Obsah

1 Neuronové sítě - jednotlivý neuron

1.1 Základní informace

Následující text je součástí učebních textů předmětu Umělá inteligence a je určen hlavně pro studenty Matematické biologie. Tuto kapitolu lze považovat za úvodní kapitolu k bloku věnujícímu se umělým neuronovým sítím. Zavádí základní pojmy, definuje matematický model neuronu, vysvětluje jeho klasifikační možnosti. Pochopení matematického modelu jednotlivého neuronu je nutným východiskem pro porozumění konceptu složitějších struktur využívajících propojených množin jednotlivých neuronů - umělých neuronových sít.

1.2 Výstupy z učení

Zvládnutí učebního textu umožní studentům:

- Seznámit se a porozumět analogii mezi umělými neuronovými sítěmi a jejich biologickou motivací, porozumí pojmům z topologie NN - neuron, vrstva, spoj
- Parametricky definovat matematický model jednotlivého neuronu a definovat význam jednotlivých parametrů a typů přenosových funkcí neuronu
- Porozumět aktivní dynamice neuronu a její geometrické interpretaci, matematicky ji formalizovat
- Osvojit si základní pojmy a algoritmy adaptační dynamiky jednotlivého neuronu
    - Hebbovo pravidlo
    - Delta pravidlo
    - Učení dle Widrowa
    - Trénovací a testovací množiny
- Dát si do vztahu matematické logické funkce AND, OR, NOT a XOR s klasifikačními schopnostmi jednotlivého neuronu
- Budou schopni pomocí nastavení parametrů neuronu implementovat modelový návrh logických funkcí AND, OR, NOT
- Pochopit klasifikační schopnosti jednotlivého neuronu se dvěma vstupy a binárním výstupem a jeho geometrickou interpretaci v rovině

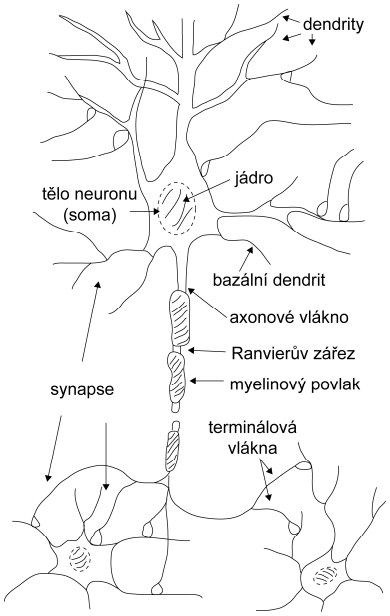
1.3 Úvod do neuronových sítí

1.3.1 Biologická analogie

Koncept umělých neuronových sítí byl v minulosti inspirován a vytvářen na základě biologické analogie s nervovým systémem. Nervový systém můžeme rozdělit v nejjednodušší podobě na centrální nervový systém, který představuje mozek a míchu a periferní nervový systém představovaný zejména periferními nervy. Hlavním posláním CNS je řídit organismus, zpracovávat signály (vzruchy), které se do CNS šíří ze smyslových receptorů dostředivými drahami a vydat na základě vyhodnocení tohoto signálu pokyn, který se opět šíří ve formě signálu odstředivými neuronovými drahami směrem k efektorům, například svalům. V této aktivní dynamice lze zde tedy vypozorovat existenci vstupní informace, její zpracování procesní jednotkou CNS a následné generování výstupu CNS, který je dále využit efektorem, například pro pohyb ruky při hodu míčem na

cíl. Co se stane, pokud cíl při hodu mineme? Máme možnost vzniklou situaci pomocí receptorů (např. visuelně) znova vyhodnotit a v případě, že cíl například přehodíme, pokus korigovat a opakovat jej s menší intenzitou hodu. Zjistili jsme tedy odchylku od požadovaného ideálního stavu na výstupu a na základě zpětné vazby trénujeme náš nervový systém tak, aby byl schopen danou úlohu plnit co nejlépe. Člověk tedy získává prostřednictvím svého nervového systému i schopnost učit se a zapamatovat si získané informace.

Celý nervový systém člověka je představován obrovským počtem navzájem provázaných biologických funkčních jednotek, neuronů. Například mozek obsahuje až 10^{11} neuronů, kde každý neuron může mít až 5000 spojů s ostatními neurony. Tyto spoje, synapse, mohou mít jak excitační (budící), tak inhibiční (tlumící) charakter, mohou tedy vést k posílení, či potlačení odezvy neuronu na přicházející podněty, vzruchy.



Obr. 1 *Biologický neuron (podle [3]).*

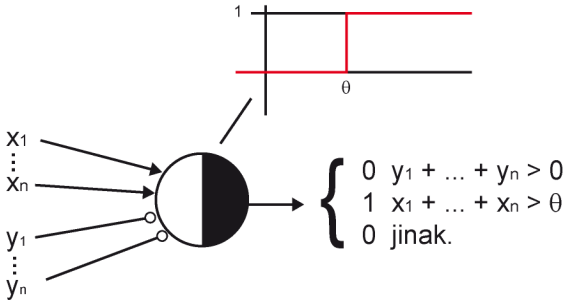
Pokud se opět přidržíme zjednodušujícího popisu biologického neuronu, který nás má dovést k jeho matematickému modelu, můžeme konstatovat, že každý biologický neuron se skládá z těla a dvou typů výběžků, dendritů a axonu. Dendrity představují z pohledu šíření vzruchu vstupy do těla neuronu, kterými přichází signály (vzruchy) z okolních neuronů. Jedná se o plastické spoje, které mají schopnost větvení, prodlužování, zmnožení. Předpokládá se, že plasticita dendritů představuje podklad dlouhodobé paměti. Axon představuje výstup neuronu, kterým se šíří vzruch z těla neuronu a který je pomocí synapsí napojen na dendrity neuronů dalších.

Umělé neuronové sítě byly ve svých počátcích inspirovány biologickou analogií, nicméně po ustavení základních technických konceptů probíhal jejich rozvoj zcela samostatně, již bez přímé vazby na původní motivaci. Matematická reprezentace a modely umělých neuronových sítí jsou tedy oproti známé biologické realitě pouze velmi zjednodušenými modely, navíc často poměrně vzdálenými.

1.3.2 Historie NN

Uveďte si stručně v bodech základní linii vývoje umělých neuronových sítí. V roce 1943 vytvořili Warren McCulloch a Walter Pitts jednoduchý matematický model neuronu, který představoval

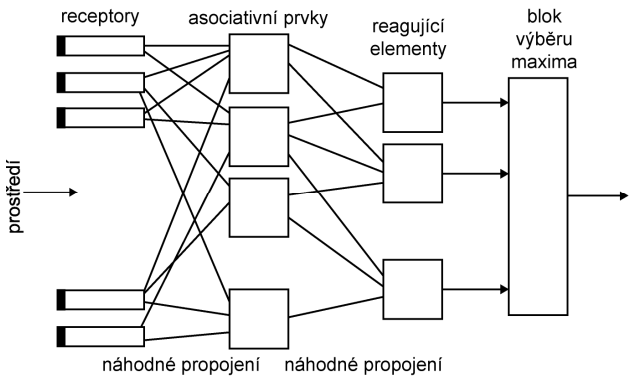
teoretický model části nervové soustavy – biologického neuronu. Nepředpokládali, že se jejich návrh uplatní v praxi, nicméně model se stal předmětem zkoumání a rozvoje řady dalších badatelů. V jejich modelu měl každý neuron několik vstupů a jeden výstup. Vstupy byly rozděleny na excitační a inhibiční a model akceptoval pouze m binárních vstupů a jeden výstup. Prováděl tedy vyčíslení funkce z $\{0,1\}^m$ -> {0,1}. Princip byl takový, že když převáží na vstupu excitační buzení nad inhibičním, je excitován i výstup neuronu. V opačném případě neuron excitován nebude a na výstupu neuronu tedy bude 0. Vstupní synapse nebyly nijak váhovány, sloužily pouze pro přenos hodnoty.



Obr. 2 *Model neuronu dle McCulloch-Pittse (podle [3]).*

V roce 1949 definoval Donald Hebb opět na základě biologické analogie a studia podmíněných reflexů pravidlo, které umožňovalo učit neuron změnou vah jeho vstupů. Vycházel z předpokladu, že pokud je neuron excitován korektně, pak se posílí spoje, které k excitaci vedly. Naopak, pokud je neuron excitován nesprávně, pak je třeba tyto spoje oslabit.

V roce 1957 zobecnil Frank Rosenblatt McCulloch-Pittsův neuron a představil umělou neuronovou síť, perceptron. Tato síť sloužila k rozpoznávání znaků promítaných na plátno a opět byla inspirována biologií – tedy objevem světločivých buněk na sítnici oka a způsobem přenosu signálu do CNS. Architektura Rosenblattova perceptronu je na následujícícm obrázku.



Obr. 3 *Rosenblattův perceptron (podle [3]).*

V roce 1960 Bernard Widrow představil další zobecnění základního neuronu, adaptivní lineární element (ADALINE). Provedl zobecnění na reálné vstupní hodnoty a představil nové učící pravidlo neuronu.

V roce 1969 došlo ke krizi vývoje v oblasti umělých neuronových sítí, kdy významná osoba umělé inteligence Marvin Minsky poněkud účelově zdiskreditoval koncept umělých neuronových sítí. Využil k tomu tzv. XOR problém, kdy jednovrstvá neuronová síť není schopna logickou funkci XOR vyčíslit.

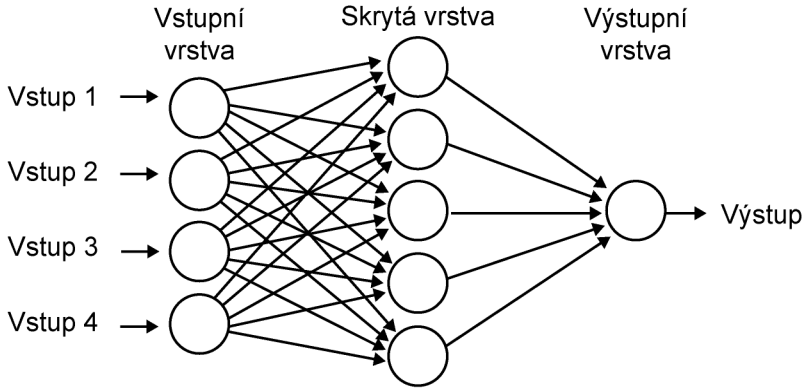
Konstatoval, že řešením je použití více vrstev neuronové sítě, nicméně není formalizován žádný algoritmu, který by umožnil takové sítě učit. Chybně předpokládal, že učící algoritmus nebude možné vzhledem ke komplikovanosti struktury sítě nalézt. Jeho závěry vedly k přesměrování grantových prostředků do jiných oblastí umělé inteligence koncept umělých neuronových sítí tak čekala stagnace. Nicméně v roce 1986 skupina badatelů odvodila algoritmu, kterým lze vícevrstvé sítě učit na principu zpětného šíření chyby - Backpropagation. Toto zjištění překonalo problém XOR a vedlo k intenzivnímu zájmu o oblast neuronových sítí, která trvá dodnes.

1.3.3 Koncept umělé neuronové sítě

Umělá neuronová síť je tvořena matematickými neurony, primitivními jednotkami, kde každá zpracovává váhované vstupní signály a generuje výstup.

Neuronová síť představuje topologické uspořádání jednotlivých neuronů do struktury komunikující pomocí orientovaných ohodnocených spojů. Každá umělá neuronová síť je tedy mimo jiné charakterizována typem neuronů, jejich topologickým uspořádáním a strategií adaptace při trénování (učení) sítě.

Základní představu o pojmu umělá neuronová síť si ukažme na principu nejčastěji používané dopředné neuronové sítě. Dopřednou je nazývána, protože signál se sítí šíří od vstupu jednosměrně směrem k výstupům sítě. Uspořádání neuronů pro dopřednou síť je uvedeno na následujícím obrázku.



Obr. 4 *Uspořádání neuronů do vrstev v dopředné neuronové sítě.*

Na obrázku lze rozeznat, že jednotlivé neurony této sítě jsou uspořádány do vrstev. Mezi neurony jedné vrstvy není propojení, mezi neurony sousedních vrstev zpravidla existuje propojení úplné. Spoje mezi neurony, které představují dráhy pro šíření signálu, jsou orientovány a každý spoj je ohodnocen vahou, která modifikuje intenzitu procházejícího signálu. Neexistenci konkrétního spoje lze modelovat spojem s vahou rovnou nule. První vrstvu dopředné sítě nazýváme vstupní vrstvou, poslední vrstvou nazýváme vrstvou výstupní a ostatní mezilehlé vrstvy nazýváme vrstvami skrytými. Zpravidla se dopředné neuronové sítě realizují jako sítě s jednou, eventuelně dvěma skrytými vrstvami.

Dopředná neuronová síť představuje masivně paralelní distribuovaný dynamický systém, kde každá jednotka, neuron, pracuje lokálně, samostatně. Taková neuronová síť se vyznačuje značnou robustností, je odolná vůči poškození, kdy obvykle dokáže poskytovat relevantní výstupy i při poškození některých elementů. Tyto vlastnosti jsou vhodné pro konstrukci neuropočítačů, tedy

počítačů založených na neuronových sítích. Je totiž mnohem snazší, vytvořit prvek s vysokou mírou integrace a některými chybnými elementy, než prvek s nižší integrací, kde budou všechny elementy bezvadné.

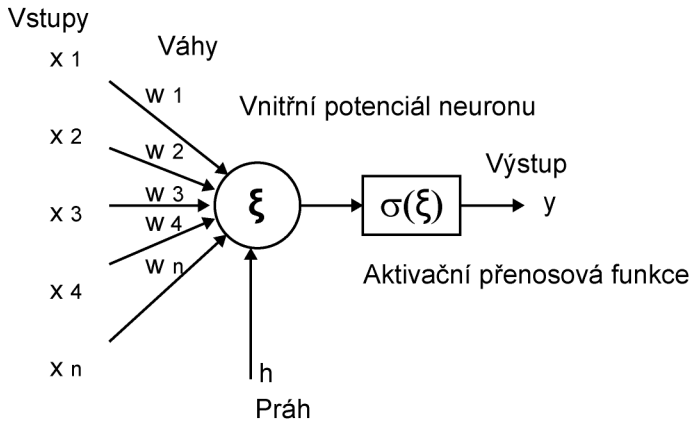
U neuronové sítě dané topologie, ale i jednotlivého neuronu, můžeme rozeznat dvě fáze aktivity. Fázi, kdy síť na základě předložených vstupů produkuje výstupy, topologie a nastavení parametrů sítě se nemění. Této fázi předchází fáze učení, kdy síť trénovacím algoritmem modifikuje své parametry, v některých případech i topologii sítě. Tyto fáze nazveme v souladu s [1] **aktivní** a **adaptační** dynamikou sítě či neuronu. U neuronových sítí lze rozeznat ještě dynamiku třetí, **organizační,** která představuje uspořádání neuronů do propojené struktury sítě a vytváří její výslednou topologii. Tato dynamika nemá smysl u jednotlivého neuronu.

Představená topologie dopředné sítě je pouze jedním, byť nejobvyklejším, z možných uspořádání neuronů. Existuje celá řada dalších konceptů, kde je propojení realizováno jinak, například je ve struktuře zavedena zpětná vazba, nebo jsou neurony propojeny každý s každým. S některými z těchto konceptů se také seznámíme, byť podrobněji se vzhledem k univerzálnosti řešení budeme věnovat dopředným neuronovým sítím.

1.4 Jednotlivý neuron

1.4.1 Matematický model a aktivní dynamika neuronu

Schematický model jednotlivého neuronu je uveden na následujícím obrázku. Neuron lze rozdělit na několik částí: Synapse ohodnocené vahami w, které přivádějí vstupy x do těla neuronu, vlastní tělo neuronu ve kterém je získáván vnitřní potenciál neuronu ξ, blok aktivační přenosové funkce σ a konečně výstup neuronu y. Do těla neuronu vstupuje ještě konstantní hodnota prahu h.



***Obr. 5** Model neuronu.*

Vstupem neuronu je vektor o n prvcích, které jsou obecně reálné. Podobně práh může být reálnou hodnotou a často je modelován jako jeden ze vstupů, jak si ukážeme dále. Vnitřní potenciál neuronu je hodnota, která vzniká porovnáním váhovaných a sečtených vstupů s prahem. Váhy jsou opět reálná čísla, mohou být tedy kladné i záporné a modelovat tak aktivační i inhibiční synapse. Aktivační přenosová funkce provádí obecně nelineární transformaci vnitřního potenciálu na jednu, obecně opět reálnou hodnotu. Neuron tedy provádí zobrazeni z $\{R\}^n$ -> R.

Podívejme se nyní podrobněji na aktivní dynamiku neuronu. Po přiložení vstupního vektoru x může být vnitřní potenciál neuronu ξ vyčíslen jako

$$\xi = \sum_{i=1}^{n} w_i x_i - h \quad (1)$$

Každý prvek vstupního vektoru je tedy násoben příslušnou vahou, tato hodnota je v těle neuronu sečtena pro všechny prvky vstupního vektoru a od výsledné hodnoty je odečten práh. Takto získaná hodnota vnitřního potenciálu se stává následně argumentem obecně nelineární aktivační přenosové funkce.

Z praktických důvodů se práh zpravidla modeluje jako jedna z vah tak, že vstupní vektor i vektor vah je rozšířen o nultou pozici. Vstup na nulté pozici je vždy uvažován za roven 1 a nultá váha je nastavena na hodnotu –h. V takovém případě se práh stává jednou z vah a v průběhu trénování podléhá adaptaci.

Celkovou odezvu, tedy výstup neuronu pak můžeme vyjádřit jako

(2)

, kde $x_0 = 1$ a $w_0 = -h$.

Aktivační přenosová funkce σ je obecně nelineární funkcí transformující hodnotu vnitřního potenciálu neuronu. Předpokládejme její nejjednodušší typ, ostrou nelinearitu, kdy platí

$$\sigma(\xi) = \begin{cases} 1 \; pokud \; \xi \geq 0 \\ 0 \; jinak \end{cases}$$

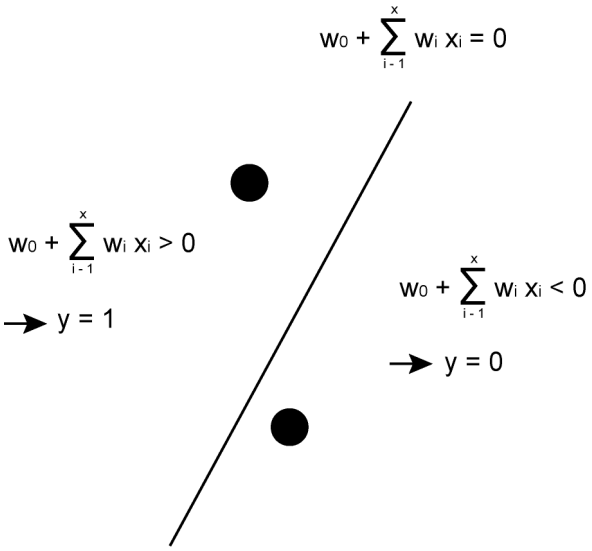
(3)

Nyní předpokládejme neuron pro n=2 se dvěma reálnými vstupy x_1, x_2 a odpovídajícími vahami w_1, w_2. Takto definovaný neuron realizuje zobrazení z $\{R\}^2$ -> {0,1}. Uvažujme, že vstupy x_1 a x_2 představují body v dvourozměrném Euklidovském prostoru. Neuron ve své aktivní dynamice reaguje na vstupy předkládané z tohoto prostoru a přiřazuje jim hodnoty 0 nebo 1. Provádí tak vlastně klasifikaci těchto bodů roviny do dvou skupin podle hodnoty aktivační přenosové funkce, tj. výstupu neuronu.

Pokud se podrobněji podíváme na vztah vyjadřující odezvu neuronu, zjistíme, že v tomto konkrétním případě je zařazení bodů dáno jejich pozicí vůči přímce definované aktivační funkcí, respektive přesněji vahami neuronu. Dělící přímka odlišující dvě skupiny bodů má tedy v dvourozměrném prostoru rovnici

$w_1x_1 + w_2x_2+w_0 = 0$

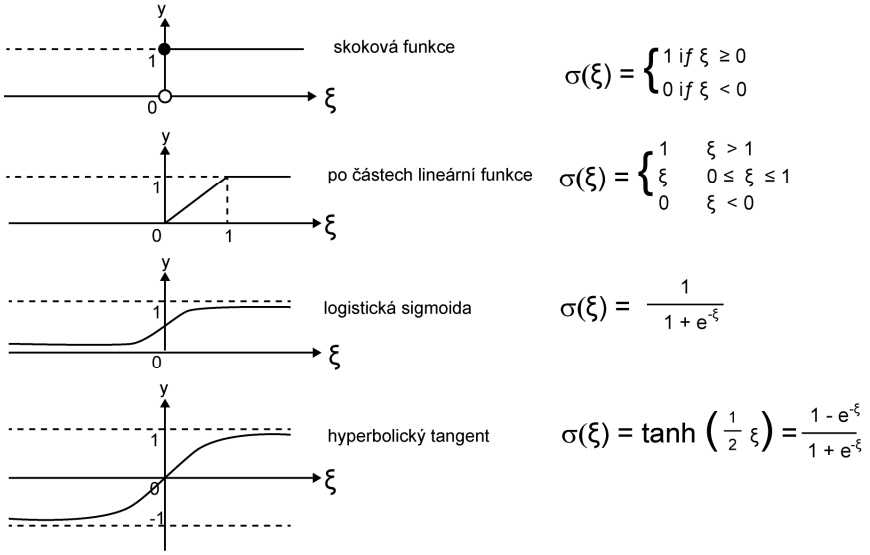
(4)



Obr. 6 *Ilustrace klasifikace neuronu v rovině (podle* http://www.byclb.com/TR/Tutorials/neural_networks/ch8_1.htm*]).*

Uvedenou ilustraci lze zobecnit na n-rozměrné Euklidovské prostory, kdy n-rozměrný vektor vah neuronu představuje nadrovinu dělící prostor na dva poloprostory.

Aktivační přenosové funkce nejčastěji používané v dopředných neuronových sítích jsou na následujícím obrázku. Obvyklou podmínkou pro volbu přenosové funkce je její (ideálně snadná) diferencovatelnost (logistická sigmoida, hyperbolický tangens), což umožňuje využít učících algoritmů založených na gradientních metodách. V jednoduchých případech, kdy nastavujeme parametry sítě jinými metodami (například přímým výpočtem), lze využít i funkce, které tuto podmínku nesplňují (skoková funkce, po částech lineární).



Obr. 7 *Aktivační přenosové funkce.*

Závěrem tohoto odstavce je třeba alespoň poznamenat, že jsou používány i odlišné koncepce neuronů, kde je výstup neuronu kalkulován jiným způsobem. Příkladem mohou být sítě s radiální bází

(RBF sítě), kde neuron vyčísluje vzdálenost vstupního vektoru x od vektoru vah w. Dalším reprezentantem jsou vlnkové sítěVlnkové sítě se používají zejména při analýze signálů a obrazů.

1.5 Adaptační dynamika neuronu

1.5.1 Principy učení neuronu obecně

Průběhu aktivní dynamiky provádí neuron transformaci vstupních vektorů na hodnotu výstupní. Parametry neuronu jsou v tuto chvíli konstantní. Oproti tomu adaptační dynamika je proces, který má za úkol tyto parametry neuronu nastavit tak, aby neuron prováděl požadovanou transformaci. Parametry, které jsou v průběhu učení neuronu adaptovány, jsou zpravidla jen váhy vstupních synapsí neuronu, včetně synapse představující práh. Vstupní váhy představují tedy paměť neuronu. Učení neuronu obstarává adaptační algoritmus, který nastavuje hodnotu vah neuronu. Tento proces probíhá zpravidla iterativně na základě předkládaných příkladů, kdy má adaptační algoritmus k dispozici množinu dvojic vstupních hodnot a jim odpovídajících výstupům. Ve výjimečných případech je možné hledané váhy určit přímým výpočtem.

Adaptační algoritmy tedy postupují podobně jako člověk, kdy hledají řešení na základě analogie se známými příklady. Nastavení vah neuronu tedy odpovídá nelezení co nejvýstižnější transformace vstupních vektorů na vektory výstupní na základě známých vstupních hodnot s předpokladem, že nalezená transformace bude dostatečně obecná i pro další neznámé příklady dané oblasti, byť tomu tak být nemusí.

Principy adaptace jednotlivého neuronu uvedené v dalších kapitolách lze snadno zobecnit i na dopředné vrstevnaté neuronové sítě. Základní výhodou neuronových sítí je možnost nalezení transformace i u těch úloh, které jsou analyticky obtížně řešitelné, či neřešitelné. Postačí k tomu dostatečné množství příkladů. Nevýhodou je naopak skutečnost, že výsledná transformace je ukryta ve struktuře sítě, nelze ji použít k vysvětlení řešení úlohy. Je třeba si dále uvědomit, že byť jsou sítě výkonným výpočetním modelem řešícím řadu obtížných úloh, záruku nalezení dostatečně obecné transformace neuronové sítě nezaručují.

1.5.2 Učení bez učitele

Adaptační algoritmy můžeme rozdělit do dvou základních oblastí, nazývané jako učení bez učitele, respektive učení s učitelem.

Pro učení bez učitele je typické, že adaptační algoritmus nemá k dispozici žádné kriterium správnosti hledané transformace vstupních dat. Pracuje na principu shlukování, kdy ve vstupním prostoru dat hledá „sobě podobné“ elementy. Na základě předkládaných vzorků vstupních dat provádí jejich třídění do skupin, bez možnosti posouzení správnosti zatřídění. Počet hledaných skupin může být předem dán. Do adaptace nevstupuje žádný arbitr, celé učení je založenou pouze na informacích obsažených ve vstupních datech. Algoritmus bez arbitra je tedy možné použít i v okamžiku aktivní dynamiky sítě, kdy je možné na základě každého nového předloženého vzorku upravit současně i parametry sítě. Učení tak může být i trvalého charakteru. Učení bez učitele nemá v případě jednotlivého neuronu význam, nicméně je používáno například pro celou třídu neuronových sítí nazývanou jako samoorganizující se mapy, kterým bude věnována samostatná podkapitola.

1.5.3 Učení s učitelem

V případě učení s učitelem má adaptační algoritmus dostupnou spočetnou konečnou (pmax prvků) množinu M dvojic x a y_d, které představují vstupy a odpovídající korektní výstupy řešené úlohy. Má tedy k dispozici příklady správného chování, správné transformace z vektorů vstupních na vektory výstupní. Tyto příklady mohou být získány například měřením a zaznamenáním vstupních a výstupních hodnot systému, který chceme modelovat. Přímá transformace vstupních hodnot na hodnoty výstupní není známa. Analogie s usuzováním lidského experta je zde zřejmá.

$$M = \{[\mathbf{x}^1, y_d{}^1], [\mathbf{x}^2, y_d{}^2] \ldots . [\mathbf{x}^{pmax}, y_d{}^{pmax}]\}$$

(4)

Množina všech dostupných hodnot tedy představuje známou část chování systému. Tato množina je pak použita adaptačním algoritmem k naučení sítě a také k ověření její funkce. Zpravidla je množina M všech dostupných dat rozdělena na dvě části, na trénovací a testovací množinu. Poměr počtu prvků v trénovací a testovací množině není pevně dán. Je třeba jej volit s ohledem na charakter řešené úlohy a také dat dostupných adaptačnímu algoritmu. Trénovací množina obvykle představuje při dostatečném počtu vstupních dat cca 70%-80%, zbytek dvojic je pak zařazen do množiny testovací. Často používaný je i způsob, kdy je množina dostupných hodnot rozdělena místo dvou na tři disjunktní části, mimo množiny trénovací a testovací je vytvořena také množina validační.
V průběhu učení je periodicky vyhodnocována chyba nikoli jen na trénovací množině, ale je určována i chyba na množině validační a adaptace je zastavena, pokud tato chyba po nějakou dobu neklesá. Tím je dosaženo větší obecnosti nalezené transformace.

Vlastní trénování neuronu (a po zobecnění i dopředných sítí) probíhá obvykle iterativně, kdy algoritmus jednotlivé prvky trénovací množiny předkládá postupně neuronu, zjišťuje jeho odezvu na předložený vstup a na základě odchylky jeho výstupu od výstupu požadovaného provádí korekci vah neuronu. Interval, ve kterém dojde k předložení všech vzorů trénovací množiny alespoň jednou, nazýváme epochou učení. K naučení sítě může být dle komplexnosti problému zapotřebí desítky až tisíce epoch.

Okamžik zastavení adaptace může být definován různými způsoby, nejčastěji dosažením požadovaně malé chyby transformace nad trénovací množinou či zastavením jejího poklesu, dosažením maximálního počtu epoch. Za účelem vyhodnocení okamžiku, kdy je vhodné adaptaci ukončit (nalezená transformace je dostatečně přesná, ale zároveň neztrácí na obecnosti) vyčleňujeme z trénovací množiny, respektive dostupných dat často ještě třetí, opět se nepřekrývající množinu validační. V průběhu adaptace je nad ní periodicky testována výkonnost nalezené transformace a při zastavení poklesu chyby nad validační množinou je proces adaptace ukončen. Rozdíl mezi validační množinou a množinou testovací je ten, že validační množina vstupuje do procesu adaptace jako parametr ukončující učení sítě, testovací množina nikoli a je na ní po ukončení adaptace pouze otestována předpokládaná výkonnost sítě v praxi.

Výkonnost naučeného neuronu či neuronové sítě pak po ukončení adaptace ověříme na testovací množině. Trénovací a testovací množinu volíme tedy jako nepřekrývající se, aby byla zaručena možnost nezávislého otestování úspěšnosti adaptace nad testovací množinou. Kriterií výkonnosti sítě a úspěšnosti adaptace může být více, nejčastěji se jedná o střední kvadratickou odchylku mezi výstupy nalezené transformace a výstupy očekávanými.

Rozsah a obsah trénovací množiny je často limitován už dostupnými daty. Přesto se snažíme, aby měl adaptační algoritmus k dispozici co nejvíce příkladů, které pokrývají pokud možno rovnoměrně celý vstupní prostor. Klademe tedy požadavek na reprezentativnost příkladů, které jsou adaptačnímu algoritmu předkládány.

Úspěšnost transformace nalezené nad trénovací množinou hodnotíme nad množinou testovací a odhadujeme tak úspěšnost proběhlé adaptace a obecnost nalezené transformace i pro jiné vstupy, než se nacházely v trénovací množině. Výkonnost nad testovací množinou je tedy hodnoceným kriteriem úspěšnosti adaptace.

Předpokládáme, že pokud je výkonnost nad testovací množinou dobrá, bude přibližně stejně dobrá i pro vstupy zcela mimo trénovací a testovací množinu. Je ale třeba zmínit, že tuto záruku obecnosti pro dosud neznámé vstupy nemáme.

Typické kroky iterativního učení neuronu či dopředné neuronové sítě s učitelem.

1. Předzpracování vstupních dat
2. Definice trénovací, testovací a případně validační množiny
3. Definice struktury sítě/parametrů neuronu
4. Inicializace vah neuronů, obvykle malá náhodná čísla
5. n=0; počítadlo epoch učení
6. Start: nastav epochu učení na n=n+1
7. Pokud je počet provedených epoch učení >= max => ukonči učení, jdi na 14
8. n-tá epocha učení:
9. Výběr a předložení jednoho vstupního vektoru neuronu z trénovací množiny (deterministicky, náhodně)
10. Získání odezvy neuronu, vyhodnocení chyby klasifikace na základě porovnání skutečného a předpokládaného výstupu => chyba klasifikace
11. Korekce vah neuronu na základě získané chyby
12. Pokud není epocha učení dokončena (neprověřeny všechny vstupy z trénovací množiny), jdi na bod 9
13. Na konci epochy vyhodnoť chybu klasifikace přes celou trénovací množinu. Je-li chyba<ε ukonči učení (bod 14), jinak jdi na bod 4. (Vyhodnocovat chybu lze i častěji a použít i validační množinu a použít tak jiné kriterium pro zastavení adaptace)
14. Vyhodnocení úspěšnosti činnosti sítě na testovací množině, pokud je nedostatečná, jdi na bod 4, ev. 3. Případně ukonči algoritmus neúspěchem.

1.5.4 Hebbovo učení

Představuje nejstarší intuitivní pravidlo pro neuron s binárními vstupy i výstupem. Bylo definováno v roce 1949 Donaldem Hebbem, který je definoval při studiu podmíněných reflexů. Předpokládal, že podmíněné reflexy jsou ustavovány na základě posilování či oslabování vazeb mezi jednotlivými neurony. Předpokládal, že platí, že pokud jsou dva propojené sousední neurony aktivní současně, vazba mezi nimi se posiluje, zatímco při nesouhlasné aktivaci oslabuje. Pro jeden samotný neuron lze tuto formulaci chápat tak, že v případě, že je neuron excitován svými vstupy korektně, jsou posilovány buzené vstupy (respektive jejich váhy), v opačném případě nekorektní excitace jsou naopak buzené spoje oslabovány. Hebbovo pravidlo tedy vychází z neurofyziologické analogie. Pro neuron s binárním vstupem x, vahami w, výstupem y a předpokládaným výstupem y_d lze Hebbovo pravidlo zapsat následně.

1. pokud je neuron excitován korektně ($y=1;y_d=1$), pak se v příštím diskrétním časovém kroku n+1 posilují o Δ spoje w_i, které tuto excitaci vyvolaly (${}_{n+1}w_i = w_i + \Delta, \forall i:x_i = 1$)

   (5)

2. pokud je neuron excitován nekorektně ($y=1;y_d=0$), pak se oslabují o Δ spoje, které tuto excitaci vyvolaly (${}_{n+1}w_i = w_i - \Delta, \forall i:x_i = 1$)

   (6)

3. pokud není neuron excitován (y=0), nic se neděje (váhy se nemění)

1.5.5 Delta pravidlo

Dalším, původně heuristickým pravidlem, které je použitelné i pro obecně reálné vstupy a výstupy neuronu je delta pravidlo. Lze jej zapsat jako

${}_{n+1}w_i = {}_nw_i + \mu*(y_d-y)$

(7)

Vektorově (tučný font) pak jako

${}_{n+1}\mathbf{w} = {}_n\mathbf{w} + \mu*(y_d-y)$

(8)

, kde μ je vhodně zvolená konstanta mezi (0,1) ovlivňující rychlost adaptace.

Delta pravidlo platí přesně pro lineární neurony, tedy neurony s lineární aktivační přenosovou funkcí, ale po modifikaci je použitelné i pro neurony s nelineární aktivační přenosovou funkcí, jak si ukážeme později v kapitole věnované učení vícevrstvých dopředných sítí.

1.5.6 Učení neuronu podle Widrowa

Vychází z geometrické interpretace učení neuronu s binárním výstupem. Představa je taková, že neuron rozděluje v souladu s kapitolou „Matematický model a aktivní dynamika neuronu“ výstupní prostor na dva poloprostory a provádí tak klasifikaci vstupních vektorů do jednoho z nich.

V případě, že je klasifikace daného vzoru ze vstupní množiny správná, není zapotřebí žádné úpravy vah. V případě, že je klasifikace chybná, leží klasifikovaný bod ve špatné části poloprostoru rozděleného dělící nadrovinou

$\mathbf{w}\bullet\mathbf{x}=0$

(9)

Symbol ● značí skalární součin vektorů, tučně označené symboly jsou vektory**.** Vzdálenost tohoto chybně klasifikovaného bodu od dělící nadroviny lze vyjádřit jako

$$d = \left| \mathbf{w} \bullet \mathbf{x} \right|$$

(10)

, kde tato hodnota může být použita kvantifikátor chyby klasifikace.

Zavedeme chybovou funkci, která vyjadřuje sumu vzdáleností od nadroviny pro všechny špatně klasifikované vzory X_f trénovací množiny jako funkci vektoru **w.**

$$E(\mathbf{w}) = \sum_{\in X_f} \left| \mathbf{w} \bullet \mathbf{x} \right| = \sum_{\in X_f} \mathbf{w} \bullet \pm \mathbf{x}$$

(11)

, kde **+x** použijeme při falešně negativní klasifikaci a **–x** při falešně pozitivní klasifikaci.

Cílem adaptačního algoritmu je minimalizovat změnou vah **w** chybovou funkci E přes všechny vzory trénovací množiny. Minimalizace chybové funkce se provádí ve směru (funkce už obsahuje změnu znamének dle chybné klasifikace) gradientu E(**w**) a platí:

$_{n+1}\mathbf{w} = {}_{n}\mathbf{w} + \mu * \nabla E(\mathbf{w}_n)$

(12)

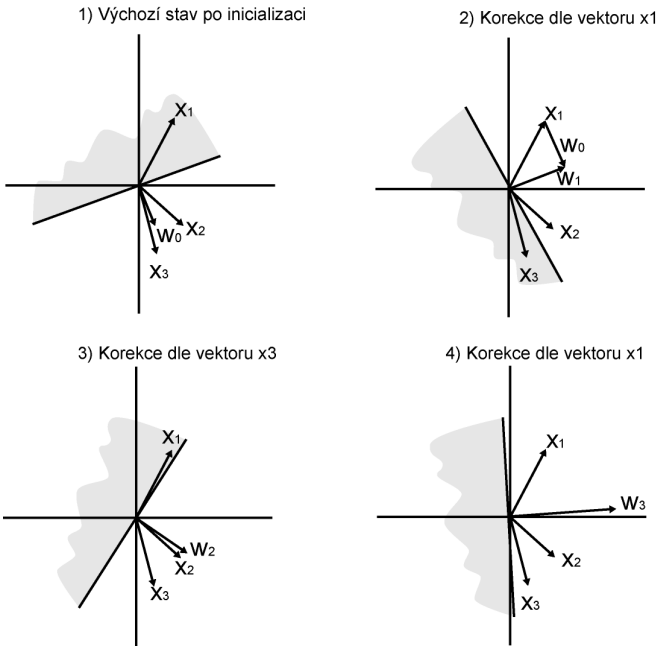
$$\nabla E({}_{n}\mathbf{w}) = \sum_{\in X_f} \pm \mathbf{x}$$

(13)

Funkce E(**w**) je tedy chybovou funkcí přes všechny klasifikované vzory trénovací množiny v jedné epoše učení. Uvedený postup lze samozřejmě následně opakovat pro epochy další, až do ukončení adaptace vah. Dle tohoto pravidla lze postupovat nejen po jednotlivých epochách, ale i iterativně pro jednotlivé vstupní vzory **x** a po částech se tak blížit minimu chybové funkce E(**w**).

Geometrická interpretace adaptace binárního neuronu je tedy taková, že při adaptaci dochází k posunu dělící nadroviny definované vektorem vah neuronu. Vektor vah **w** představuje normálový vektor dělící nadroviny.

Algoritmus provádí úpravu vah vektoru **w** dle pravidla pro chybně klasifikované vzory $_{n+1}\mathbf{w} = {}_{n}\mathbf{w} \pm \mathbf{x}$, nový vektor vah tedy získáme sečtením vektoru původních vah a přiloženého, chybně klasifikovaného vektoru. Pokud je přiložený vzor **x** chybně klasifikovaný a zároveň platí, že v daném kroku n **|x|>>|w|,** provede tato iterace významnou korekci vektoru **w** a vektor **w** se prakticky v prostoru velmi přiblíží k vektoru **x.** V průběhu učení obvykle dochází s rostoucími iteracemi postupným doučováním ke zmenšování uhlu rotace vektoru w a ke stabilizaci umístění dělící nadroviny.



Obr. 8 *Geometrická ilustrace adaptace v případě, že vektory X_1,X_2,X_3 mají být korektně zařazeny do bílé části poloroviny.*

1.6 Klasifikační schopnosti jednotlivého neuronu

V rámci popisu adaptační a aktivní dynamiky jednotlivého neuronu s binárním výstupem jsme konstatovali, že vektor vah neuronu tvoří normálový vektor nadroviny. Jednotlivý neuron je tedy schopen klasifikovat vstupní vektory do dvou tříd. V následujících odstavcích si na modelu logických funkcí představíme klasifikační schopnosti a omezení jednotlivého neuronu. Budeme předpokládat neuron $\{0,1\}^2$ -> {0,1} se dvěma binárními vstupy (x_1, x_2), jedním binárním výstupem a prahem rovným $x_0 = 1$.

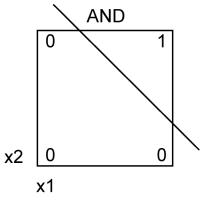
1.6.1 Realizace logické funkce AND

Logická funkce AND dvou proměnných je definována jako

X_1	X_2	X_1 AND X_2
0	0	0
0	1	0
1	0	0
1	1	1

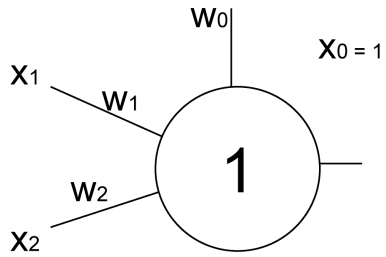
Tab. 1 *Logická funkce AND.*

V rovině lze logickou funkci AND zobrazit dle Obrázku 8.



Obr. 9 *Logická funkce AND v rovině.*

Cílem je nastavit váhy neuronu tak, aby neuron realizoval logickou funkci AND.



Obr.10 *Neuron pro realizaci funkce AND.*

Nastavení vah je možné provést nerovnostmi, kdy pro korektní klasifikaci musí být splněny podmínky

$(0, 0, 1) \cdot (w_1, w_2, w_0) < 0$

$(1, 0, 1) \cdot (w_1, w_2, w_0) < 0$

$(0, 1, 1) \cdot (w_1, w_2, w_0) < 0$

$(1, 1, 1) \cdot (w_1, w_2, w_0) > 0$

,tedy

$w_1 0 + w_2 0 + w_0 1 < 0$

$w_1 0 + w_2 1 + w_0 1 < 0$

$w_1 1 + w_2 0 + w_0 1 < 0$

$w_1 1 + w_2 1 + w_0 1 > 0$

Řešením této soustavy nerovnic jsou například hodnoty $w_1 = 1$, $w_2 = 1$ a $w_0 = -1{,}5$.

$1*0 + 1*0 - 1{,}5*1 < 0$ ➔ $y = 0$;

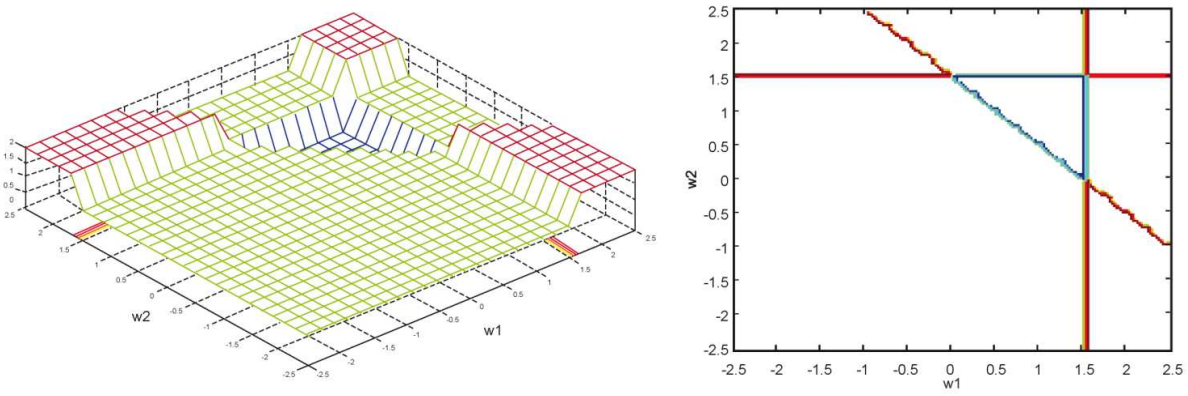
$1*0 + 1*1 - 1{,}5*1 < 0$ ➔ $y = 0$;

$1*1 + 1*0 - 1{,}5*1 < 0$ ➔ $y = 0$;

$1*1 + 1*1 - 1{,}5*1 > 0$ ➔ $y = 1$;

Logickou funkci AND lze tedy bezproblémově realizovat dle uvedeného postupu jediným neuronem.

Pro dokreslení způsobu řešení problému ještě na chvíli předpokládejme, že w_0 je konstantní, $w_0 = -1{,}5$. Hledáme tedy takovou konfiguraci vah w_1 a w_2, pro kterou budou výstupy neuronu korektní pro všechny čtyři možné kombinace vstupů x_1 a x_2. Pokud vyneseme závislost počtu chybných výstupů neuronu na vahách w_1 a w_2 v intervalu <-2,5;2,5> do grafu, získáváme následující obrázky.



Obr. 11 *Grafické řešení nastavení vah neuronu pro realizaci funkce AND.*

Z obrázku je zřejmé, že řešení, tedy možné hodnoty hledaných vah, představuje modrý trojúhelník. Dříve ověřené nastavení vah $w_1 = 1$ a $w_2 = 1$ se nachází uvnitř této modré oblasti. V této oblasti je chybová funkce nulová, neuron realizuje logickou funkci AND. Oblast, kde je hodnota chybové funkce rovna 2, je vyznačena červeně, oblast s hodnotou chybové funkce rovnou 1 pak zeleně.

V obecném případě nebývá adaptační algoritmus schopen dosáhnout nad trénovací množinou nulovou hodnotu chybové funkce, jako v případě realizace logické funkce AND. Nicméně postupuje dle svého algoritmu po jednotlivých krocích prostorem chybové funkce a snaží se korekcí vah neuronu nalézt její minimální hodnotu. Komplexnost této funkce závisí na počtu vstupů trénovací množiny a počtu vah neuronu.

1.6.2 Realizace logických funkcí OR a NOT

K analogickému závěru dospějeme v případě realizace logických funkcí OR a NOT pomocí jednotlivého neuronu.

X_1	X_2	X_1 OR X_2	NOT X_1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0

Tab. 2 *Logické funkce OR a NOT.*

Řešení soustavy nerovnic pro logickou funkci OR může být následující, tedy $w_1 = 1$, $w_2 = 1$ a $w_0 = -0,5$.

$w_1 0 + w_2 0 + w_0 1 < 0$ $\quad 1*0 + 1*0 - 0,5*1 < 0$ ➔ $y = 0$;

$w_1 0 + w_2 1 + w_0 1 > 0$ $\quad 1*0 + 1*1 - 0,5*1 > 0$ ➔ $y = 1$;

$w_1 1 + w_2 0 + w_0 1 > 0$ $\quad 1*1 + 1*0 - 0,5*1 > 0$ ➔ $y = 1$;

$w_1 1 + w_2 1 + w_0 1 > 0$ $\quad 1*1 + 1*1 - 0,5*1 > 0$ ➔ $y = 1$;

Řešením pro logickou funkci NOT pak dostáváme při $w_2 = 0$

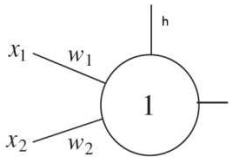
$w_1 1 + w_0 1 < 0$ -1*1 + 0,5*1 < 0 ➔ y = 0;

$w_1 0 + w_0 1 > 0$ -1*0 + 0,5*1 > 0 ➔ y = 1;

1.6.3 Realizace logické funkce XOR

Předcházející odstavec demonstroval, že výpočetní síla jednotlivého neuronu je zcela dostatečná pro výpočet logických funkcí AND, OR a NOT. Jaká je situace v případě realizace logické funkce XOR (exlusivní OR)?

Předpokládejme stejný neuron jako v předcházejícím případě, pouze pro zjednodušení s prahem označeným jako h (víme, že je realizován vahou w_0 a konstantní vstupem x_0).

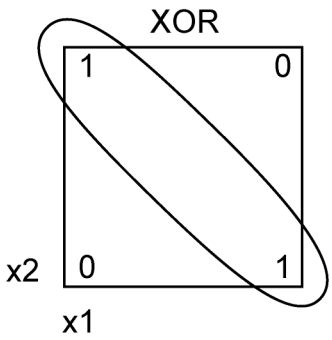


Obr. 12 *Neuron pro realizaci funkce OR a NOT.*

Tabulka hodnot funkce XOR a její grafické znázornění je uvedeno na následujícím obrázku

X_1	X_2	X_1 XOR X_2
0	0	0
0	1	1
1	0	1
1	1	0

Tab. 3 *Logická funkce XOR.*



Obr. 13 *Logická funkce XOR v rovině.*

Musí být tedy současně splněny následující předpoklady:

$w_1 0 + w_2 0 < h$ ➔ $0 < h$

(13)

$w_1 0 + w_2 1 > h$ ➔ $w_2 > h$

(14)

$w_1 1 + w_2 0 > h \rightarrow w_1 > h$

(15)

$w_1 1 + w_2 1 < h \rightarrow w_1 + w_2 < h$

(16)

Pokud vztahy analyzujeme podrobněji, zjistíme, že:

(13) implikuje, že práh je kladný.

(13) a (14) implikuje, že w_2 je kladná a větší než práh.

(13) a (15) implikuje, že w_1 je kladná a větší než práh.

Může být součet dvou kladných hodnot, kdy jsou obě vetší než práh, současně menší než práh, jak vyžaduje bod (16)? Je zřejmé, že tomu tak být nemůže. Předpoklady pro realizaci funkce XOR jednotlivým neuronem tedy nemůžou být splněny a logická funkce XOR nemůže být jednotlivým neuronem realizována.

1.6.4 Souhrn klasifikačních schopností jednotlivého neuronu

Jednotlivý neuron s binárním výstupem dokáže tedy odlišit pouze dvě lineárně separabilní třídy. Neuron rozděluje prostor vstupů na dva podprostory nadrovinou, jejíž normálový vektor je tvořen vektorem vah neuronu. Učení neuronu představuje zpravidla iterativní postup, kdy je nastavován vektor vah neuronu a tedy i měněna poloha dělící nadroviny na základě vyhodnocení klasifikace vstupních vektorů trénovací množiny do jedné z výstupních tříd. Vyhodnocení úspěšnosti klasifikace je realizováno prostřednictvím chybové funkce, kdy učení představuje pokus o její minimalizaci.

Logická funkce XOR není lineárně separabilní, jednotlivým neuronem tedy nemůže být realizována. Toto omezení vedlo ke krizi ve vývoji umělých neuronových sítí, kterého využil Minsky při své argumentaci proti konceptu umělých neuronových sítí.

Východisko z této situace ale existuje a spočívá v použití dopředných neuronových sítí s více vrstvami vzájemně propojených neuronů.

1.7 Seznam použité literatury

[1] Šíma, J., Neruda, R.: Teoretické otázky neuronových sítí, MATFYZPRESS, 1996, ISBN 80-85863-18-9.

[2] Jan, J.: Číslicová filtrace, analýza a restaurace signálů. VUT v Brně, VUTIUM, 2002. ISBN 80-214-1558-4.

[3] V. Kvasnička, L. Beňušková, J. Pospíchal, I. Farkaš, P. Tiňo, and A. Kráľ. Úvod do teórie neurónových sietí, IRIS, Bratislava, 1997, ISBN 80-88778-30-1.

[4] Volná, E.: Neuronové sítě 1. Skripta Ostravská universita v Ostravě, Ostrava, 2008.

[5] Volná, E.: Neuronové sítě 2. Skripta Ostravská universita v Ostravě, Ostrava, 2008.

[6] Osička P.:Jednoduché modely neuronu, [online], 2012, *phoenix.inf.upol.cz/~osicka/courses/uns/uvod.pdf*.