tento prostor na dva poloprostory, ve kterých má hodnota výstupu *y* zapsaného rovnicí

$$y = \sum_{i=1}^{n} w_i x_i$$

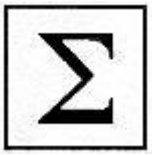
odlišné znaménko (tj. je buď kladná, nebo záporná).

Absolutní hodnota výstupu z neuronu Adaline závisí lineárně na vzdálenosti vstupu x od nadroviny ve vstupním prostoru. Body ze vstupního prostoru, které mají stejný výstup, leží na jedné nadrovině rovnoběžné se separující nadrovinou. Uvedená situace je načrtnuta na obrázku 12, kde nadrovina určená stejným výstupem je znázorněna přerušovanou čarou.



Obrázek 12: Geometrická interpretace funkce neuronu Adaline.

Shrnutí kapitoly



V této kapitole jsme si vysvětlili princip adaptačního algoritmu neuronu Adaline. Tento adaptační algoritmus byl nýsledně srovnán s adaptačním algoritmem percetronu.

Kontrolní otázky a úkoly:

1. Vysvětlete princip adaptačního algoritmu neuronu Adaline.
2. Srovnejte adaptační algoritmus neuronu Adaline s adaptačním algoritmem percetronu.
3. Jaká je geometrická interpretace funkce neuronu Adaline?

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu neuronu Adaline.

Citovaná a doporučená literatura

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

7 Madaline

V této kapitole se dozvíte:

- Jaká je topologie neuronové sítě Madaline.
- Jaký je princip adaptačního algoritmu Madaline.

Po jejím prostudování byste měli být schopni:

- vysvětlit princip adaptačního algoritmu Madaline,
- objasnit geometrickou interpretaci funkce neuronové sítě Madaline.

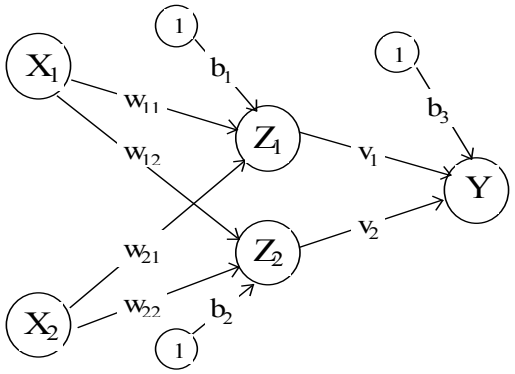
Klíčová slova kapitoly: Madaline, geometrická interpretace funkce neuronové sítě Madaline.

Průvodce studiem

V této kapitole si vysvětlíme princip adaptačního algoritmu neuronové sítě Madaline. Dále si zde objasníme geometrickou interpretaci funkce neuronové sítě Madaline.

Na zvládnutí této kapitoly budete potřebovat asi 3 hodiny.

Madaline, tj. *Many Adaptive Linear Neurons*. Základním prvkem v tomto modelu je neuron Adaline, který je velmi podobný perceptronu. Jednoduchá architektura neuronové sítě Madaline je zobrazena na obrázku 13. Výstupy (z_1 a z_2) z obou skrytých neuronů typu Adaline (Z_1 a Z_2), jsou určeny stejnými signály (x_1 a x_2) vycházejícími z neuronů X_1 a X_2, které samozřejmě závisí na příslušné prahové funkci. Pak i skutečný výstup y je nelineární funkcí vstupního vektoru (x_1, x_2) a příslušné prahové funkce. Použití skrytých neuronů Z_1 a Z_2 sice dává síti větší výpočtové možnosti, ale naproti tomu komplikuje adaptační proces.



Obrázek 13: Madaline se dvěma skrytými neurony Adaline a jedním výstupním neuronem Adaline.

Původní adaptační algoritmus MRI (z roku 1960) adaptuje pouze váhové hodnoty příslušející oběma skrytým neuronům, zatímco váhové hodnoty příslušející výstupnímu neuronu jsou fixní. Adaptační algoritmus MRII (z roku 1987) upravuje všechny váhové hodnoty. Dále budeme pracovat pouze s adaptačním algoritmem MRI: Váhové hodnoty v_1 a v_2 a bias b_3, příslušející výstupnímu neuronu *Y*, jsou určeny tak, že výstupní signál z *Y* je roven *1*, pokud je alespoň jedna hodnota signálu vycházejícího ze skrytých neuronů (tj. Z_1 a Z_2 nebo obou z nich) rovna jedné. Pokud jsou oba signály vysílané ze Z_1 i Z_2 rovny -1, má výstupní signál z *Y* hodnotu *-1*. Jinými slovy, výstupní neuron *Y* provádí logickou funkci „OR“ na signálech vysílaných z neuronů Z_1 a Z_2. Můžeme tedy přiřadit

$$v_1 = \frac{1}{2},$$
$$v_2 = \frac{1}{2},$$
$$b_3 = \frac{1}{2}.$$

Váhové hodnoty příslušející prvnímu skrytému neuronu Adaline (w_{11} a w_{21}) a váhové hodnoty příslušející druhému skrytému neuronu Adaline (w_{12} a w_{22}) jsou adaptovány podle algoritmu MRI takto:

Aktivační funkce pro Z_1, Z_2 a Y je dána následovně:

$$f(x) = \begin{cases} 1 & \text{pokud } x \geq 0; \\ -1 & \text{pokud } x < 0. \end{cases}$$

Adaptační algoritmus MRI (Fausett, 1994)**:**

Krok 0. Váhové hodnoty v_1 a v_2 a bias b_3 jsou inicializovány výše uvedeným způsobem.
Inicializace zbývajících vah malými náhodnými hodnotami.
Přiřazení inicializační hodnoty koeficientu učení α. stejným způsobem jako v adaptačním algoritmu pro neuron Adaline.

Krok 1. Dokud není splněna podmínka ukončení výpočtu, opakovat kroky (2 až 8).

Krok 2. Pro každý bipolární tréninkový pár **s:t** provádět kroky (3 až 7).

Krok 3. Aktivovat vstupní neurony:
$x_i = s_i$.

Krok 4 Vypočítat vstupní hodnoty skrytých neuronů:
$z\_in_1 = b_1 + x_1 w_{11} + x_2 w_{21}$,
$z\_in_2 = b_2 + x_1 w_{12} + x_2 w_{22}$.

Krok 5 Stanovení výstupních hodnot skrytých neuronů:
$z_1 = f(z\_in_1)$,
$z_2 = f(z\_in_2)$.

Krok 6 Stanovení skutečné výstupní hodnoty signálu neuronové sítě Madaline:
$y\_in = b_3 + z_1 v_1 + z_2 v_2$;
$y = f(y\_in)$.

Krok 7 Aktualizovat váhové hodnoty*:*
Pokud je $y = t$, nenastávají žádné změny.
Jinak (pro $y \neq t$)*:*

Je-li $t = 1$, potom pro váhové hodnoty na spojeních vedoucích k Z_J ($J=1,2$) platí:

$w_{iJ}(new) = w_{iJ}(old) + \alpha(1 - z\_in_J)\, x_i$.

$b_J(new) = b_J(old) + \alpha(1 - z\_in_J)$.

Je-li $t = -1$, potom pro váhové hodnoty na spojeních vedoucích k Z_K ($K=1,2$) platí:

$w_{iK}(new) = w_{iK}(old) + \alpha(-1 - z\_in_K)\, x_i$.

$b_K(new) = b_K(old) + \alpha(-1 - z\_in_K)$.

Krok 8. Podmínka ukončení:
pokud již nenastávají žádné změny váhových hodnot nebo pokud již bylo vykonáno maximálně definované množství váhových změn, *stop*; jinak, *pokračovat*.

Příklad:

Adaptační algoritmus MRI pro logickou funkci XOR pro bipolární vstupní i výstupní hodnoty. V tabulce 5 je uvedena trénovací množina a řešení je zapsáno v následujících krocích.

VSTUP		**POŽADOVANÝ**
x_1	x_2	**VÝSTUP**
1	1	-1
1	-1	1
-1	1	1
-1	-1	-1

Tabulka 5: Algoritmus Madaline pro logickou funkci XOR pro bipolární vstupní i výstupní hodnoty.

Krok 0. $\alpha = 0.5$;

Inicializace váhových hodnot, viz tabulka 5.

váhy vedoucí do Z_1			**váhy vedoucí do Z_2**			**váhy vedoucí do Y**		
w_{11}	w_{21}	b_1	w_{12}	w_{22}	b_2	v_1	v_2	b_3
0,05	0,2	0,3	0,1	0,2	0,15	0,5	0,5	0,5

Tabulka 5 Inicializace váhových hodnot pro algoritmus Madaline pro logickou funkci XOR pro bipolární vstupní i výstupní hodnoty.

Krok 1. Adaptace:

Krok 2. Pro první tréninkový pár; (1,1):-1

Krok 3. $x_1 = 1$,

$x_2 = 1$.

Krok 4 $z\_in_1 = 0{,}3 + 0{,}05 + 0{,}2 = 0{,}55$,

$z\_in_2 = 0{,}15 + 0{,}1 + 0{,}2 = 0{,}45$

Krok 5 $z_1 = 1$,

$z_2 = 1$.

Krok 6 $y\_in = 0{,}5 + 0{,}5 + 0{,}5$;

$y = 1$.

Krok 7 $t - y = -1-1 = -2 \neq 0$,

Pokud je $t = -1$, potom aktualizovat váhové hodnoty na spojeních vedoucích k Z_1:

$$b_1(new) = b_1(old) + \alpha(-1 - z\_in_1)$$
$$= 0{,}3 + (0{,}5)(-1{,}55)$$
$$= -0{,}475$$

$$w_{11}(new) = w_{11}(old) + \alpha(-1 - z\_in_1)x_1$$
$$= 0{,}05 + (0{,}5)(-1{,}55)$$
$$= -0{,}725$$

$$w_{21}(new) = w_{21}(old) + \alpha(-1 - z\_in_1)x_2$$
$$= 0{,}2 + (0{,}5)(-1{,}55)$$
$$= -0{,}575$$

a

aktualizovat váhové hodnoty na spojeních vedoucích k Z_2:

$$b_2(new) = b_2(old) + \alpha(-1 - z\_in_2)$$
$$= 0{,}15 + (0{,}5)(-1{,}45)$$
$$= -0{,}575$$

$$w_{12}(new) = w_{12}(old) + \alpha(-1 - z\_in_2)x_1$$
$$= 0{,}1 + (0{,}5)(-1{,}45)$$
$$= -0{,}625$$
$$w_{22}(new) = w_{22}(old) + \alpha(-1 - z\_in_2)x_2$$
$$= 0{,}2 + (0{,}5)(-1{,}45)$$
$$= -0{,}525$$

Po čtyřech tréninkových cyklech, byly nalezeny tyto váhové hodnoty:

w_{11} = - 0,73 w_{12} = 1,27

w_{21} =1,53 w_{22} = - 1,33

b_1 = - 0,99 b_2 = - 1,09

Geometrická interpretace nalezených váhových hodnot:

Oblast kladné odezvy vznikne sjednocením obou oblastí pozitivní odezvy skrytých neuronů Z_1 a Z_2.

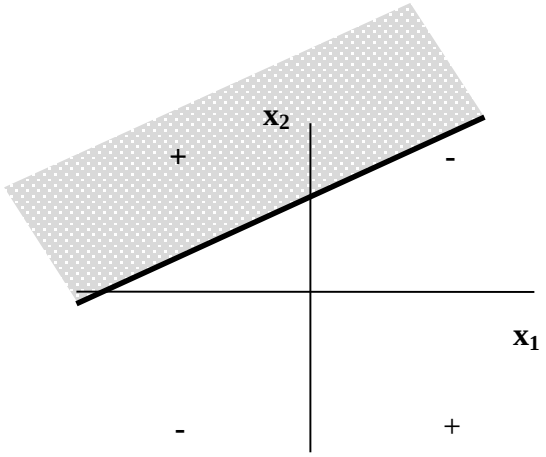
Pro skrytý neuron Z_1 má hraniční přímka tvar

$$x_2 = -\frac{w_{11}}{w_{21}}x_1 - \frac{b_1}{w_{21}}$$
$$= \frac{0{,}73}{1{,}53}x_1 + \frac{0{,}99}{1{,}53}$$
$$= 0{,}48x_1 + 0{,}65.$$

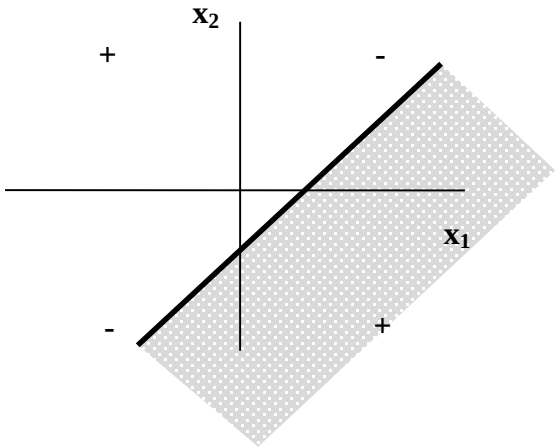
Pro skrytý neuron Z_2 má hraniční přímka tvar

$$x_2 = -\frac{w_{12}}{w_{221}}x_1 - \frac{b_2}{w_{22}}$$
$$= \frac{1{,}27}{1{,}33}x_1 + \frac{1{,}09}{1{,}33}$$
$$= 0{,}96x_1 - 0{,}82.$$

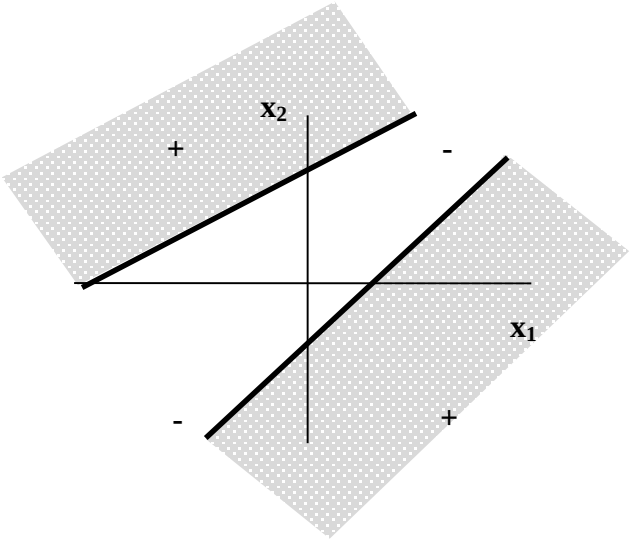
Vypočítané oblasti kladné a záporné odezvy na vstupní signál jsou znázorněny na následujících obrázcích 14-16:



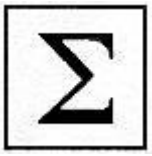
Obrázek 14: Oblast kladné odezvy pro Z_1.



Obrázek 15: Oblast kladné odezvy pro Z_2.



Obrázek 16: Oblast kladné odezvy pro Madaline pro XOR funkci.



Shrnutí kapitoly

V této kapitole jsme si vysvětlili princip adaptačního algoritmu neuronové sítě Madaline a rovněž jsme si na řešeném příkladu objasnili geometrickou interpretaci funkce neuronové sítě Madaline.

Kontrolní otázky a úkoly:

1. Jaká je topologie neuronové sítě Madaline?
2. Jaký je princip adaptačního algoritmu Madaline?
3. Vysvětlete geometrickou interpretaci funkce neuronové sítě Madaline.

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu neuronové sítě Madaline.

Citovaná a doporučená literatura

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

8 Klasifikace

V této kapitole se dozvíte:

- Co je podstatou klasifikace objektů do tříd.
- Proč je klasifikace jednou z oblastí, kde se neuronové sítě velmi dobře uplatňují.
- Jaký je princip dvouhodnotové a vícehodnotové klasifikace.

Po jejím prostudování byste měli být schopni:

- vysvětlit možnosti klasifikace za použití umělých neuronových sítí,
- objasnit podstatu klasifikace objektů do tříd.

Klíčová slova kapitoly: Klasifikace, charakteristické rysy, klasifikátor.

Průvodce studiem

V této kapitole se budeme zabývat problematikou rozpoznávání vzorů a klasifikace. Zaměříme se především na umělé neuronové sítě a možnosti jejich klasifikace.

Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny.

Klasifikace je činnost, při které se posuzované objekty zařazují do příslušných tříd. Z matematického hlediska lze tyto činnosti nazývat funkční aproximací (Zelinka, 1999).

Klasifikace je jednou z oblastí, kde se neuronové sítě velmi dobře uplatňují. Klasifikace znamená v podstatě ohodnocení daného problému a jeho zařazení do příslušné třídy. Je to aktivita, kterou člověk provádí dnes a denně, aniž by si to uvědomoval. Například když v obchodě vybíráme mezi dvěma výrobky, musíme rozhodnout, který je horší a který lepší (nyní není podstatné z jakého hlediska). Pokud chceme jít do kina, obvykle předem oklasifikujeme film a na základě tohoto výsledku se rozhodneme. Pokud se nad otázkou klasifikace v našem životě zamyslíte, jistě sami najdete mnoho podobných příkladů.

Klasifikace není vždy jednoduchá záležitost. Mnohdy se hranice, na jejichž základě se zařazují klasifikované členy do tříd, překrývají. Proč se tedy na klasifikaci používají neuronové sítě? Důvodů je hned několik. Neuronová síť svá rozhodnutí provádí prakticky ihned i v případě, že vstupních informací popisující daný problém je poměrně dost. Co to znamená „dost vstupních informací" ? Představte si, že máme na základě vstupních informací binárního charakteru rozhodnout, kam zařadíme daný výrobek. Pokud nám zmíněný výrobek popisují dvě vstupní proměnné, pak máme $2^2 = 4$ kombinace, na základě kterých můžeme tento výrobek ohodnotit. Ale co když náš výrobek popisují ne dvě, ale tři (8 kombinací), čtyři (16 kombinací), pět (32 kombinací) a více vstupních proměnných? Jistě každý z vás najde ve svém životě situaci, kdy se nemohl rozhodnout nebo se rozhodl špatně a přitom vycházel z malého počtu vstupních proměnných. To neuronové síti nehrozí, pokud je dobře natrénována, neboť její výhodou je, že zpracuje vše, co jí předložíme. To znamená, že pokud předložíme síti pro klasifikaci např. 30 či více vstupů popisující daný objekt na základě např. sonarového odrazu, pak je síť zpracuje. Co ale udělá člověk? Začne vypouštět některé vstupní proměnné, a pokud není dostatečné zkušený, pak může vypustit i proměnné, které jsou pro klasifikaci důležité. Jistě nyní namítáte, že neuronová síť vlastně jen zpracuje vše, co se jí předloží. Není to tak zcela pravda, protože neuronové sítě umí pomocí tzv. neurální citlivostní analýzy určit, která informace je vhodná pro zpracování a která není **(**Barnsley, 1993). Zatím jsme ale pouze hovořili o binárním rozhodování. Představte si, že by každá z výše uvedených 30 možností mohla nabývat hodnot ne 0 či 1, ale z intervalu <0, 1>. Tím se náš problém klasifikace podstatně zkomplikuje. Klasifikaci tedy můžeme provádět pomocí binární či spojité funkce pro zařazení do jedné a více tříd. V případě použití neuronových sítí se tedy problém klasifikace dělí na klasifikaci dvouhodnotovou a vícehodnotovou. Pro klasifikaci jsou „důležité“ výstupní neurony, protože právě ty nám určují, do jaké třídy patří momentálně hodnocený objekt.

8.1 Dvouhodnotová klasifikace

Dvouhodnotová klasifikace patří mezi nejjednodušší klasifikace vůbec. V jejím případě se v podstatě rozhoduje mezi „černou“ a „bílou“, „ano“ a „ne“ či 1 a 0. Tento typ klasifikace lze realizovat tak, že neurony v neuronové síti mají

skokovou funkci, která umožňuje nastavení výstupu neuronu pouze ve dvou výše zmíněných hodnotách. Na realizaci tohoto typu klasifikace v podstatě stačí jeden výstupní neuron (pouze v případě, že pracujeme s jednou třídou), který nabývá hodnot 0 a 1. Dvouhodnotovou klasifikaci lze také provést pomocí logistické funkce, která je sice kontinuální, nicméně není problém naučit síť tak, aby výstupní neurony dosahovaly binárních hodnot. V případě použití jakékoliv kontinuální funkce musíme počítat s tím, že výstupní neurony nikdy nedosáhnou přesně požadovaných hodnot, ale že se vždy naučí s nějakou chybou. Jinými slovy, místo výstupní hodnoty 1 nám daný neuron poskytne např. 0.9, 0.88, a naopak. I tyto výsledky jsou jistě akceptovatelné, nicméně, při vhodných či spíše nevhodných podmínkách, nám může síť odpovědět hodnotou 0.7, 0.65,... Z těchto důvodů je dobré, určit prahy, které jasně vymezují co je 0 a co 1. Existence prahů nabývá při vícehodnotové klasifikaci na významu ještě více.

8.2 Vícehodnotová klasifikace

Tato klasifikace je složitější než klasifikace binární. Při ní, jak už z názvu vyplývá, existuje možnost zařadit klasifikovaný objekt do více než dvou tříd. To lze realizovat jak spojitou funkcí, tak funkcí binární. V případě, že použijeme spojitou funkci, pak pro zařazení objektu do dané třídy stačí, když na výstupu sítě je jen jeden neuron, jehož funkce je rozdělena což znamená, že např. třídě A přiřadíme hodnotu 0 - 0.3, třídě B 0.3 - 0.7 a třídě C 0.7 - 1. Toto dělení může být i hrubé či jemné podle potřeb uživatele. Mějme však na paměti, že díky nepřesnosti učení sítě může dojít k špatnému zařazení objektu, což v některých aplikacích nemusí mít vážné následky, zatímco v jiných aplikacích to může mít následky katastrofální. Tomu se lze vyhnout pomocí tzv. „vícenásobné“ dvouhodnotové klasifikace. Při této klasifikaci už nevystačíme s jedním výstupním neuronem (ten je schopen nabýt pouze dvou hodnot), ale musíme použít více neuronů, obvykle tolik, kolik je tříd. V tomto případě hodnota 1 znamená, že daný objekt patří do třídy zastoupené příslušným neuronem a hodnota 0, že tam nepatří.

Pokud použijeme pro vícehodnotovou klasifikaci binární neurony, pak lze pro zařazení objektu do dané třídy použít dvojí způsob kódování - přímý a

kombinovaný. Přímý způsob je přiřazení jedné třídy jednomu neuronu, tj. pokud je stav neuronu 0, pak daný objekt do třídy nepatří a pokud jeho stav 1, pak daný objekt do třídy patří. Jiné třídě se přiřadí další neuron. Kombinované kódování je založeno na binární povaze hodnot neuronu - je to v podstatě přepočet z binárních hodnot. Jinými slovy, pokud máme např. 8 tříd, pak pro jejich realizaci stačí 3 neurony (2^3=8) na rozdíl od přímého kódování, kde bychom potřebovali celkem 8 neuronů. Na druhou stranu větší počet neuronů je nevýhodný, protože znamená delší čas učení a tím i větší finanční náklady.

Pokud však máme hodnotit objekt, který nepatří do žádné regulérní třídy, pak abychom se tomuto problému vyhnuli, je dobré přidat další třídu „zamítnutí“ a do trénovací množiny přidat představitele této třídy, aby síť dokázala v budoucím rozhodování na takové objekty reagovat. Pokud bychom tak neučinili, pak to, že všechny neurony jsou v 0 ještě nemusí nutně znamenat, že hodnocený objekt nepatří do žádné třídy. Může to být např. člen některé okrajové třídy, jehož charakteristický vzorek nebyl v trénovací množině zastoupen.

8.3 Neuronové sítě a jejich možnosti klasifikace

Klasifikace je jednou z hlavních aplikačních oblastí pro umělé neuronové sítě (Bishop, 1995). Na následujícím obrázku 17 (Beale, 1992) jsou souhrnně zobrazeny různé typy neuronových sítí (tj. neuronové sítě s různým počtem vnitřních vrstev) a jejich možnosti klasifikace.

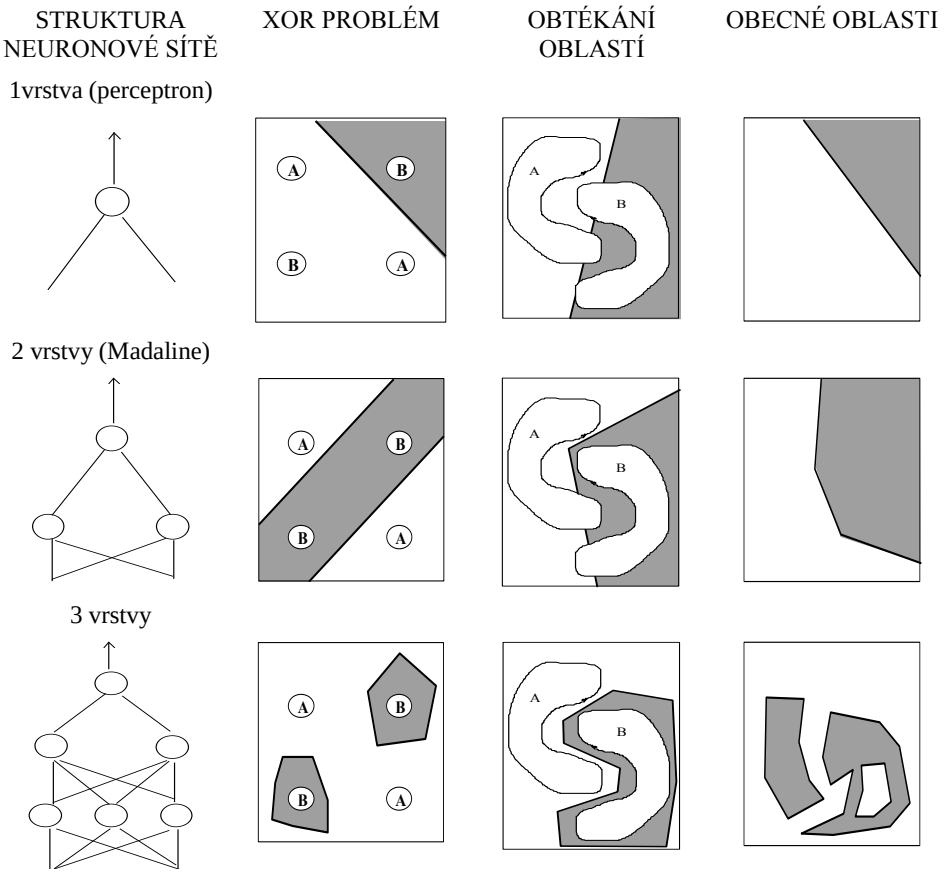
Použití klasifikátorů na bázi umělých neuronových sítí

V mnoha případech se jedná o systémy, kde klasifikace reprezentuje pouze součást řešení a celkové řešení využívá řadu dalších metod.

- OCR programy, které musí na digitalizovaném dokumentu rozpoznat jednotlivá písmena (OCR software se dodává k většině scannerů).
- kontrola dopravy, kde se klasifikace používá při detekci a rozpoznávání poznávacích značek aut
- průmyslová kontrola kvality výrobků
- rozpoznání textury povrchů

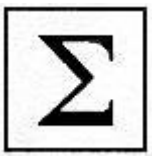
- predikce trendu časové řady (resp. prognóz vývoje ekonomických ukazatelů poskytuje managementu podniků důležité informace pro strategické plánování)
- automatické řízení systému v reálném čase (adaptace systému na změny během jeho provozu v reálném čase)

a další.



Obrázek 17: Neuronové sítě s různým počtem vnitřních vrstev a jejich možnosti klasifikace.

Shrnutí kapitoly



V této kapitole jsme se zabývali problematikou rozpoznávání vzorů a klasifikace. Zaměřili jsme se především možnosti uplatnění umělých

neuronových sítí při řešení úloh z oblasti rozpoznávání vzorů a při klasifikaci.

Kontrolní otázky a úkoly:

1. Co je podstatou klasifikace objektů do tříd?
2. Proč je klasifikace jednou z oblastí, kde se neuronové sítě velmi dobře uplatňují?
3. Jaký je princip dvouhodnotové a vícehodnotové klasifikace?

Citovaná a doporučená literatura

Zelinka, I. (1999) Aplikovaná informatika. Učební texty VUT v Brně.

Beale, R. - Jackson, T. (1992) Neural Computing: An Introduction. J W Arrowsmith Ltd, Bristol, Greit Britain.

Bishop, C. M. (1995) Neural Networks for Pattern Recognition. Oxford: Calderon Press. ISBN 0-19-853864-2.

9 Backpropagation

V této kapitole se dozvíte:

- Jak probíhá adaptace metodou backpropagation.
- Jaká je topologie vícevrstvé neuronové sítě.

Po jejím prostudování byste měli být schopni:

- vysvětlit princip adaptace metodou backpropagation,
- objasnit dopředné (feedforward) šíření signálu,
- charakterizovat topologii vícevrstvé sítě.

Klíčová slova kapitoly: Backpropagation, generalizace, overfitting, dopředné šíření signálu.

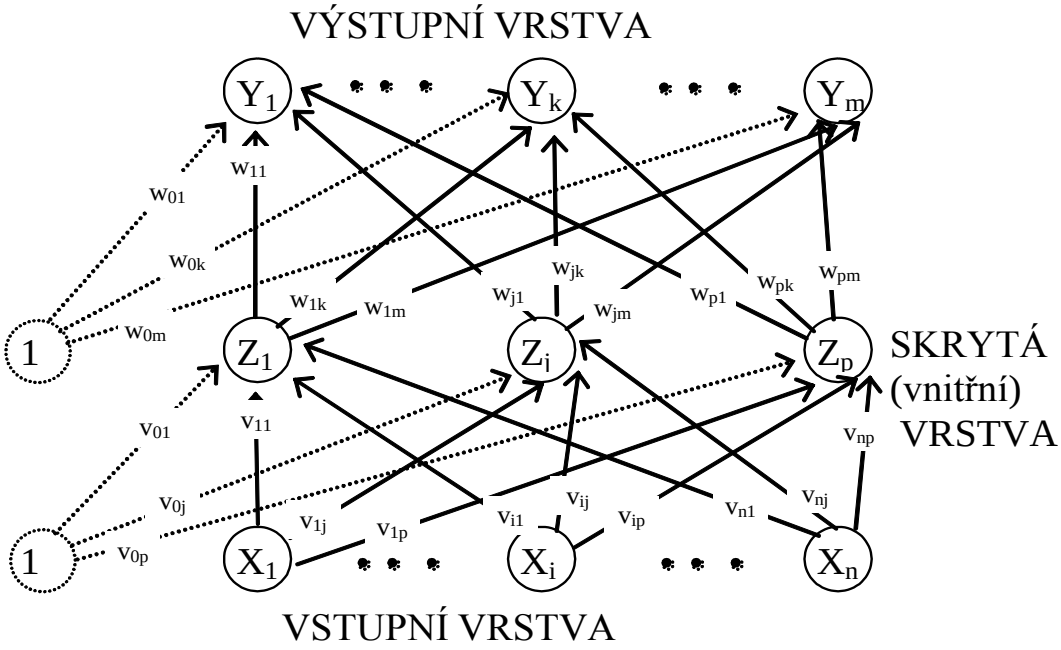
Průvodce studiem

V této kapitole rozebereme problematiku vhodné volby topologie vícevrstvé neuronové sítě, která by měla odpovídat složitosti řešeného problému. Dále se podrobně seznámíte s adaptačním algoritmem zpětného šíření chyby (backpropagation), jež je používán v přibližně 80% všech aplikací neuronových (tj. je nejrozšířenějším adaptačním algoritmem vícevrstvých neuronových sítí). Na zvládnutí této kapitoly budete potřebovat asi 4 hodiny.

9.1 Topologie vícevrstvé sítě

Pravděpodobně nejrozšířenější způsob propojení neuronů se sigmoidní aktivační funkcí jsou vícevrstvé sítě. Vícevrstvá neuronová síť s jednou vnitřní vrstvou neuronů (neurony jsou označeny Z_i, $j = 1,..., p$) je zobrazena na obrázku 18. Výstupní neurony (neurony jsou označeny Y_k, $k = 1,..., m$). Neurony ve výstupní a vnitřní vrstvě musí mít definovaný bias. Typické označení pro bias k. neuronu (Y_k) ve výstupní vrstvě je w_{0k}, a typické označení pro bias j. neuronu (Z_j) ve vnitřní vrstvě je v_{0j}. Bias (např. j. neuronu) odpovídá, jak již bylo dříve uvedeno, váhové hodnotě přiřazené spojení mezi

daným neuronem a fiktivním neuronem, jehož aktivace je vždy *1*. Z uvedeného obrázku tedy vyplývá, že vícevrstvá neuronová síť je tvořena minimálně třemi vrstvami neuronů: vstupní, výstupní a alespoň jednou vnitřní vrstvou. Vždy mezi dvěma sousedními vrstvami se pak nachází tzv. *úplné propojení neuronů*, tedy každý neuron nižší vrstvy je spojen se všemi neurony vrstvy vyšší.



Obrázek 18: Neuronová síť s jednou vnitřní vrstvou neuronů.

9.2 Standardní metoda backpropagation

Adaptační algoritmus zpětného šíření chyby (*backpropagation*) je používán v přibližně 80% všech aplikací neuronových sítí. Samotný algoritmus obsahuje tři etapy (Šíma, 1996): dopředné (*feedforward*) šíření vstupního signálu tréninkového vzoru, zpětné šíření chyby a aktualizace váhových hodnot na spojeních. Během dopředného šíření signálu obdrží každý neuron ve vstupní vrstvě (X_i, $i = 1,..., n$) vstupní signál (x_i) a zprostředkuje jeho přenos ke všem neuronům vnitřní vrstvy ($Z_1, ..., Z_p$). Každý neuron ve vnitřní vrstvě vypočítá svou aktivaci (z_j) a pošle tento signál všem neuronům ve výstupní vrstvě. Každý neuron ve výstupní vrstvě vypočítá svou aktivaci (y_k), která odpovídá jeho skutečnému výstupu (k. neuronu) po předložení vstupního vzoru. V podstatě tímto způsobem získáme odezvu neuronové sítě na vstupní podnět daný excitací neuronů vstupní vrstvy. Takovým způsobem probíhá šíření signálů i v biologickém systému, kde vstupní vrstva může být tvořena např. zrakovými buňkami a ve výstupní vrstvě mozku jsou pak identifikovány

jednotlivé objekty sledování. Otázkou pak zůstává to nejdůležitější, jakým způsobem jsou stanoveny synaptické váhy vedoucí ke korektní odezvě na vstupní signál. Proces stanovení synaptických vah je opět spjat s pojmem učení (adaptace) neuronové sítě.

Další otázkou je schopnost *generalizace* (zobecnění*)* nad naučeným materiálem, jinými slovy jak je neuronová síť schopna na základě naučeného usuzovat na jevy, které nebyly součástí učení, které však lze nějakým způsobem z naučeného odvodit.

Co je nutné k naučení neuronové sítě? Je to jednak tzv. *trénovací množina* obsahující prvky popisující řešenou problematiku a dále pak metoda, která dokáže tyto vzorky zafixovat v neuronové síti formou hodnot synaptických vah pokud možno včetně již uvedené schopnosti generalizovat. Zastavme se nejdříve u trénovací množiny. Každý vzor trénovací množiny popisuje jakým způsobem jsou excitovány neurony vstupní a výstupní vrstvy. Formálně můžeme za trénovací množinu ***T*** považovat množinu *q* prvků (vzorů), které jsou definovány uspořádanými dvojicemi následujícím způsobem (Fausett, 1994):

$$T = \left\{ \left(\mathbf{x}_k, \mathbf{t}_k \right) \middle| \; \mathbf{x}_k \in \{0,1\}^n, \mathbf{t}_k \in \{0,1\}^m, k = 1, \cdots q \right\}$$

kde q počet vzorů trénovací množiny

$\boldsymbol{x}_k$ vektor excitací vstupní vrstvy tvořené n neurony

$\boldsymbol{t}_k$ vektor excitací výstupní vrstvy tvořené m neurony

Metoda, která umožňuje adaptaci neuronové sítě nad danou trénovací množinou se nazývá *backpropagation*, což v překladu znamená metodu zpětného šíření. Na rozdíl od už popsaného dopředného chodu při šíření signálu neuronové sítě spočívá tato metoda adaptace v opačném šíření informace směrem od vrstev vyšších k vrstvám nižším. Během adaptace neuronové sítě metodou backpropagation jsou srovnávány vypočítané aktivace y_k s definovanými výstupními hodnotami t_k pro každý neuron ve výstupní vrstvě a pro každý tréninkový vzor. Na základě tohoto srovnání je definována chyba neuronové sítě, pro kterou je vypočítán faktor δ_k ($k = 1, ..., m$). δ_k, jež

odpovídá části chyby, která se šíří zpětně z neuronu Y_k ke všem neuronům předcházející vrstvy majícím s tímto neuronem definované spojení. Podobně lze definovat i faktor δ_j (j = *1, ..., p*), který je části chyby šířené zpětně z neuronu Z_j ke všem neuronům vstupní vrstvy, jež mají s tímto neuronem definované spojení. Úprava váhových hodnot w_{jk} na spojeních mezi neurony vnitřní a výstupní vrstvy závisí na faktoru δ_k a aktivacích z_j neuronů Z_j ve vnitřní vrstvě. Úprava váhových hodnot v_{ij} na spojeních mezi neurony vstupní a vnitřní vrstvy závisí na faktoru δ_j a aktivacích x_i neuronů X_i ve vstupní vrstvě. *Aktivační funkce* pro neuronové sítě s adaptační metodou backpropagation musí mít následující vlastnosti: musí být spojitá, diferencovatelná a monotónně neklesající. Nejčastěji používanou aktivační funkcí je proto standardní (logická) sigmoida a hyperbolický tangens.

Chyba sítě $E(\mathbf{w})$ je vzhledem k tréninkové množině definována jako součet parciálních chyb sítě $E_l(\mathbf{w})$ vzhledem k jednotlivým tréninkovým vzorům a závisí na konfiguraci sítě $\mathbf{w}$:

$$E(\mathbf{w}) = \sum_{l=1}^{q} E_l(\mathbf{w}).$$

Parciální chyba $E_l(\mathbf{w})$ sítě pro *l.* tréninkový vzor (l = 1, ...,q) je úměrná součtu mocnin odchylek skutečných hodnot výstupu sítě pro vstup *l.* tréninkového vzoru od požadovaných hodnot výstupů u tohoto vzoru:

$$E_l(\mathbf{w}) = \frac{1}{2}\sum_{k \in Y}(y_k - t_k)^2.$$

Cílem adaptace je minimalizace chyby sítě ve váhovém prostoru. Vzhledem k tomu, že chyba sítě přímo závisí na komplikované nelineární složené funkci vícevrstvé sítě, představuje tento cíl netriviální optimalizační problém. Pro jeho řešení se v základním modelu používá nejjednodušší varianta gradientní metody, která vyžaduje diferencovatelnost chybové funkce.

Hlavním problémem gradientní metody je, že pokud již nalezne lokální minimum, pak toto minimum nemusí být globální. Uvedený postup adaptace se v takovém minimu zastaví (nulový gradient) a chyba sítě se již dále nesnižuje.

To lze v analogii s učením člověka interpretovat tak, že počáteční nastavení konfigurace v okolí nějakého minima chybové funkce určuje možnosti jedince učit se. Inteligentnější lidé začínají svou adaptaci v blízkosti hlubších minim. I zde je však chybová funkce definovaná relativně vzhledem k požadovanému „inteligentnímu“ chování (tréninková množina), které však nemusí být univerzálně platné. Hodnotu člověka nelze měřit žádnou chybovou funkcí. Elektrické šoky aplikované v psychiatrických léčebnách připomínají některé metody adaptace neuronových sítí, které v případě, že se učení zastavilo v mělkém lokálním minimu chybové funkce, náhodně vnáší šum do konfigurace sítě, aby se síť dostala z oblastí abstrakce tohoto lokálního minima a mohla popř. konvergovat k hlubšímu minimu.

Adaptační algoritmus backpropagation (Fausett, 1994):

Krok 0. Váhové hodnoty a bias jsou inicializovány malými náhodnými čísly.

Přiřazení inicializační hodnoty koeficientu učení α.

Krok 1. Dokud není splněna podmínka ukončení výpočtu, opakovat kroky (2 až 9).

Krok 2. Pro každý (bipolární) tréninkový pár **s:t** provádět kroky (3 až 8).

Feedforward:

Krok 3. Aktivovat vstupní neurony (X_i, *i=1, ...n)*

$x_i = s_i$..

Krok 4 Vypočítat vstupní hodnoty vnitřních neuronů:

(Z_j, *j=1,..., p)*:

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}.$$

Stanovení výstupních hodnot vnitřních neuronů

$$z_j = f\left(z\_in_j\right).$$

Krok 5 Stanovení skutečných výstupních hodnoty signálu neuronové sítě (Y_k, *k=1, ..., m)*:

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j \, w_{jk},$$

$$y_k = f\left(y\_in_k\right).$$

Backpropagation:

Krok 6 Ke každému neuronu ve výstupní vrstvě (Y_k, *k=1, ..., m)* je přiřazena hodnota očekávaného výstupu pro vstupní tréninkový vzor. Dále je vypočteno $\delta_k = \left(t_k - y_k\right) f'\left(y\_in_k\right)$, které je součástí váhové korekce $\Delta w_{jk} = \alpha \, \delta_k z_j$ i korekce biasu $\Delta w_{0k} = \alpha \, \delta_k$.

Krok 7 Ke každému neuronu ve vnitřní vrstvě (*Zj, j=1, ..., p)* je přiřazena sumace jeho delta vstupů (tj. z neuronů, které se nacházejí v následující vrstvě), $\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$. Vynásobením získaných hodnot derivací jejich aktivační funkce obdržíme $\delta_j = \delta\_in_j f'\left(z\_in_j\right)$, které je součástí váhové korekce $\Delta v_{ij} = \alpha \, \delta_j x_i$ i korekce biasu $\Delta v_{0j} = \alpha \, \delta_j$.

Aktualizace vah a prahů:

Krok 8 Každý neuron ve výstupní vrstvě (Y_k, *k=1, ..., m)* aktualizuje na svých spojeních váhové hodnoty včetně svého biasu (*j=0, ..., p)*:

$w_{jk}\left(new\right) = w_{jk}\left(old\right) + \Delta w_{jk}$.

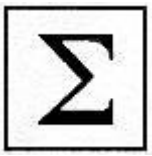
Každý neuron ve vnitřní vrstvě (Z_j, *j=1, ..., p)* aktualizuje na svých spojeních váhové hodnoty včetně svého biasu (*i=0, ..., n)*:

$v_{ij}\left(new\right) = v_{ij}\left(old\right) + \Delta v_{ij}$.

Krok 9. Podmínka ukončení:
pokud již nenastávají žádné změny váhových hodnot
nebo pokud již bylo vykonáno maximálně definované
množství váhových změn, stop; jinak, pokračovat.

Ačkoliv vlastní popis učícího algoritmu backpropagation je formulován pro klasický von neumannovský model počítače, přesto je zřejmé, že jej lze implementovat distribuovaně. Pro každý tréninkový vzor probíhá nejprve aktivní režim pro jeho vstup tak, že informace se v neuronové síti šíří od vstupu k jejímu výstupu. Potom na základě externí informace učitele o požadovaném výstupu, tj. o chybě u jednotlivých vstupů, se počítají parciální derivace chybové funkce tak, že signál se šíří zpět od výstupu ke vstupu. Výpočet sítě při zpětném chodu probíhá sekvenčně po vrstvách, přitom v rámci jedné vrstvy může probíhat paralelně.

Shrnutí kapitoly



V této kapitole jste se podrobně seznámili s topologií vícevrstvé neuronové sítě a s jejím adaptačním algoritmem zpětného šíření chyby (backpropagation), jež je používán v přibližně 80% všech aplikací neuronových (tj. je nejrozšířenějším adaptačním algoritmem vícevrstvých neuronových sítí).

Kontrolní otázky a úkoly:

1. Jaká je topologie vícevrstvé neuronové sítě?
2. Objasněte dopředné (feedforward) šíření signálu.
3. Jak probíhá adaptace metodou backpropagation?

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu backpropagation a použijte jej pro logickou funkci XOR.

Citovaná a doporučená literatura

Šíma, J., Neruda, J. (1996) Teoretické otázky neuronových sítí. Matfyzpress, Praha.

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

10 Kohonenovy samoorganizační mapy

V této kapitole se dozvíte:

- Jaké jsou principy soutěžní strategie učení.
- Jaký je princip procesu shlukování.
- Jak pracují Kohonenovy samoorganizační mapy.

Po jejím prostudování byste měli být schopni:

- vysvětlit principy soutěžní strategie učení,
- objasnit princip procesu shlukování,
- charakterizovat Kohonenovy samoorganizační mapy.

Klíčová slova kapitoly: Kohonenovy samoorganizační mapy, procesu shlukování, soutěžní strategie učení.

Průvodce studiem

V této kapitole se budeme věnovat principům soutěžní strategie učení (competitive learning). Výstupní neurony sítě spolu soutěží o to, který z nich bude aktivní. Na rozdíl od jiných učících principů (např. Hebbovo učení) je tedy v určitém čase aktivní vždy jen jeden neuron. Jedná se o adaptaci bez učitele.

Na zvládnutí této kapitoly budete potřebovat asi 3 hodiny.

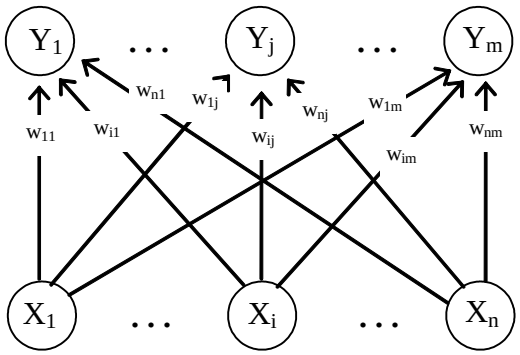
10.1 Topologie SOM

Tato neuronová síť (angl. *Self -Organizing Map*) byla poprvé popsána v roce 1982. Je nejdůležitější architekturou vycházející ze strategie soutěžního učení (tj. *učení bez učitele*). Základním principem učícího procesu je vytvoření množiny reprezentantů mající stejné pravděpodobnosti výběru. Přesněji, hledáme takové reprezentanty, pro které platí: vybereme-li náhodný vstupní vektor z rozdělení pravděpodobnosti odpovídající rozdělení tréninkové množiny, bude mít každý takový reprezentant přiřazenu pravděpodobnost,

která je mu nejblíže. Algoritmus tedy nemá informace o požadovaných aktivitách výstupních neuronů v průběhu adaptace, ale adaptace vah odráží statistické vlastnosti trénovací množiny. Jsou-li si tedy dva libovolné vzory blízké ve vstupním prostoru způsobují v síti odezvu na neuronech, které jsou si fyzicky blízké ve výstupním prostoru. Hlavní ideou těchto neuronových sítí je nalézt prostorovou reprezentaci složitých datových struktur. Mnohodimenzionální data se tímto způsobem zobrazují v daleko jednodušším prostoru. Uvedená vlastnost je typická i pro skutečný mozek, kde například jeden konec sluchové části mozkové kůry reaguje na nízké frekvence, zatímco opačný konec reaguje na frekvence vysoké.

Jedná se o dvouvrstvou síť s úplným propojením neuronů mezi vrstvami. Výstupní neurony jsou navíc uspořádány do nějaké topologické struktury, nejčastěji to bývá dvojrozměrná mřížka nebo jednorozměrná řada jednotek. Tato topologická struktura určuje, které neurony spolu v síti sousedí (pro adaptační proces je to nezbytné). Pro adaptační proces je rovněž důležité zavést pojem *okolí J* výstupního neuronu (*j**) o *poloměru* (velikosti) *R*, což je množina všech neuronů ($j \in J$), jejichž vzdálenost v síti je od daného neuronu (*j**) menší nebo rovna *R*: $J = \{j;\ d(j,j^*) \leq R\}$. To, jak měříme vzdálenost *d(j,j*)*, je závislé na topologické struktuře výstupních neuronů

Obecná architektura Kohonenovy samoorganizační mapy obsahující *m* neuronů ve výstupní vrstvě (tj. $Y_1,\ldots, Y_m$) a *n* neuronů ve vstupní vrstvě (tj. $X_1,\ldots, X_n$) je zobrazena na obrázku 19.



Obrázek 19: Kohonenova samoorganizační mapa.

10.2 Kohonenův algoritmus

Princip adaptivní dynamiky je jednoduchý: Procházíme celou tréninkovou množinu a po předložení jednoho tréninkového vzoru proběhne mezi neurony sítě kompetice. Její vítěz pak spolu s neurony, které jsou v jeho okolí, změní své váhové hodnoty. Reálný parametr učení $0 < \alpha \leq 1$ určuje míru změny vah. Na počátku učení je obvykle blízký jedné a postupně se zmenšuje až na nulovou hodnotu, což zabezpečuje ukončení procesu adaptace. Rovněž i velikost okolí R není konstantní: na začátku adaptace je okolí obvykle velké (např. polovina velikosti sítě) a na konci učení potom zahrnuje jen jeden samotný vítězný neuron (tj. $R = 0$).

Kohonenův algoritmus (Fausett, 1994):

Krok 0. Inicializace všech váhových hodnot w_{ij}:
Inicializace poloměru sousedství; tj okolí (R).
Inicializace parametru učení (α).

Krok 1. Pokud není splněna podmínka ukončení, provádět kroky (2-8).

Krok 2. Pro každý vstupní vektor $\boldsymbol{x} = (x_1,...,x_n)$ opakovat kroky 3 až 5.

Krok 3. Pro každé j $(j = 1,..., m)$ vypočítat:

$$D(j) = \sum_i \left(w_{ij} - x_i\right)^2.$$

Krok 4. Najít index J takový, že $D(J)$ je minimum.

Krok 5. Aktualizace váhových hodnot všech neuronů $(j \in J)$ tvořících topologické sousedství charakterizované indexem J, tj. pro všechna i $(i = 1,..., n)$ platí:

$$w_{ij}(new) = w_{ij}(old) + \alpha\left[x_i - w_{ij}(old)\right].$$

Krok 6. Aktualizace parametru učení.

Krok 7. Zmenšení poloměru R topologického sousedství.

Krok 8. Test podmínky ukončení.

Geometrický význam popsaného algoritmu je takový, že vítězný neuron *i* a všichni jeho sousedé v síti, kteří by od něj neměli být příliš vzdáleni ani ve vstupním prostoru, posunou svůj váhový vektor o určitou poměrnou vzdálenost směrem k aktuálnímu vstupu. Motivací tohoto přístupu je snaha, aby vítězný neuron, který nejlépe reprezentuje předložený vstup (je mu nejblíže), ještě více zlepšil svou relativní pozici vůči němu. Problémem vzniklým při adaptaci může být nevhodná náhodná inicializace vah, která vede k blízkým počátečním neuronům ve výstupní vrstvě a tudíž pouze jeden z nich vyhrává kompetici zatímco ostatní zůstávají nevyužity.

V ***aktivním režimu*** se pak sousedství neuronů neprojevuje: předložíme-li síti vstupní vektor, soutěží výstupní neurony o to, kdo je mu nejblíže a tento neuron se pak excituje na hodnotu rovnu jedné, zatímco výstupy ostatních neuronů jsou rovny nule. Každý neuron tak reprezentuje nějaký objekt, či třídu objektů ze vstupního prostoru: tj. pouze jeden neuron horní vrstvy, jehož potenciál $(\Sigma w.x)$ je maximální odpovídá vstupnímu vektoru $\boldsymbol{x}$. Tento neuron je navíc schopen rozpoznat celou třídu takových, podobných si vektorů. Princip „vítěz bere vše“ se realizuje tzv. *laterální inhibicí*; všechny výstupní neurony jsou navzájem propojeny laterálními vazbami, které mezi nimi přenášejí inhibiční signály. Každý výstupní neuron se pak snaží v kompetici zeslabit ostatní neurony silou úměrnou jeho potenciálu, který je tím větší, čím je neuron blíže vstupu. Výsledkem tedy je, že výstupní neuron s největším potenciálem utlumí ostatní výstupní neurony a sám zůstane aktivním.

Příklad:

Mějme 4 vektory: (1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, ,0); (0, 0, 1, 1).

Maximální počet shluků je: $m = 2$.

Předpokládejme, že parametr učení je definován vztahy: $\alpha(0) = 0.6$;

$$\alpha(t+1) = 0.5\ \alpha(t).$$

Protože jsou k dispozici pouze dva shluky, okolí bodu J (*krok 4*) je nastaveno tak, že v každém kroku aktualizuje své váhové hodnoty pouze jeden neuron výstupní vrstvy, tj. $R = 0$.

Krok 0. Inicializace váhové matice:

$$\begin{bmatrix} 0{,}2 & 0{,}8 \\ 0{,}6 & 0{,}4 \\ 0{,}5 & 0{,}7 \\ 0{,}9 & 0{,}3 \end{bmatrix}.$$

Inicializace poloměru sousedství:

$R = 0$.

Inicializace parametru učení:

$\alpha(0) = 0{,}6$.

Krok 1. Adaptace:

Krok 2. Pro první vektor (1, 1, 0, 0) opakovat kroky 3-5.

Krok 3. $D(1) = (0{,}2-1)^2 + (0{,}6-1)^2 + (0{,}5-0)^2 + (0{,}9-0)^2 = 1{,}86;$

$$D(2) = (0{,}8-1)^2 + (0{,}4-1)^2 + (0{,}7-0)^2 + (0{,}3-0)^2 = 0{,}98.$$

Krok 4. Vstupní vektor je blíže uzlu 2, tak $J = 2$.

Krok 5. Aktualizace váhových hodnot vítězného neuronu:

$$\begin{aligned} w_{i2}(new) &= w_{i2}(old) + 0{,}6\left[x_i - w_{i2}(old)\right] \\ &= 0{,}4 w_{i2}(old) + 0{,}6 x_i. \end{aligned}$$

Aktualizace druhého sloupce váhové matice

$$\begin{bmatrix} 0{,}2 & 0{,}92 \\ 0{,}6 & 0{,}76 \\ 0{,}5 & 0{,}28 \\ 0{,}9 & 0{,}12 \end{bmatrix}.$$

Krok 2. Pro druhý vektor (*0, 0, 0, 1*) opakovat kroky 3-5.

Krok 3.

$$D(1) = (0{,}2-0)^2 + (0{,}6-0)^2 + (0{,}5-0)^2 + (0{,}9-1)^2 = 0{,}66;$$

$$D(2) = (0{,}92-0)^2 + (0{,}76-0)^2 + (0{,}28-0)^2 + (0{,}12-1)^2 = 2{,}27.$$

Krok 4. Vstupní vektor je blíže uzlu 1, tak $J = 1$.

Krok 5. Aktualizace prvního sloupce váhové matice

$$\begin{bmatrix} 0{,}08 & 0{,}92 \\ 0{,}24 & 0{,}76 \\ 0{,}20 & 0{,}28 \\ 0{,}96 & 0{,}12 \end{bmatrix}.$$

Krok 2. Pro třetí vektor (*1, 0, 0, 0*) opakovat kroky 3-5.

Krok 3. $D(1) = (0{,}08-1)^2 + (0{,}24-0)^2 + (0{,}2-0)^2 + (0{,}96-0)^2 = 1{,}86;$

$D(2) = (0{,}92-1)^2 + (0{,}76-0)^2 + (0{,}28-0)^2 + (0{,}12-0)^2 = 0{,}67$

Krok 4. Vstupní vektor je blíže uzlu 2, tak $J = 2$.

Krok 5. Aktualizace druhého sloupce váhové matice

$$\begin{bmatrix} 0{,}08 & 0{,}968 \\ 0{,}24 & 0{,}304 \\ 0{,}20 & 0{,}112 \\ 0{,}96 & 0{,}048 \end{bmatrix}.$$

Krok 2. Pro čtvrtý vektor (*0, 0, 1, 1*) opakovat kroky 3-5.

Krok3. $D(1) = (0{,}08-0)^2 + (0{,}24-0)^2 + (0{,}2-1)^2 + (0{,}96-1)^2 = 0{,}705;$

$D(2) = (0{,}968-0)^2 + (0{,}304-0)^2 + (0{,}112-1)^2 + (0{,}048-1)^2 = 2.72$

Krok 4. Vstupní vektor je blíže uzlu 1, tak $J = 1$.

Krok 5. Aktualizace prvního sloupce váhové matice

$$\begin{bmatrix} 0{,}032 & 0{,}968 \\ 0{,}096 & 0{,}304 \\ 0{,}680 & 0{,}112 \\ 0{,}984 & 0{,}048 \end{bmatrix}.$$

Krok 6. Zmenšení parametru učení:

$\alpha = 0{,}5\ (0{,}6) = 0{,}3.$

Aktualizace váhových hodnot vítězného neuronu *j* (*j = 1, 2*) ve druhém cyklu bude prováděna podle vztahu:

$$w_{ij}(new) = w_{ij}(old) + 0.3\left[x_i - w_{ij}(old)\right]$$
$$= 0{,}7w_{ij}(old) + 0{,}3x_i.$$

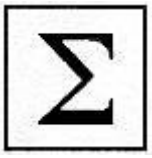
Parametr učení zmenšil svou hodnotu během 0.6 na 0.01 a výsledná váhová matice nabývala konverguje k matici:

$$\begin{bmatrix} 0{,}0 & 1{,}0 \\ 0{,}0 & 0{,}5 \\ 0{,}5 & 0{,}0 \\ 1{,}0 & 0{,}0 \end{bmatrix}.$$

Její první sloupec nabývá hodnot, které odpovídají průměrným hodnotám složek obou vektorů přiřazeným prvnímu neuronu výstupní vrstvy, tj. vektoru 2: (0, 0, 0, 1) a vektoru 4: (0, 0, 1, 1)). Její druhý sloupec nabývá hodnot, které odpovídají průměrným hodnotám složek obou vektorů přiřazeným druhému neuronu výstupní vrstvy, tj. vektoru 1: (1, 1, 0, 0) a vektoru 3: (1, 0, 0, ,0).

Proces shlukování ještě jednou vysvětlíme prostřednictvím funkce hustoty pravděpodobnosti. Tato funkce reprezentuje statistický nástroj popisující rozložení dat v prostoru. Pro daný bod prostoru lze tedy stanovit pravděpodobnost, že vektor bude v daném bodu nalezen. Je-li dán vstupní prostor a funkce hustoty pravděpodobnosti, pak je možné dosáhnout takové organizace mapy, která se této funkci přibližuje (za předpokladu, že je k dispozici reprezentativní vzorek dat). Jinými slovy řečeno, pokud jsou vzory ve vstupním prostoru rozloženy podle nějaké distribuční funkce, budou váhové vektory rozloženy analogicky.

Shrnutí kapitoly



V této kapitole jsme se věnovali principům soutěžní strategie učení (competitive learning). Výstupní neurony sítě zde spolu soutěží o to, který z

nich bude aktivní. Na rozdíl od jiných učících principů (např. Hebbovo učení) je tedy v určitém čase aktivní vždy jen jeden neuron. Jedná se o adaptaci bez učitele.

Kontrolní otázky a úkoly:

1. Jaké jsou principy soutěžní strategie učení?
2. Jaký je princip procesu shlukování?
3. Jak pracují Kohonenovy samoorganizační mapy?

Úkoly k textu

Mějme pět vektorů: (1,1,0,0), (0,0,0,1), (0,0,1,1), (1,0,0,0), (0,1,1,0). Maximální počet shluků je: m=2. Řešte příklad a) algoritmem adaptace Kohonenovy samoorganizační mapy (vhodně si definujte vztah pro parametr učení).

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu pracujícího na principu soutěžní strategie učení.

Citovaná a doporučená literatura

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

11 Diskrétní Hopfieldova síť

V této kapitole se dozvíte:

- Jaký je princip adaptace diskrétní Hopfieldovy sítě.
- Jaká je topologie diskrétní Hopfieldovy sítě.

Po jejím prostudování byste měli být schopni:

- vysvětlit princip adaptace diskrétní Hopfieldovy sítě,
- objasnit energetickou funkci Hopfieldovy sítě,
- charakterizovat topologii diskrétní Hopfieldovy sítě.

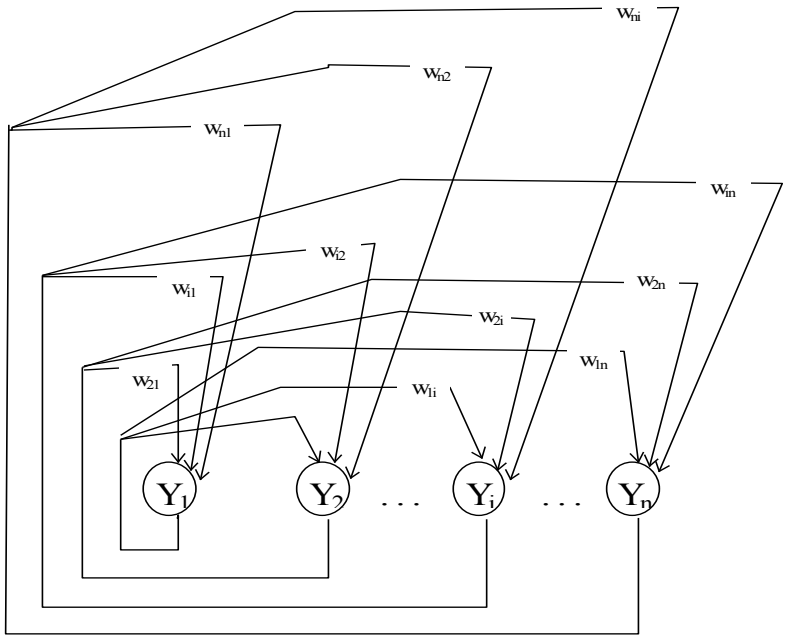
Klíčová slova kapitoly: 11 Diskrétní Hopfieldova síť, energetická funkce.

Průvodce studiem

V této kapitole se seznámíte s model diskrétní Hopfieldovy sítě, zejména s její topologií a principem adaptace. V závěru je pak vysvětlena energetická funkce Hopfieldovy sítě.

Na zvládnutí této kapitoly budete potřebovat asi 3 hodiny.

Model Hopfieldovy neuronové sítě je založen na využití energetické funkce svázané s neuronovou sítí tak, jak je to běžné u fyzikálních systémů. Organizační dynamika diskrétní Hopfieldovy sítě specifikuje úplnou topologii cyklické neuronové sítě s n neurony, kde každý neuron v síti je spojen se všemi ostatními neurony sítě, tj. má všechny neurony za své vstupy. Obecně platí, že může být spojen i sám se sebou. Všechny neurony v síti jsou tedy zároveň vstupní i výstupní. Architektura Hopfieldovy sítě je znázorněna na obrázku 20. Každý spoj v síti mezi neuronem i ($i = 1, ..., n$) a neuronem j ($j = 1, ..., n$) je ohodnocen celočíselnými synaptickými vahami w_{ij} a w_{ji}, které jsou symetrické, tj. $w_{ij} = w_{ji}$. V základním modelu platí, že žádný neuron není spojen sám se sebou, tj. odpovídající váhy $w_{jj} = 0$ ($j = 1, ..., n$) jsou nulové.



Obrázek 26: Model diskrétní Hopfieldovy sítě.

Hlavní myšlenka adaptace Hopfieldova modelu spočívá v tom, že jsou nejprve inicializovány všechny neurony sítě buď binárními hodnotami {0, 1} nebo bipolárními hodnotami {-1,+1}. Vzhledem k tomu, že jsou všechny neurony navzájem propojeny, začnou se ovlivňovat. To znamená, že jeden neuron se snaží ostatní neurony excitovat na rozdíl od jiného, který se snaží o opačné. Probíhá cyklus postupných změn excitací neuronů až do okamžiku nalezení kompromisu - síť relaxovala do stabilního stavu. Jinými slovy výstupy předchozího kroku se staly novými vstupy současného kroku. Tento proces je vysvětlitelný následujícím algoritmem: tréninkové vzory nejsou v Hopfieldově síti uloženy přímo, ale jsou reprezentovány pomocí vztahů mezi stavy neuronů.

První popis adaptačního algoritmu Hopfieldovy sítě pochází z roku 1982 a používá binární hodnoty pro excitace neuronů. Požadovaná funkce sítě je specifikována tréninkovou množinou *P* vzorů $\mathbf{s}(p)$, $p = 1, ..., P$, z nichž každý je zadán vektorem n binárních stavů vstupních resp. výstupních neuronů, které v případě autoasociativní paměti splývají:

$$\mathbf{s}(p) = \left(s_1(p), \ldots, s_i(p), \ldots, s_n(p)\right),$$

potom je váhová matice $\mathbf{W} = \{w_{ij}\}$ dána následujícím vztahem:

$$w_{ij} = \sum_p \left[2s_i(p) - 1\right]\left[2s_j(p) - 1\right] \qquad \text{pro } i \neq j$$

a $w_{i\,i} = 0.$

Jiný popis adaptačního algoritmu Hopfieldovy sítě pochází z roku 1984 a pracuje s bipolárními hodnotami pro excitace neuronů. Požadovaná funkce sítě je rovněž specifikována tréninkovou množinou *P* vzorů **s***(p), p = 1, ..., P,* z nichž každý je zadán vektorem *n* bipolárních stavů vstupních resp. výstupních neuronů, které v případě autoasociativní paměti splývají:

$$\mathbf{s}(p) = \left(s_1(p), \ldots, s_i(p), \ldots, s_n(p)\right),$$

potom je váhová matice **W** = $\{w_{ij}\}$ dána následujícím vztahem:

$$w_{ij} = \sum_p s_i(p)\, s_j(p) \qquad \text{pro } i \neq j$$

a $w_{i\,i} = 0.$

Adaptační algoritmus Hopfieldovy sítě (Fausett, 1994):

Krok 0. Inicializace vah, tj. zapamatování vzorů.
Použitím Hebbova adaptačního pravidla
Dokud síť nezrelaxovala do stabilního stavu, opakovat kroky (1 až 7).

Krok 1. Pro každý vstupní vektor ***x***, opakovat kroky (2 až 6).

Krok 2. Inicializace sítě vnějším vstupním vektorem ***x***:

$$y_i = x_i, \quad (i = 1, \ldots, n)$$

Krok 3. Pro každý neuron Y_i opakovat kroky (4 až 6).
(Neurony jsou uspořádány náhodně)

Krok 4 Vypočítat vnitřní potenciál neuronu:

$$y\_in_i = x_i + \sum_j y_j w_{ji}.$$

Krok 5 Stanovení výstupu neuronu lze chápat jako aplikaci aktivační funkce:

$$y_i = \begin{cases} 1 & \text{pokud } y\_in_i > \theta_i \\ y_i & \text{pokud } y\_in_i = \theta_i \\ 0 & \text{pokud } y\_in_i < \theta_i . \end{cases}$$

Krok 6 Transport hodnoty y_i ostatním neuronům. (Takto budeme aktualizovat hodnoty aktivačního vektoru.)

Krok 7. Test konvergence.

Prahová hodnota θ_i je obvykle nulová. Aktualizace neuronů probíhají sice v náhodném pořadí, ale musí být prováděny stejnou průměrnou rychlostí.

Příklad:

Pomocí diskrétního Hopfieldova modelu určete, zda je vstupní vektor „naučeným“ vzorem (tj. byl součástí trénovací množiny).

Krok 0. Váhová matice pro zapamatování vektoru (1, 1, 1, 0) (přepis vektoru do bipolární reprezentace (1, 1, 1, -1)) má tvar:

$$\mathbf{W} = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

Krok 1. Vstupní vektor je $\boldsymbol{x} = (0, 0, 1, 0)$.
Pro tento vektor opakovat kroky (2 až 6).

Krok 2. $\boldsymbol{y} = (0, 0, 1, 0)$.

Krok 3. Vybrat Y_1 a aktualizovat jeho aktivaci:

Krok 4 $y\_in_1 = x_1 + \sum_j y_j w_{j1} = 0 + 1.$

Krok 5 $y\_in_1 > 0 \rightarrow y_1 = 1.$

Krok 6 $\boldsymbol{y} = (1, 0, 1, 0)$.

Krok 3. Vybrat Y_4 a aktualizovat jeho aktivaci:

Krok 4 $y\_in_4 = x_4 + \sum_j y_j w_{j4} = 0 + (-2)$.

Krok 5 $y\_in_1 < 0 \rightarrow y_4 = 0$.

Krok 6 $y = (1, 0, 1, 0)$.

Krok 3. Vybrat Y_3 a aktualizovat jeho aktivaci:

Krok 4 $y\_in_3 = x_3 + \sum_j y_j w_{j3} = 1 + 1$.

Krok 5 $y\_in_3 > 0 \rightarrow y_3 = 1$.

Krok 6 $\boldsymbol{y} = (1, 0, 1, 0)$.

Krok 3. Vybrat Y_2 a aktualizovat jeho aktivaci:

Krok 4 $y\_in_2 = x_2 + \sum_j y_j w_{j2} = 0 + 2$.

Krok 5 $y\_in_2 > 0 \rightarrow y_2 = 1$.

Krok 6 $\boldsymbol{y} = (1, 1, 1, 0)$.

Krok 7. Test konvergence.

Aktivace každého neuronu byla aktualizována alespoň jednou během celého výpočtu. *Vstupní vektor konverguje k uloženému vzoru*.

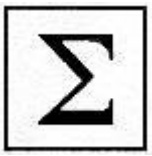
K lepšímu pochopení aktivní dynamiky Hopfieldovy sítě byla Hopfieldem, v analogii s fyzikálními ději definována tzv. *energetická funkce E* sítě, která každému stavu sítě přiřazuje jeho potenciální energii. Energetická funkce je tedy funkce, která je zdola ohraničená a pro daný stav systému je nerostoucí. V teorii neuronových sítí se *stavem systému* rozumí množina aktivací všech neuronů. Pokud je již tato energetická funkce nalezena, bude síť konvergovat ke stabilní množině aktivací neuronů v daném časovém okamžiku. Energetická funkce pro diskrétní Hopfieldovu síť je dána následovně:

$$E = -0{,}5\sum_{i\neq j}\sum_{j} y_i y_j w_{ij} - \sum_i x_i y_i + \sum_i \theta_i y_i .$$

Z definice energetické funkce vyplývá, že stavy sítě s nízkou energií mají největší stabilitu.

Hopfieldova síť má ve srovnání s vícevrstvou sítí adaptovanou učícím algoritmem backpropagation opačný charakter aktivní a adaptivní dynamiky. Zatímco adaptace Hopfieldovy sítě podle Hebbova zákona je jednorázovou záležitostí, jejíž trvání závisí jen na počtu tréninkových vzorů, učící algoritmus backpropagation realizuje iterativní proces minimalizující chybu sítě gradientní metodou bez záruky konvergence. Na druhou stranu délka trvání aktivní fáze vícevrstvé sítě je dána pouze počtem vrstev, zatímco aktivní režim Hopfieldovy sítě představuje iterativní proces minimalizující energii sítě diskrétní variantou gradientní metody s nejistou konvergencí. Cílem adaptace Hopfieldovy sítě podle Hebbova zákona je nalezení takové konfigurace, aby funkce sítě v aktivním režimu realizovala autoasociativní paměť. To znamená: bude-li vstup sítě blízký nějakému tréninkovému vzoru, výstup sítě by měl potom odpovídat tomuto vzoru. Z hlediska energie by každý tréninkový vzor měl být lokálním minimem energetické funkce, tj. stabilním stavem sítě. V jeho blízkém okolí, v tzv. *oblasti atrakce*, se nachází všechny vstupy blízké tomuto vzoru. Ty představují počáteční stavy sítě, ze kterých se při minimalizaci energetické funkce v aktivním režimu síť dostane do příslušného minima, tj. stabilního stavu odpovídajícího uvažovanému tréninkovému vzoru. Geometricky se tedy energetická plocha rozpadá na oblasti atrakce lokálních minim a příslušná funkce Hopfieldovy sítě přiřadí v aktivním režimu ke každému vstupu náležejícímu do oblasti atrakce nějakého lokálního minima právě toto minimum. Při učení Hopfieldovy sítě podle Hebbova zákona pro asociativní sítě samovolně vznikají na energetické ploše lokální minima, tzv. *nepravé vzory* (*fantomy*), které neodpovídají žádným tréninkovým vzorům. Výstup sítě pro vstup dostatečně blízký takovému fantomu neodpovídá žádnému vzoru, a tudíž nedává žádný smysl. Existují varianty adaptivní dynamiky Hopfieldovy sítě, při nichž se takto vzniklé fantomy mohou dodatečně odučit.

Shrnutí kapitoly



V této kapitole jste se seznámili s model diskrétní Hopfieldovy sítě, zejména s její topologií a principem adaptace. V závěru pak byla vysvětlena energetická funkce Hopfieldovy sítě.

Kontrolní otázky a úkoly:

1. Jaký je princip adaptace diskrétní Hopfieldovy sítě?
2. Jaká je topologie diskrétní Hopfieldovy sítě?
3. Objasněte energetickou funkci Hopfieldovy sítě.

Korespondenční úkoly

Vytvořte počítačový program pro realizaci adaptačního algoritmu diskrétní Hopfieldovy sítě.

Citovaná a doporučená literatura

Fausett, L. V. (1994) Fundamentals of Neural Networks. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

12 Aplikace neuronových sítí

V této kapitole se dozvíte:

- Jaké jsou hlavní aplikační oblasti umělých neuronových sítí.

Po jejím prostudování byste měli být schopni:

- objasnit použití umělých neuronových sítí v různých aplikačních oblastech,

Klíčová slova kapitoly: Predikace, analýza signálu, komprese dat, expertní systémy.

Průvodce studiem

V této závěrečné kapitole se postupně zamyslíme základními aplikacemi neuronových sítí. Nebudeme zde uvádět hlavní aplikační oblast umělých neuronových sítí, což je rozpoznávání vzorů a klasifikace, protože této problematice již byla věnována předchozí kapitola.

Na zvládnutí této kapitoly budete potřebovat asi 2 hodiny.

Text kapitoly je převážně převzat z (Šíma, 1996). Neuronové sítě v současnosti patří mezi významnou část počítačově orientované umělé inteligence, kde zaujali postavení univerzálního matematicko-informatického přístupu ke studiu a modelování procesů učení. Kromě umělé inteligence mají neuronové sítě nezastupitelné uplatnění také v kognitivní vědě, lingvistice, neurovědě, řízení procesů, přírodních a společenských vědách, kde se pomocí nich modelují nejen procesy učení a adaptace, ale i široké spektrum různých problémů klasifikace objektů a také problémů řízení složitých průmyslových systémů. Původním cílem výzkumu neuronových sítí byla snaha pochopit a modelovat způsob, jakým myslíme a způsob, jak funguje lidský mozek. Při vytváření modelů umělých neuronových sítí nám nejde o vytvoření identických kopií lidského mozku, ale napodobujeme zde pouze některé jeho základní funkce. Neurofyziologie zde slouží jen jako zdroj inspirací a navržené modely umělých

neuronových sítí jsou dále rozvíjeny bez ohledu na to, zda modelují lidský mozek, či nikoliv. Nejvýznamnější oblasti použití umělých neuronových sítí kromě úloh z oblasti klasifikace a rozpoznávání vzorů jsou následující.

Oblastí aplikace neuronových sítí je *řízení* složitých zařízení v dynamicky se měnících podmínkách. V minulé kapitole jsme uvedli dva motivační příklady z této oblasti: balancování koštěte a regulace přítoku látek ve složitém výrobním procesu. Dalším demonstračním příkladem řídícího systému popsaného v literatuře je autopilot automobilu, který se v počítačové simulaci pohybuje na dvouproudé dálnici spolu s auty jedoucími stejným směrem. Auto řízené neuronovou sítí určovalo na základě vzdálenosti a rychlosti nejbližších aut v obou pruzích svou vlastní rychlost a změnu pruhu. Dále neuronová síť ovládala volant podle zakřivení dálnice, polohy auta v pruhu a aktuálního úhlu volantu. Je zajímavé, že neuronová síť se kromě úspěšného řízení vozidla (bez kolizí) včetně předjíždění naučila i různé zvyky a styl jízdy (např. riskantní rychlá jízda a časté předjíždění nebo naopak opatrná pomalá jízda) podle řidičů - trenérů, od kterých byly získány tréninkové vzory.

Jinou důležitou aplikační oblastí neuronových sítí je *predikce* a příp. následné *rozhodování*. Typickými příklady z této oblasti jsou předpověď počasí, vývoj cen akcií na burze, spotřeba elektrické energie apod. Např. při meteorologické předpovědi jsou vstupem neuronové sítě odečty základních parametrů (např. teplota, tlak apod.) v čase a učitelem je skutečný vývoj počasí v následujícím období. Uvádí se, že u předpovědi počasí v rozpětí několika dnů byla síť úspěšnější než meteorologové.

Jiným příkladem uplatnění neuronových sítí je *analýza signálů* jako např. EKG, EEG apod. Spojitý signál je vzorkován ve stejných časových intervalech a několik posledních diskrétních hodnot úrovně signálu slouží jako vstup do např. dvouvrstvé neuronové sítě. Naučená neuronová síť je schopna identifikovat specifický tvar signálu, který je důležitý pro diagnostiku. Např.

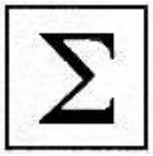
neuronová síť s topologií 40 - 17 - 1 byla použita pro klasifikaci EEG signálů se specifickými α-rytmy.

Další oblastí aplikace neuronových sítí je *transformace signálů*, jehož příkladem je systém NETtalk, určený pro převod anglicky psaného textu na mluvený signál. Tento systém je založen na neuronové síti s topologií 203 - 80 - 26 s 7×29 vstupními neurony pro zakódování kontextu 7 písmen psaného textu odpovídá jeden neuron, který je při jejich výskytu aktivní, 80 skrytými neurony v mezilehlé vrstvě 26 výstupními neurony reprezentují fonény odpovídajícího mluveného signálu. Funkce sítě je následující: vstupní text se postupně přesouvá u vstupních neuronů po jednom písmenu zprava doleva a přitom je aktivní právě ten výstupní neuron, který reprezentuje fonén odpovídající prostřednímu ze 7 písmen vstupního textu. V našem příkladě se čte prostřední písmeno „C“ v anglickém slově „CONCEPT“ s výslovností [´konsept], kterému odpovídá fonén [s]. Stejné písmeno „C“ na začátku tohoto slova však v daném kontextu odpovídá fonénu [k]. Úspěšná implementace systému NETtalk vedla ke snaze vytvořit systém založený na neuronové síti s obrácenou funkcí, která by převáděla mluvený jazyk do psané formy (tzv. fonetický psací stroj).

Další možností využití neuronových sítí je *komprese dat* např. pro přenos televizního signálu, telekomunikaci apod. Pro tento účel byla vyvinuta technika použití neuronové sítě se dvěma vnitřními vrstvami a s topologií *n - n/4 - n/4 - n* (tj. *n* neuronů ve vstupní vrstvě, *n/4* neuronů ve vnitřních vrstvách a n neuronů ve výstupní vrstvě). Počet neuronů ve vnitřních vrstvách je výrazně menší než je počet neuronů ve vstupní a výstupní vrstvě. Počet neuronů ve vstupní i výstupní vrstvě je stejný, protože obě vrstvy reprezentují stejný obrazový signál. Tato neuronová síť se učí různé obrazové vzory tak, že vstup i výstup tréninkových vzorů představují totožný obraz. Síť tak pro daný obrazový vstup odpovídá přibližně stejným výstupem. Při vlastním přenosu je pro daný obrazový signál u vysílače nejprve vypočten stav skrytých neuronů a takto komprimovaný obraz je přenášen informačním kanálem k příjemci, který

jej dekóduje výpočtem stavů výstupních neuronů. Tímto způsobem je získán téměř původní obraz. Při vlastním experimentu se ukázalo, že kvalita přenosu (srovnatelná s jinými způsoby komprese dat) závisí na tom, zda jsou přenášené obrazy podobné tréninkovým vzorům, na které se síť adaptovala.

Posledním oborem aplikace neuronových sítí, který zde uvedeme, jsou *expertní systémy.* Velkým problémem klasických expertních systémů založených na pravidlech je vytvoření báze znalostí, která bývá časově velmi náročnou záležitostí s nejistým výsledkem. Neuronové sítě představují alternativní řešení, kde reprezentace znalostí v bázi vzniká učením z příkladových inferencí. V tomto případě aktivní režim neuronové sítě zastupuje funkci inferenčního stroje. Na druhou stranu implicitní reprezentace znalostí neumožňuje pracovat s neúplnou informací a neposkytuje zdůvodnění závěrů, což jsou vlastnosti, bez kterých se prakticky použitelný expertní systém neobejde. Tento problém částečně řeší univerzální neuronový expertní systém EXPSYS, který obohacuje vícevrstvou neuronovou síť o intervalovou aritmetiku pro práci s nepřesnou informací a o heuristiku analyzující síť, která umožňuje jednoduché vysvětlení závěrů. Systém EXPSYS byl úspěšně aplikován v energetice a medicíně. Např. v lékařské aplikaci jsou zakódované příznaky onemocnění a výsledky různých vyšetření vstupem do neuronové sítě a diagnózy, popř. doporučená léčba jsou jejím výstupem. Tréninkovou množinu lze získat z kartotéky pacientů.



Shrnutí kapitoly

V této závěrečné kapitole jste se postupně seznámili se základními aplikacemi neuronových sítí. Nebyla zde uvedene hlavní aplikační oblast umělých neuronových sítí, což je rozpoznávání vzorů a klasifikace, protože této problematice již byla věnována předchozí kapitola.

Kontrolní otázky a úkoly:

Jaké jsou hlavní aplikační oblasti umělých neuronových sítí?

Korespondenční úkoly

Vypracujte seminární práci na téma „*Použití neuronových sítí v …“* (*oblast použití si zvolte sami*). Informace hledejte především na www-stránkách. Zaměřte se pouze na jednu oblast, v níž použití neuronových sítí rozeberete podrobněji. Nedělejte přehled použití neuronových sítí v různých oblastech.

Citovaná a doporučená literatura

Šíma, J., Neruda, J. (1996) Teoretické otázky neuronových sítí. Matfyzpress, Praha.