PB016 – Úvod do umělé inteligence

NEURONOVÉ SÍTĚ
ZÁKLADY Z TEORIE A PRAXE

David Kabáth

podzim 2009

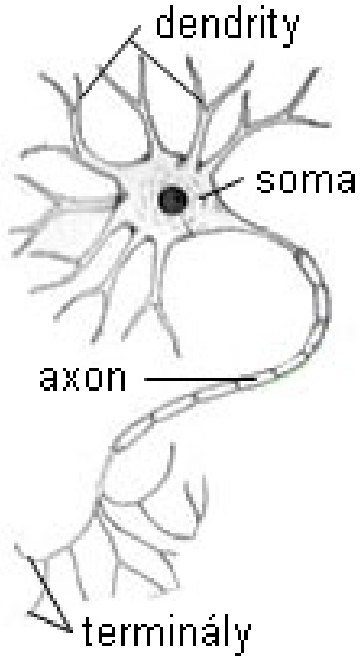
1 Úvod

Neuronové sítě jsou jedním z výpočetních modelů používaných v umělé inteligenci. Oproti klasické von Neumannovské architektuře počítače nabízí velký prostor pro paralelizaci, a tím pádem i zrychlení výpočtů. V dnešní době však existují více v podobě matematických modelů a softwarových řešení a další rozvoj je možný až pouze se specializovanými neuronovými počítači.

Následující stránky v kapitolách 2 a 3 prezentují biologickou předlohu a stručnou historii umělých neuronových sítí. Kapitola 4 je úvodem do problematiky a v dalších kapitolách 5, 6 a 7 jsou představeny základní principy fungování jednoduchých neuronových sítí na nejznámějších modelech – *Percetronu*, *Vícevrstvé síti* a *Hopfieldově síti*. Každý z těchto modelů je prezentován pomocí názorného JavaAppletu.

2 Biologická motivace

Stejně jako mnoho jiných informatických odvětví[1], tak i umělé neuronové sítě získaly prvotní inspiraci v biologické předloze. *Biologické neuronové sítě*, jakožto ucelené části formující nervovou soustavu jedince, byly zkoumány od konce 19. století. První velké výsledky přišly na počátku 20. století po objevu chemických metod[2], které umožňovaly pozorování základní stavení částí neuronových sítí, neuronů. Zjednodušeně popsáno, se *neuron* skládá z těla nazývaného *soma* a vstupních a výstupních výběžků (obrázek 1). Vstupní se nazývají *dendrity* a jejich množství se může blížit až ke 100 000. Na druhou stranu výstupní je pouze jeden a nazývá se *axon*. Konec axonu se rozděluje do velkého množství větví pojmenovaných *terminály*.



Obrázek 1: Stavba neuronu

Nejdůležitější vlastností neuronů je jejich schopnost mezi sebou komunikovat. Pokud je neuron aktivován předchozí komunikací s jinými neurony, šíří svoji informaci dále. Ta je jako elektrický impuls přenesena přes axon do terminálů. V tomto místě, nazývaném *synapse*, kde se terminál jednoho neuronu téměř dotýká dendritu jiného neuronu[3], probíhá samotná mezineuronová komunikace. Je přitom vygenerována chemická látka, která je přenesena do druhého neuronu.

Na závěr je třeba zmínit velmi důležitou schopnost neuronových sítí – paměť. Ta je simulována pomocí měnící se propustnosti membrány, která pokrývá celý neuron. Pokud spolu dva sousedící neurony komunikují, propustnost membrány roste, pokud spíše mlčí, propustnost klesá.

[1] dalšími jsou například Celulární automaty, Genetické algoritmy

[2] o to se zasloužil španělský lékař Santiago Ramón y Cajal (1852 – 1934), který pomocí Golgiho metody obarvil neurony tak, že byla mikroskopově pozorovatelná jejich struktura

[3] jde o vzdálenost přibližně 20 nanometrů

3 Pohled do historie

> **Poznámka:** Od této chvíle budou *umělé neuronové sítě* nazývány pouze jako *neuronové sítě*. Následující text se věnuje hlavně těm umělým, proto by nemělo docházet k žádným nedorozuměním. Pokud by se jednalo o biologické, bude tato skutečnost náležitě zdůrazněna.

Za první zmínku o neuronových sítích se považuje práce Warrena McCullocha a Waltera Pittse[1] z roku 1943. Přinesli v ní návrh matematické reprezentace biologického neuronu a na jeho základech pomocí elektrických obvodů sestrojili jednoduchou neuronovou síť. Ačkoliv šlo o nesporný úspěch, stále ještě chyběla charakteristika *učícího pravidla* (tzn. simulace biologického procesu změny propustnosti membrány). Tu představil v roce 1949 Donald Hebb ve své knize *The organization of behavior*[2].

V následujících letech pozvolna rostl zájem o výzkum na poli neuronových sítí. Nicméně opravdu prudký rozvoj přišel až roku 1957 s novým modelem neuronu – *perceptronem*, jehož autorem je Frank Rosenblatt[3]. Šlo o rozšíření základního modelu od McCullocha& Pittse. Rosenblatt zformuloval nové učící pravidlo, o kterém matematicky dokázal, že pro náhodně zadanou konfiguraci sítě nalezne po konečném počtu kroků takové nastavení sítě, které řeší daný problém (za předpokladu, že existuje).

Po tomto objevu se výzkum neuronových sítí rozběhl naplno a získal si zájem jak vědců, tak veřejnosti. Paradoxně tento velký zájem způsobil o několik let později i přicházející úpadek. Velká očekávání požadovala velké výsledky, které ale nepřicházely. Vědci i finance se postupně od neuroprojektů odkláněli. Poslední pokusy o obnovení výzkumu udusil Marvin Minsky a Seymour Papert svým dílem *Perceptrons*[4]. Svoji kritiku postavili na známém faktu nemožnosti řešení funkce XOR pomocí jednoho neuronu. Vznesli otázku, zda-li má cenu dále pokračovat ve výzkumu neuronových sítí, když nedokáží vyřešit ani tento jednoduchý úkol. V té době sice byla známa dvojvrstvá architektura o třech neuronech, která tento problém řešila, ale nebyl znám učící algoritmus pro vícevrstvé sítě. Tato kritika byla poté rozšířena a na další období se výzkum neurosítí téměř zastavil[4].

Roku 1982 přišla postupná renesance, ke které přispěl zejména fyzik John Hopfield se svým modelem, kterým umně propojil neuronové sítě s fyzikálními modely magnetických materiálů[5]. O čtyři roky později byl dokonce vyřešen problém učícího pravidla pro vícevrstvé sítě. David Rumelhart, Geoffrey Hinton a Ronald Williams zveřejnili řešení *backpropagation*[6]. Zajímavostí určitě je, že podobný algoritmus byl prezentován již dříve například Arthurem Brysonem a Yu-Chi Ho v roce 1969[7], ale sílící kritikou neuronových sítí byl utlumen.

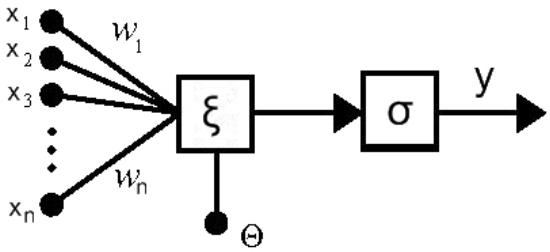
Od osmdesátých let, kdy byl výzkum obnoven, bylo dosaženo několika slibných výsledků. Jedním příkladem za všechny může být systém NETtalk[5]. Výzkum dále probíhá a pokrok neuronových sítí leží zejména ve vývoji specializovaného hardware. Až blízká budoucnost ukáže, zda-li tato snaha přinese kýžené ovoce.

[4] Někdy je toto období označováno jako *AI winter*

[5] *NETtalk* je projekt Terrence Sejnowského a Charlese Rosenberga z poloviny osmdesátých let postavený na učícím algoritmu *Backpropagation*. Zabývá se převodem anglicky psaného textu v mluvené slovo.

4 Od obecného ke konkrétnímu

V předcházejících dvou kapitolách jsme se seznámili s biologickou motivací a historií neuronových sítí. Následující tři kapitoly jsou již věnovány konkrétním modelům. Dříve než k nim ale postoupíme, stanovme jednotný formát, který použijeme při prezentování jednotlivých modelů. Následující řádky se drží výkladu dle knihy Jiřího Šímy a Romana Nerudy *Teoretické otázky neuronových sítí*[8].



Obrázek 2: Neuron

Základním stavebním kamenem každé neuronové sítě je *neuron*. Postupujme podle obrázku 2 zleva doprava a popišme si jeho jednotlivé části. Nejprve máme dáno n *vstupů* $x_1, x_2, ..., x_n$, které jsou ohodnoceny n *váhami* $w_1, w_2, ..., w_n$. Dále Θ značí *vnitří práh*, ξ označuje *vnitřní potenciál* a σ *aktivační funkci*. Nakonec máme y, které označuje *výstup* neuronu.

Jednotlivé části matematického modelu jsou známy a nyní si vysvětleme jejich funkci. Neuron začne pracovat tak, že z daných vstupů vypočte vnitřní potenciál.

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

Poté aktivační funkce za pomocí Θ vyhodnotí, zda-li je ξ dostatečně vysoký. Toto je nejjednodušší interpretace aktivační funkce, pojmenovaná *ostrá nelinearita*, kde je neuron buďto aktivní (vrací 1) či nikoliv (vrací 0). Na konkrétních příkladech uvedených dále se zmíníme i o jiných typech aktivačních funkcí. Formálně zapsáno:

$$\sigma(\xi) = \begin{cases} 1, & \xi \geq \Theta \\ 0, & \xi < \Theta \end{cases} \tag{2}$$

Nyní již stačí jen dodat, že pro výstup neuronu y platí:

$$y = \sigma(\xi) \tag{3}$$

Neuronová síť se tedy skládá z neuronů, které jsou vzájemně pospojovány tak, že výstup neuronu může být zároveň vstupem dalších neuronů. Pro přehlednější popis neuronových modelů použijme pojmy *konfigurace*, *topologie*, *aktivní fáze*, *adaptivní fáze*.

- *Konfigurace* je vektor vah všech neuronů tvořících danou síť.
- *Topologie* označuje architekturu sítě – kolik má neuronů, jak jsou pospojovány apod.
- *Aktivní fáze* popisuje činnost od konkrétního vstupu sítě až k dosažení výstupu.
- *Adaptivní fáze* značí změnu konfigurace sítě – proces učení.

Následující kapitoly se již věnují konkrétním modelům, u kterých se již budeme držet výše uvedených pojmů. Jednotlivé modely jsou diskutovány na Internetu volně přístupných JavaAppletech, doporučujeme proto jednotlivé applety zároveň vyzkoušet.

5 Perceptron

WWW: *http://lcn.epfl.ch/tutorial/english/perceptron/html/index.html*

Autor: Fred Corbett

Topologie: Jedná se o jeden neuron přesně odpovídající modelu uvedenému v kapitole 4.

- *Vstupy*: x_1, x_2
- *Váhy*: w_1, w_2
- *Vnitřní práh*: u
- *Použitá aktivační funkce*: přednastavená ostrá nelinearita (viz. kapitola 3, vzorec 2).
- *Výstup*: y

Aktivní fáze: Výpočet pro daný vstup x_1, x_2 probíhá přesně podle schématu uvededeného v kapitole 3. Nejprve výpočet vnitřního potenciálu ξ dle rovnice 1. Dále určení hodnoty aktivační funkce $\sigma(\xi)$ dle rovnice 2. Hodnota aktivační funkce je zároveň hodnotou výstupu y (rovnice 3).

Adaptivní fáze: V této fázi jsou perceptronu předkládány jednotlivé vzory (ve tvaru vstupy: x_1, x_2, výstup d), které chceme, aby síť dokázala rozpoznávat. Po každém vzoru je podle učícího pravidla vyhodnoceno, zda-li je vzor rozpoznán správně. Vzory lze síť učit opakovaně a libovolně dlouho. Učícím předpisem je původní *perceptronové učící pravidlo* od Marka Rosenblatta (viz. kapitola 3), jehož předpis je:

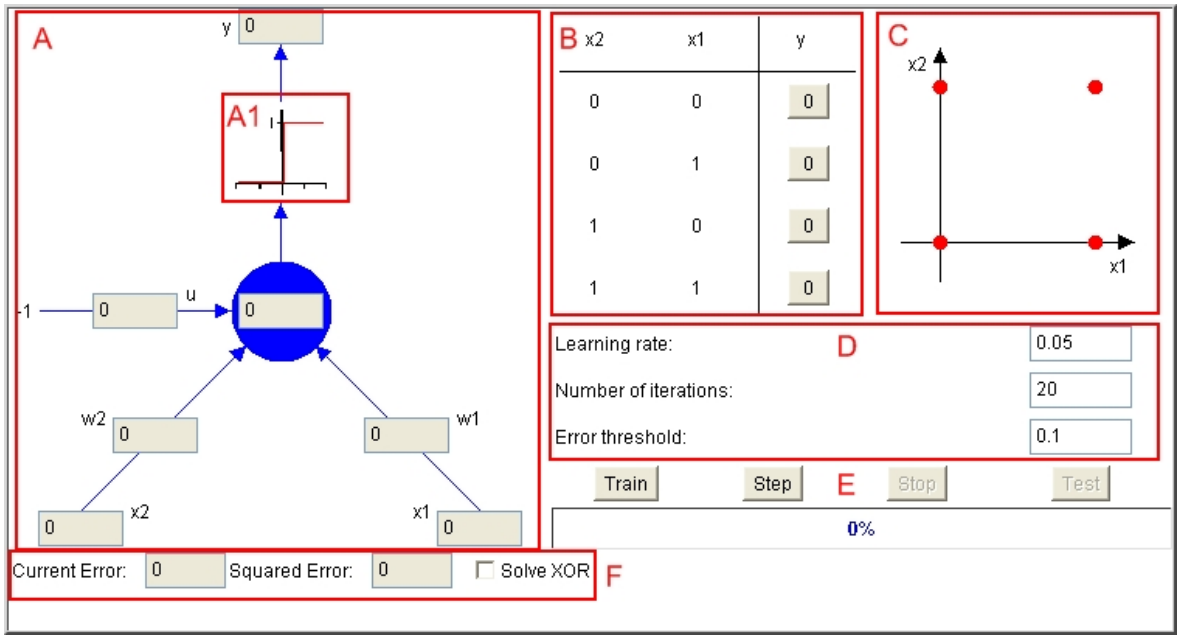
$$w_j(t+1) = w_j(t) + \eta(d-y)x_j \tag{4}$$

kde w_j označuje váhu j-tého vstupu neuronu ($j \in \{1, 2\}$). t si lze představit jako časovou jednotku s počátkem v 0 ($t \in N$). η je *rychlost učení* ($0 < \eta \leq 1$). d značí očekávaný výstup perceptronu (tzn. výstup právě zadaného vzoru), y skutečný výstup a konečně x_j označuje j-tý vstup právě zadaného vzoru.

Podstatou adaptivní fáze je fakt, že pokud je očekávaný výstup d shodný se skutečným výstupem y, celý člen $\eta(d-y)x_j = 0$ a váhy se neadaptují. Obdobně funguje i člen x_j, který zaručí, že se adaptují pouze ty váhy, které označují nenulové vstupy, tzn. pro $x_j \neq 0$.

Bližší popis JavaAppletu: Postupujme podle obrázku 3 na straně 5. *Část A* zobrazuje perceptron i všechny jeho vnitřní hodnoty v průběhu aktivní i adaptivní fáze. Navíc nabízí kliknutím na obrázek v *části A1* další možnosti aktivační funkce. *Část B* slouží k nastavení tréninkových vzorů. *Část C* s grafem, zobrazuje aktuální rozlišovací schopnosti sítě. *Část D* nabízí 3 parametry pro fázi učení, *learning rate* – odpovídající η v rovnici 4, *Number of interations* – kolikrát se má nechat naučit daný set 4 vzorů uvedených v části B, *Error threshold* – chyba sítě[6]. Adaptivní fáze končí dosažením zadaného počtu iterací nebo úrovně chyby. *Část E* nabízí tlačítka pro ovládání adaptivní fáze. A konečně *část F* zobrazuje aktuální chybu sítě. Jako bonus je zde políčko *Solve XOR*, které přidáním nového vstupu řeší logickou funkci XOR.

[6] Chyba sítě $E = \frac{1}{2}\sum_{i=1}^{p}(y^{(i)} - d^{(i)})^2$ pro p vzorů



Obrázek 3: JavaApplet 1 – perceptron – autor: Fred Corbett

6 Vícevrstvá síť - Backpropagation

WWW adresa: *http://is.muni.cz/www/139613/index.html*

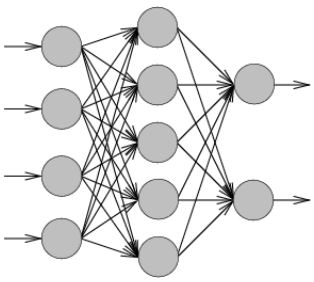
Autor: David Kabáth

Topologie: 3-vrstvá sít perceptronů (pro představu viz. obrázek 4). První vrstva je vstupní, druhá skrytá a třetí výstupní. Každý neuron z nižší vrstvy je spojen se všemi neurony vrstvy následující. Vstupní vrstva se skládá z 26 neuronů, počet skrytých neuronů je variabilní (standardně přednastavený na 40) a výstupní vrstva obsahuje 10 neuronů. Jako aktivační funkce je použita *standardní sigmoida* daná předpisem:

$$\sigma(\xi) = \frac{1}{1 + e^{(-\xi)}} \tag{5}$$

Aktivní fáze: Výpočet pro zadané vstupy $x_1, x_2, ..., x_{26}$ postupuje po jednotlivých vrstvách. V každé vrstvě probíhá výpočet postupně pro všechny neurony. Po zkompletování výstupů všech neuronů jedné vrsty se postupuje k vrstvě následující. Výstup poslední vrstvy $y_1, y_2, ..., y_{10}$ je výstupem celé sítě.

Adaptivní fáze: Jednotlivé vzory jsou ve tvaru $x_1, x_2, ..., x_{26}$, $y_1, y_2, ..., y_{10}$. Použitý učící alogoritmus se jmenuje *Backpropagation*. Pro vstupy $x_1, x_2, ..., x_{26}$ každého vzoru, jsou vypočteny reálné výstupy $y_1, y_2, ..., y_{10}$.



Obrázek 4: Vícevrstvá perceptronová síť

Poté dojde k výpočtu aktuální chyby sítě k množině vzorů:

$$E(\mathbf{w}) = \sum_{k=1}^{p} E_k(\mathbf{w}) \tag{6}$$

kde $\mathbf{w}$ značí aktuální konfiguraci vah sítě a p počet vzorů. Celková chyba sítě odpovídá chybám jednolitvých vzorů. Chyba k-tého vzoru odpovídá:

$$E_k(\mathbf{w}) = \frac{1}{2} \sum_{j \in Y} (y_j - d_j)^2 \tag{7}$$

kde Y je množina všech neuronů ve výstupní vrstvě. y značí reálné výstupy a d očekávané výstupy pro k-tý vzor.

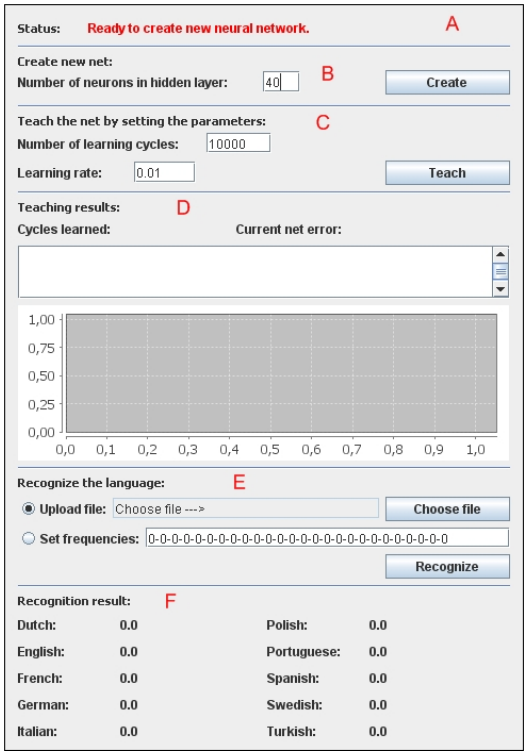
Minimalizace chyby a adaptace jednotlivých vah probíhá obdobně jako při použití perceptronového pravidla (viz. rovnice 4), nicméně poněkud složitěji, jelikož se jedná o vícevrstvou neuronovou síť. Pro každou váhu tedy platí:

$$w^{(t)} = w^{(t-1)} - \Delta w^{(t)} \tag{8}$$

kde $w^{(t)}$ označuje novou váhu na spoji mezi dvěma neurony a $w^{(t-1)}$ původní. Přírustek $\Delta w^{(t)}$ je počítán jako diferenciál funkce celkové chyby sítě vzhledem k dané váze. Pro podrobnější popis doporučujeme využít publikaci Šímy a Nerudy[8] na stranách 53–57.

Bližší popis JavaAppletu: Applet rozpoznává jazyk vloženého textu podle četnosti výskytu jednotlivých písmen. Dokáže rozpoznat 10 jazyků – nizozemštinu, angličtinu, francouzštinu, němčinu, italštinu, polštinu, portugalštinu, španělštinu, švédštinu a turečtinu. Jeden vzor sítě odpovídá jednomu jazyku a reprezentuje jej vektor 36 čísel. Prvních 26 označuje četnosti písmen abecedy *a-z*. Zbylých 10 očekávané výstupy. Každý výstup reprezentuje jeden jazyk, tzn. vektor vzoru je na posledních deseti pozicích reprezentován jednou 1 a devíti 0 (pozice 1 se pro dané vzory liší).

Applet je rozdělen do *částí A-F* (viz. obrázek 5). *Část A* informuje o právě probíhající činnosti. *Část B* slouží k vytvoření nové sítě. Modifikovatelným parametrem je počet neuronů skryté vrstvy. Následuje *část C* reprezentuje učící fázi. Parametry jsou počet učících cyklů a rychlost učení.

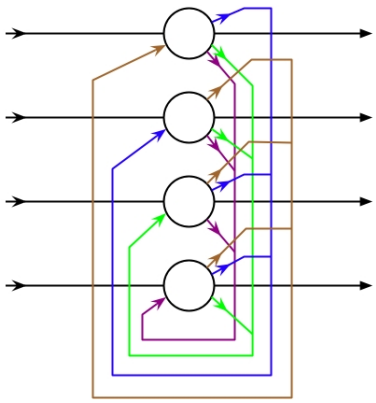

Status:
Ready to create new neural network.
A
Create new net:
Number of neurons in hidden layer:
40
B
Create
Teach the net by setting the parameters:
C
Number of learning cycles:
10000
Learning rate:
0.01
Teach
Teaching results:
D
Cycles learned:
Current net error:
1,00
0,75
0,50
0,25
0,00
0,0 0,1 0,2 0,3 0,4 0,5 0,6 0,7 0,8 0,9 1,0
Recognize the language:
E
Upload file:
Choose file --->
Choose file
Set frequencies:
0-0-0-

Výsledky učení jsou znázorněny v *části D. Část E* slouží k vložení relativních četností výskytu písmen. Lze jej vložit dvěmi způsoby. Buďto jako soubor, z kterého jsou jednotlivé četnosti vypočteny či přímo jako vektor 26 čísel. A konečně *část F* zobrazuje výsledky rozpoznání jazyka. Údaje u jednotlivých jazyků odpovídají výstupům neuronů ve výstupní vrstvě sítě. Neuron s nejvyšší hodnotou je vyhlášen za vítěze a jazyk, který reprezentuje, je zvýrazněn.

7 Hopfieldova síť

WWW: *http://www.cbu.edu/~pong/ai/hopfield/hopfieldapplet.html*

Autor: Kriangsiri Malasri



Obrázek 6: Hopfieldova síť

Topologie: Hopfieldova síť (viz. obrázek 6) se skládá z n neuronů, z nichž je každý zároveň vstupem i výstupem sítě. Zároveň jsou také všechny neurony každý s každým vzájemně propojené (žádný neuron ale není spojen sám se sebou).

Nyní budeme pokračovat adaptivní fází, jelikož narozdíl od perceptronových modelů probíhá pouze jednou. To znamená, že odpovídá spíše aktivní fázi perceptronu. Stejně tak aktivní fáze Hopfieldovy sítě odpovídá spíše adaptivní fázi perceptronu.

Adaptivní fáze: Řídí se *Hebbovým zákonem.* Opět máme tréninkové vzory, které jsou dány stavem neuronů sítě (tzn. jakou hodnotu by neurony pro rozpoznání daného vzoru měly mít). Vzor sítě o n neuronech je tedy vektor $x_1, x_2, ..., x_n$. Každý vzor je učen pouze jednou, a tím jsou stanoveny výsledné váhy sítě podle pravidla:

$$w_{ji} = \sum_{k=1}^{p} x_{kj} x_{ki} \quad 1 \leq j \neq i \leq n \tag{9}$$

kde p je počet vzorů a w_{ji} značí váhu mezi neurony i a j. x_{kj} a x_{ki} i-tý a j-tý neuron v k-tém vzoru.

Aktivní fáze: Na začátku aktivní fáze se stavy neuronů nastaví na vstup sítě. Fáze výpočtu probíhá po jednotlivých neuronech. Pro každý neuron je vypočten vnitřní potenciál:

$$\xi_j^{(t-1)} = \sum_{i=1}^{n} w_{ji} y_i^{(t-1)} \tag{10}$$

kde y_i značí výstup neuronu i.

Následně je za pomocí $\xi_j^{(t-1)}$ určen nový výstup y_j:

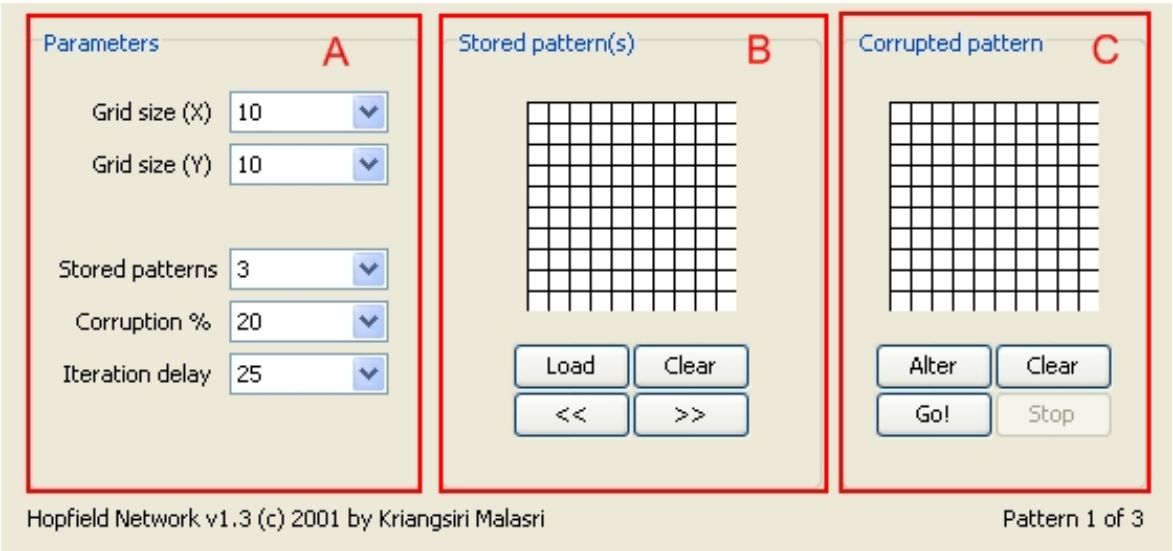
$$y_j^{(t)} = \begin{cases} 1, & \xi_j^{(t-1)} > 0 \\ y_j^{(t-1)} & \xi_j^{(t-1)} = 0 \\ -1, & \xi_j^{(t-1)} < 0 \end{cases} \tag{11}$$

Z výše uvedeného je vidět, že výsledné stavy jsou bipolární (obsahují pouze hodnoty 1 a -1), což má blíže k biologické předloze umělých neuronů. Tímto způsobem je výpočet aktivní fáze opakován tak dlouho, dokud se stavy neuronů mění.

Pro podrobnější popis a možná rozšíření Hopfieldovy sítě opět odkazujeme na Šímu a Nerudu[8], strany 73–93.

Bližší popis JavaAppletu: Applet sám o sobě reprezentuje nejzákladnější schopnost Hopfieldovy sítě – schopnost *asociativní paměti*. Podle vložených vzorů dokáže opravovat zadané obrazce. To je užitečné například při čištění šumu z obrázků.

Ovládání appletu je víceméně intuitivní, nicméně jej zběžně můžeme projít. Rozdělme jej na 3 části. *Část A* slouží k nastavení parametrů sítě. Lze zde nastavit několik parametrů. Počet polí (neuronů) – *Grid size (X/Y)*, počet uložených vzorů – *Stored patterns*, procentuální poškození při náhodném generování obrazců – *Corruption %*, pauzu mezi dvěmi iteracemi aktivní fáze sítě – *Iteration delay*. V *části B* lze zobrazovat jednolivé vzory, připadně i vkládat nové. A nakonec *část C* obsahuje prostor pro vkládání vstupu sítě – poškozených obrazců, které mají být sítí rozpoznány. Funkce jednotlivých tlačítek by měly být intuitivní, zmiňme se jen o funkci tlačítka *Alter*. To slouží k vygenerování náhodného obrazce s procentuálním poškozením zadaným v části A.



Obrázek 7: JavaApplet 3 – Hopfieldova síť – autor: Kriangsiri Malasri

8 Závěr

Neuronové sítě jsou velmi rozsáhlý a silný nástroj. Cílem této práce bylo přiblížit jejich teorii a prezentovat základní modely na reálných případech. Konkrétně se jednalo o *perceptron*, *vícevrstvou síť* a *Hopfieldovu síť*. Pro další studium doporučujeme publikaci Jiřího Šímy a Romana Nerudy *Teoretické otázky neuronových sítí*[8] dostupnou online na adrese: *http://www2.cs.cas.cz/~sima/kniha.html*

Reference

[1] McCulloch, Warren S., Pitts, Walter: *A logical calculus of the ideas immanent in nervous activity.* Bulletin of mathematical biophysics, svazek 5, 1943, s. 115–133.

[2] Hebb, Donald O.: *The Organization of behavior: A neuropsychological theory.* New York, Wiley, 1949.

[3] Rosenblatt, Frank: *A probabilistic model for information storage and organization in the brain.* Psychological review, svazek 65, 1958, s. 386–408.

[4] Minsky, Marvin L., Papert, Seymour: *Perceptrons.* Cambridge MA, MIT Press, 1969.

[5] Hopfield J. John: *Neural networks and physical systems with emergent collective computational abilities.* Proceedings of the national academy of sciences, svazek 79, 1982, s. 2554–2558.

[6] Rumelhart, David E., Hilton, Geoffrey E., Williams, Ronald J.: *Learning internal representations by error propagation.* Parallel distributed processsing: explorations in the microstructure of cognition, svazek 1, Cambridge MA, MIT Press, 1986, s. 318–362.

[7] Bryson, Arthur E., Ho, Yu-Chi: *Applied optimal control.* New York, Blaisdell, 1969.

[8] Šíma, Jiří, Neruda, Roman: *Teoretické otázky neuronových sítí.* Prague, MATFYZPRESS, 1996.