UMĚLÉ NEURONOVÉ SÍTĚ A ASOCIACE

Náležejí k učicím se algoritmům, které pracují s rozsáhlými soubory podobných jednoduchých atributů. Tyto algoritmy zahrnují:

- individuální neuron;
- perceptrony ve vrstvách a algoritmus zpětného šíření chyb (back-propagation);
- WISARD;
- Boltzmanovy stroje a žíhání (simulované);
- Kohonenovy mapy;
- reprodukce z jednoho rodiče s mutací;
- genetické algoritmy.

Tyto algoritmy mají velmi různé vlastnosti. Jejich chování záleží na:

- prostoru, který prohledávají;
- stupni heuristického řízení.

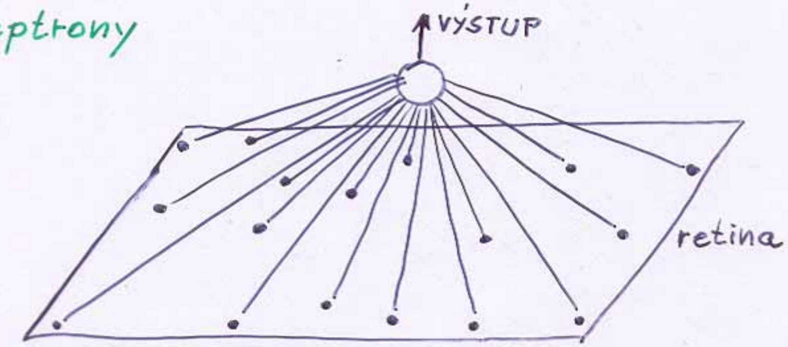
Vstupem pro uvedené algoritmy jsou hodnoty velmi vysokého počtu atributů. Všechny atributy jsou téhož typu. Atributy se liší např. pozicí apod., což nemá žádný vliv na jejich možné hodnoty. Každá hodnota je velmi jednoduchá: typicky 1 nebo 0, či ANO/NE, případně reálné číslo.

Prototypem je vstup nervového systému lidského oka. Vstup přichází z retiny (sítnice), která se skládá z vysokého počtu buněk aktivovaných světlem. Za retinou je kaskáda jiných buněk - neuronů -, které jsou fyziologicky pouze několika druhů. Každý neuron je schopen přijímat impulsy z jiných buněk a každý může svým aktivovaným výstupem stimulovat další neurony, pokud je svými vstupy aktivován.

Základní výpočtové modely

- jednoduchý perceptron: jeho výstup závisí na váženém součtu vstupů. Možnosti má omezené.
- mnohovrstvé perceptrony: jsou tvořeny z jednoduchých perceptronů.
- WISARD a logické neuronové sítě připomínají mnohovrstvé perceptrony (strukturou i funkcí), avšak jednotlivé uzly jsou tvořeny jednobitovými pamětmi s libovolným přístupem (RAM).
- Boltzmanův stroj je důmyslnější, ale pomalý.
- Kohonenovy sítě jsou navrženy ke hledání klasifikací.
- Genetické algoritmy nezahrnují nic z neuronů, ale náležejí sem, protože pracují se stejnými typy vstupních dat nízké úrovně (nestrukturovaná data).

Perceptrony



Jednoduchým perceptronem se většinou v literatuře nyní rozumí model jednotlivého neuronu. Jeden uzel má mnoho vstupů a jeden výstup. Je to jednoduchá výpočetní jednotka. Typy neuronů se mohou lišit formou vstupů a typem závislosti výstupu na vstupech.

Nejjednodušší typ rozeznává na vstupu dvě možné hodnoty, "1" a "0" či "ano" a "ne". Je-li na vstupu "1", neuron je aktivován. Každý vstup má přiřazenu nějakou váhu (reálné číslo). Dále má uzel tzv. práh (reálné číslo). Výstupem uzlu je

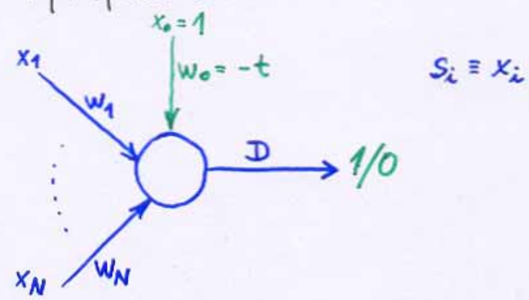
1 pokud $\sum_{i=1}^{N} s_i w_i > t$ (s_i = vstup, w_i = váha, t = práh)

0 v ostatních případech.

Znalost je v neuronu zakódována pomocí vah a prahu. Práh je redundantní a lze jej nahradit dalším vstupem s_0 jehož hodnota je vždy = 1 a jehož příslušná váha $w_0 = -t$:

1 pro $\sum_{i=0}^{N} s_i w_i = D > 0$

0 v ostatních případech.



Kterýkoliv perceptron rozeznává určitou třídu všech svých možných vstupů, C:

$$\{(x_0, x_1, \dots, x_N) \mid D>0 \wedge x_j \in \{0,1\}\ \forall j \le N\}$$

Je to třída retinálních obrazů, která perceptron aktivuje.

Uzel se učí změnou svých vah w_i. Vstupem při učení je (teoreticky) nekonečná sekvence obrazů:

$$I_1, I_2, I_3, \dots$$

přičemž každý vstupní obraz je doplněn informací:

ano je-li ve třídě C, kterou má uzel umět rozeznat, a
ne pokud obraz do třídy C nepatří.

Předpokládejme, že uzel je ve druhém stavu, tj. jeho práh je 0. Algoritmus učení je následovný:

REPEAT pro každé vstupní I_j
  IF $D(I_j) > 0$ ale I_j nepatří do C
    THEN nahraď každé w_m hodnotou $w_n - x_n(I_j)$;
  IF $D(I_j) < 0$ ale I_j patří do C
    THEN nahraď každé w_n hodnotou $w_n + x_n(I_j)$.

První "IF" koriguje všechny váhy, které jsou příliš vysoké a tím nutí uzel rozeznat obraz, nepatřící však do C. Druhý "IF" naopak koriguje příliš nízké váhy. Kdykoliv uzel rozezná klasifikovaný obrazec správně, váhy se nezmění.

Uvedený algoritmus má tu příznivou vlastnost, že splňuje konvergenční teorém:

Předpokládejme, že existují váhy $w_0^*, w_1^*, w_2^*, \dots, w_N^*$ takové, že součet

$$x_0 w_0^* + x_1 w_1^* + x_2 w_2^* + \dots + x_N w_N^* > 0$$

kdykoliv vstup I patří do třídy C a tento součet < 0 kdykoliv vstup I nepatří do C.

Pak platí, že pro jakékoliv počáteční váhy w_i a pro jakoukoliv trénovací posloupnost I_j existuje nějaké číslo k takové, že perceptron správně klasifikuje I_j pro všechna $j > k$.

Znamená to, že pokud libovolný uzel je schopen rozpoznat koncept C, pak po určitém omezeném čase uzel ukončí své učení. Teorém ovšem nic neříká o tom, kdy poznáme, že požadovaného stavu bylo dosaženo.

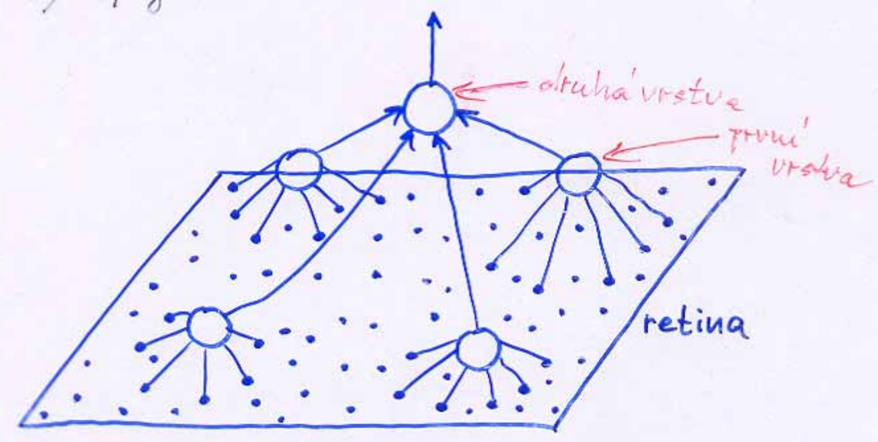
Teorém je nutno používat obezřetně: každý uzel má totiž pouze omezenou (i když třeba velikou) množinu vstupů x_i, a každý vstup může nabývat jen 0 nebo 1. Proto existuje pouze omezený soubor možných instancí I, i když posloupnost I_j může být neomezená. Důsledek je ten, že po nějakém čase se už nemohou vyskytnout nové (odlišné) instance, na vstup přicházejí hodnoty opakované.

Vícevrstvé perceptrony

Nejjednodušší typ perceptronu je tvořen jediným uzlem. Nejjednodušší vrstvený typ se skládá ze dvou vrstev neuronů: Spodní vrstva obsahuje mnoho uzlů, jejichž vstupy jsou přímo spojeny s retinou. Jedna buňka retiny zároveň může tvořit vstup pro více uzlů. Druhá vrstva se skládá z jediného neuronu, jehož vstupy jsou výstupy všech uzlů první vrstvy.

Výstupem dvouvrstvého perceptronu je výstup uzlu druhé vrstvy. Hodnota výstupu má být 1 pokud je na retině obrazec určitého druhu, např. 1 pokud obrazec je konvexní, jinak 0.

Perceptron složený z jediného uzlu je v podstatě pouze sečítačka. Dvouvrstvý perceptron umožňuje rozeznat konvexní oblast na retině. Trojvrstvý perceptron umí rozeznat libovolné mnohoúhelníkovité oblasti, které nemusí ani být konvexní a dokonce nemusí být spojené.



Dvouvrstvý perceptron se 4 uzly ve spodní vrstvě.

Předem není jasné, které třídy mohou být nějakým perceptronem rozeznány. Jednotlivý perceptron je omezen. Žádný jediný uzel neumí rozeznat třídu obrazců obsahujících v retině pouze jedinou tečku.

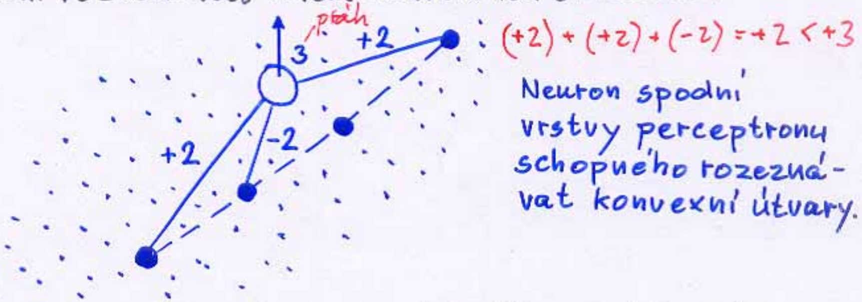
Pro každou konečnou třídu C lze vytvořit dvouvrstvou strukturu, která ji rozezná.

Existují některé jednoduché a zajímavé omezené formy vrstvových perceptronů. Kupř. každému uzlu první vrstvy lze přiřadit podmínky. O perceptronu lze říci, že je

- **řádově omezený** s řádem k pokud každý z uzlů první vrstvy má nanejvýš k vstupů.
- **diametricky omezený** s limitem d pokud každý z uzlů první vrstvy může mít libovolný počet vstupů, avšak všechny vstupní buňky retiny pro daný uzel leží v okruhu o diametru (průměru) d.

Příklad: Rozeznávání konvexních oblastí.

Mějme dvouvrstvý řádově omezený perceptron s omezením $k=3$ jenž umí rozeznávat třídu konvexních obrazců:



Neuron spodní vrstvy perceptronu schopného rozeznávat konvexní útvary.

Obrazec je konvexní, pokud každá úsečka s oběma konci v obrazci je ve skutečnosti celá v tomto obrazci.

Horní vrstva perceptronu obsahuje právě 1 uzel se vstupem z každého uzlu spodní vrstvy. Perceptron je aktivován vždy když žádný z uzlů spodní vrstvy aktivní není.

Proto je např. možné všechny váhy jeho vstupů nastavit na -2 a jeho práh na -1.

Ve spodní vrstvě je jeden uzel pro každou množinu tří různých kolineárních bodů retiny. Nastavení vah ukazuje předchozí obrázek $(+2, -2, +2)$. Práh = 3. Neuron je aktivován, pokud oba konce jsou "osvětleny" avšak prostřední nikoliv.

Uvedené jednoduché zařízení může být rozsáhlé. Je-li retina čtvercová s n buňkami podél každé strany, pak existuje $n(n-1)/2$ dvojic, které mohou být konci úsečky. Na každé takové úsečce je v průměru asi $c \times n$ možných prostředních bodů (c je číslo nezávislé na n). Počet uzlů spodní vrstvy je tedy řádu $K = O(n^3)$.

Další formy perceptronů

Je možné připustit i $0 \le x_j \le 1$, což umožňuje rozeznávat nejen siluety (0=bílá, 1=černá), ale i stupně šedi.

Dále lze používat i perceptrony, kde hodnota výstupu rovněž nabývá hodnotu reálného čísla mezi 0.0 a 1.0. Existuje obecný model zvaný sigmoidální nelinearita, kde výstup sigmoidálního perceptronu je

$$f(D) = \frac{e^D}{e^D + 1}$$

kde $0 < f(D) < 1$ pro reálné hodnoty D. Tato funkce má také výhodnou vlastnost:

$$f(-D) = 1 - f(D)$$

Sigmoidální perceptron rozeznává třídu vstupů, pro něž platí:

$$f(D) > 0.5$$

Je ovšem možné použít i jiné funkce pro ovlivnění výstupu. Taková funkce $f(D)$ by měla splňovat podmínky:
- monotónně rostoucí
- (téměř) nulová pro velké záporné hodnoty D
- (téměř) jednotková pro velké kladné hodnoty D

Často užívanou metodou učení vícevrstvých neuronových sítí se sigmoidními perceptrony je tzv. metoda zpětného šíření chyby. Váhy jsou (zpětně) upravovány poté, co byla zjištěna míra chyby na výstupu:

$$w_j \leftarrow w_j + \underline{K} \cdot \Delta \cdot x_j \quad \text{kde}$$

w_j ... hodnota váhy j-tého vstupu uzlu
K ... kladná konstanta zvaná též "zisk"
x_j ... j-tý vstup daného uzlu
Δ ... míra chyby na výstupu uzlu

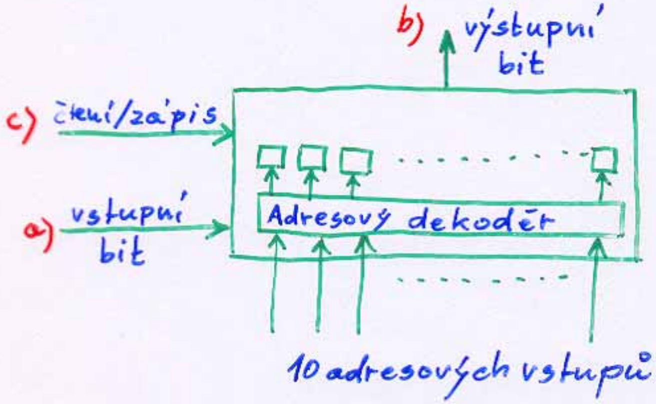
Váhy se upravují zpětně od nejvyšší vrstvy směrem k vrstvě nejnižší. V principu jde o minimalizaci střední kvadratické odchylky všech výstupů uzlů.
Konstanta K řídí rychlost konvergence: pro velké K algoritmus "skočí" do řešení rychle, ale může při tom "skočit" příliš daleko, tj. již za optimum vah.
Není zde jistota, že budou nalezeny optimální váhy a není známo, kdy skončit, ale v praxi funguje algoritmus dobře.

Logické neuronové sítě

jsou zařízení sestavená z individuálních uzlů spojených do sítě. Vstup, výstup a vnitřní struktura jsou obdobou vícevrstvých perceptronů, avšak individuální uzly a jejich funkce jsou zcela odlišné.

Každý uzel je paměťové zařízení (něco jako RAM čip). Neprovádí žádnou aritmetiku. Vstupem jsou adresové přívody, které mohou být 0 nebo 1. Má-li čip N adresových vstupů, pak obsahuje 2^N paměťových buněk, z nichž každá může uchovávat "0" nebo "1". (Pro $N=10$ jde o čip 1Kb paměti).

Objeví-li se na vstupu určitý bitový obrazec, je považován za adresu jedné z paměťových buněk. RAM čip má dále 3 další přívody: (a) vstupní datový, (b) výstupní datový, (c) "čtení/zápis" řídící (rozhoduje, zda se jedná o zápis do- nebo o čtení z paměti). Pro "čtení" je okopírován obsah adresované buňky na výstup, pro "zápis" je přepsán obsah na dané adrese hodnotou na vstupním přívodu:

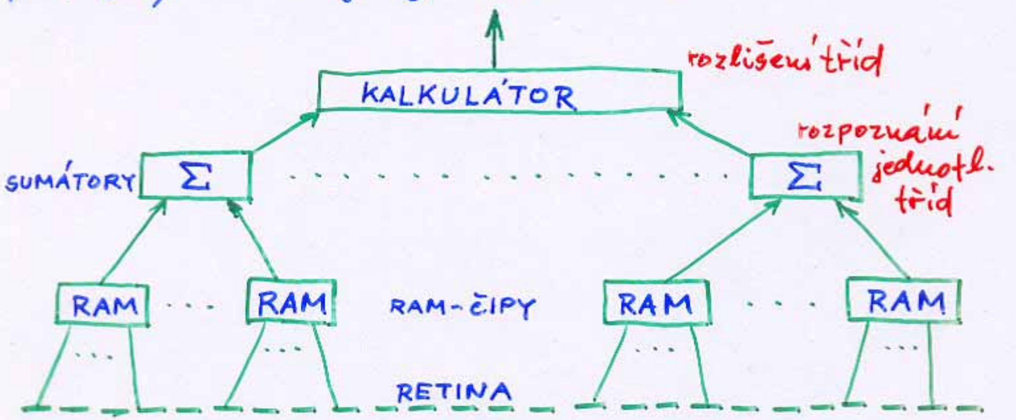


I když se zdá, že pro absenci výpočtů není zařízení příliš užitečné, není tomu tak.

WISARD

je název zařízení navrženého a sestaveného týmem vedeným Alexanderem: WIlkie, Stonham, and Alexander's Recognition Device. Některé z uzlů ovšem nějaké výpočty provádějí, ale učící se část je postavena pouze z čipů RAM.

Vstupy z WISARDu jsou také z retiny a mohou nabývat hodnot 0/1. WISARD se skládá ze tří vrstev a umí 3 operace: naučit se rozeznávat třídu obrazců, rozeznat, zda nová instance je v naučené třídě, a dále rozlišovat mezi třídami. Tyto 3 operace jsou prováděny v následných vrstvách. Celé zařízení je zapojeno ve stromové struktuře. Horním uzlem je "kalkulátor" jenž rozlišuje mezi třídami. Vstupem jsou podstromy schopné rozeznat jednotlivou třídu, přičemž horním uzlem je sumátor. Na nejnižší úrovni (listy stromu) jsou čipy RAM, jejichž vstupy jsou připojeny na zdroj signálu (retinu):



Učení WISARDu.

Každý "rozpoznávač" se učí rozeznat právě jednu třídu. Je trénován separátně za použití svého vlastního tréninkového souboru. Tréninkové soubory potřebují pouze pozitivní příklady rozpoznávačovy třídy, ovšem bez šumu a chyb. Učící procedura:

- na počátku nastav všechny RAM čipy na nulu
- obrazec je promítnut na retinu a každý RAM čip používá své vstupy jako adresy svých paměťových buněk; je nastaven režim "zápis".

Rozeznávání

- obrazec (který má být klasifikován) je promítnut na retinu (jako při učení). Čip je nastaven na režim "čtení". Na výstup je zkopírován obsah paměťové buňky adresované obrazcem, takže "1" se objeví v případě, kdy je na retině obrazec shodující se s naučeným; jinak se objeví "0".
- Výstupy RAM-čipů každého rozpoznávače jsou přivedeny na vstup sumátoru, jehož výstupem je prostý počet jedniček na jeho vstupu.

Předpokládejme, že rozpoznávač obsahuje K RAM-čipů:

- je-li rozeznávaný obrazec přesně shodný s trénovací instancí, pak všechny RAM-čipy budou na výstupu mít "1" a na výstupu sumátoru bude hodnota K.

- je-li obrazec zcela nepodoben jakékoliv trénovací instanci, pak hodnoty N vstupních adresových přívodů každého RAM-čipu nebudou mít korelaci s žádnou hodnotou, na niž byly čipy trénovány. Buňka adresovaná v každém čipu bude vybrána náhodně a protože původní obsah byl na začátku nastaven na "0", veliká většina RAM-čipů bude mít na výstupu "0". Hodnota na výstupu sumátoru bude << K.
- je-li předkládaný obrazec podobný některé trénovací instanci, pak je dobrá šance, že vstupy některého z RAM-čipů se shodnou se známým obrazcem. Takové RAM-čipy dají "1" na výstupu. Obrazec připomínající trénovací instance ve většině hledisek způsobí, že většina RAM-čipů dá "1" a sečítačka poskytne číslo blízké K.

Popsaným způsobem tedy rozpoznávací zařízení umožní generalizaci z trénovacího souboru. Generalizovaná třída rozeznaných příkladů se skládá z těch, které poskytují na výstupu sumátoru čísla blízká ke K.

Tyto třídy jsou do určité míry "fuzzy", tj. neostré. Je-li suma přesně = K, pak obrazec zcela jistě patří do dané třídy. Menší suma, k < K, znamená, že klasifikovaný příklad je pravděpodobně z dané třídy pokud k je blízké K. S klesajícím k klesá postupně i míra příslušnosti příkladu do dané třídy.

Rozlišování mezi různými třídami

WISARD kombinuje několik rozpoznávačů. Jednotlivé rozpoznávače umí rozeznávat jen jednu třídu.

Horní vrstva (kalkulátor) má jako vstupy hodnoty ze sumátorů. Vypočítává hodnotu poměru

$$R = \frac{k}{K}$$

pro všechny sumátory. Dále je schopen zjistit nejvyšší poměr.

Výstup celého zařízení WISARD se skládá z:

- názvu třídy odpovídající největšímu poměru R;
- absolutní věrohodnosti R, že příklad náleží do dané třídy;
- relativní věrohodnosti, že příklad je v této třídě spíše než v další nejbližší. Je-li poměr dané třídy R a jí nejbližší třídy R', pak relativní věrohodnost r se vypočítá jako

$$r = \frac{R - R'}{R} \quad \left(= 1 - \frac{R'}{R} \right)$$

Když jsou k dispozici kvalitní trénovací data, má WISARD některé výhody vůči vícevrstvým perceptronům. Jako perceptrony, umí WISARD rozeznávat a rozlišovat třídy a generalizovat z jejich trénovacích množin. Poskytuje dále míru věrohodnosti své klasifikace, což většina perceptronů neumožňuje. Citlivost metody je omezena pouze množstvím RAM-čipů. S jejich dostatečným počtem a dostatečně rozsáhlými trénovacími množinami je v principu možné rozeznávat libovolnou třídu obrazců s absolutní věrohodností (tj. s rostoucí pamětí roste přesnost klasifikace).

Důležitou vlastností je i to, že učení je velmi rychlé. Během fáze učení je každá instance předkládána jen jednou a učící se proces sestává pouze z jediného zápisu do každého RAM-čipu. To může trvat necelou mikrosekundu!

Navrhování logických sítí

Na rozdíl od moderních počítačů, kde je levnější používat velké RAM-čipy s mnoha adresovými vodiči, menší RAM-čipy pracují v logických sítích lépe. Během každého učení je poměr buněk čipu, které by měly být nastaveny na "1", pravděpodobně nezávislý na počtu buněk čipu, takže počet učících cyklů je zhruba úměrný rozměru čipu. Má-li každý RAM-čip N vstupů (adresových), pak obsahuje 2^N buněk. Z toho pak plyne, že čas potřebný k natrénování zhruba roste exponenciálně s N.

Další výhodou menších čipů s méně adresovými vstupy je to, že každá buňka ukládá pouze velmi hrubý fragment znalosti o obrazci na retině. V logických sítích závisí generalizace na šanci přesné shody mezi nějakou částí retinového obrazu a trénovací instance. Čím více bitů zkoumaných čipem, tím přesnější bude shoda, zároveň bude ovšem menší poměr shody mezi všemi zkoumanými příznaky. Proto bude schopnost generalizace lepší pro menší N.

Učení založené na soutěžení

Většina neuronových sítí je trénována buď tzv. s pomocí instruktora (učitele) nebo pomocí nějaké metody stupňování výkonnosti. Znamená to, že je síti poskytnuta požadovaná odpověď nebo dostane k dispozici (pomocí nějaké zpětné vazby) informace o úrovni své výkonnosti.

Existuje třída neuronových sítí, které však žádnou zpětnou vazbu nepotřebují. Tyto sítě se nazývají jako samoorganizující, protože sílu spojení si modifikují samy na základě předkládaných příkladů.

Kohonenovy mapy

Tyto struktury lze použít jak samy o sobě, nebo je lze začlenit jako vrstvu do složitější architektury. Sama o sobě tvoří Kohonenova mapa jeden z nejjednodušších samoorganizujících se systémů. Mapy sestávají ze dvou vrstev, přičemž vstupní vrstva je plně propojena s tzv. Kohonenovou vrstvou.

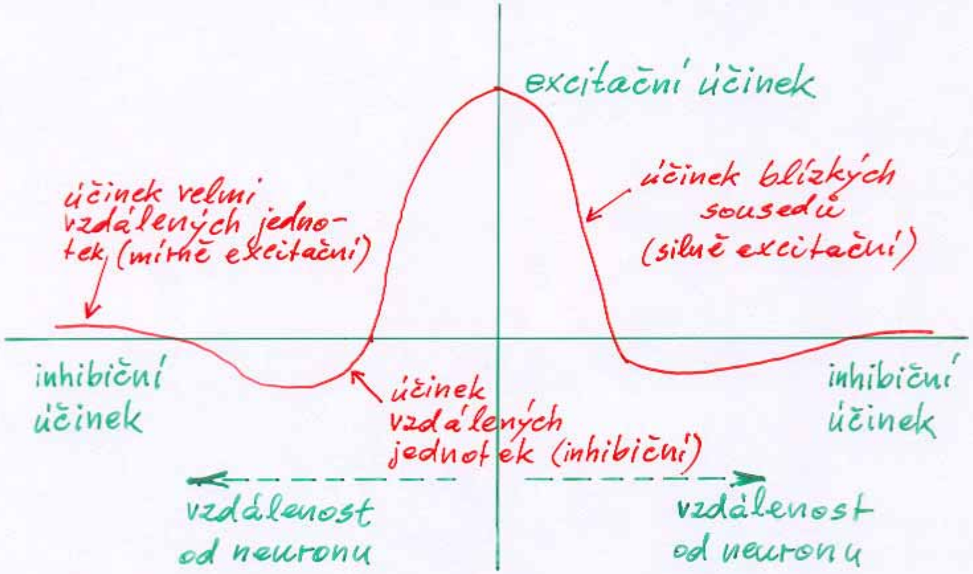
Vstupní vrstva působí jako distributor vstupů na všechny neurony Kohonenovy vrstvy.

Kohonenova vrstva působí jako výstupní vrstva a své výstupy předává okolí. Kromě propojení se vstupními jednotkami existuje v Kohonenově vrstvě velké množství spojů navzájem spojujících neurony. Tyto spoje tvoří kritickou část samoorganizující se mapy.

Podobně jako ostatní druhy samoorganizujících se sítí, i Kohonenovy mapy modelují vlastnost mozku zvanou laterální inhibice:

Předpokládejme např., že existuje Kohonenova síť tvořící strukturu o velikosti 10 × 10 jednotek. Neurony v této síti mají propojení „pod sebe" a „nad sebe", avšak navíc existují ještě určitá vzájemná propojení. Tyto spoje mají tendenci být pozitivně váhovány (excitační), když vedou k jednotkám v bezprostřední blízkosti, a negativně váhovány (inhibiční), když vedou ke vzdáleným jednotkám.

Síla propojení se mění obecně nepřímo úměrně vzdálenosti mezi neurony (mezi sousedy jsou spoje nejsilnější). Závislost mezi silou propojení a vzdáleností se často vyjadřuje křivkou zvanou „mexický klobouk":



Význam interních spojení spočívá ve vytvoření jakési soutěže mezi neurony Kohonenovy vrstvy:

Když se neuronům poskytne např. nějaký vstupní obrazec (prostřednictvím vstupní vrstvy), každá jednotka obdrží kompletní kopii vstupního vzoru, ačkoliv je vstupní signál modifikován váhami spojů mezi vstupy a Kohonenovou vrstvou. Měnící se odezva K. vrstvy vytváří soutěž, která se šíří napříč K. vrstvou.

Smyslem soutěže je stanovit, který neuron poskytuje nejsilnější odezvu na předložený vzor. V důsledku se každý neuron snaží zvýšit hodnotu výstupu svých bezprostředních sousedů a zároveň potlačit aktivitu zbývajících jednotek ve vrstvě. Bylo dokázáno, že takovýto systém se vždy stabilizuje, takže jediný neuron s nejvyšší celkovou hodnotou výstupu (jako reakce na vstup) předává svůj signál dále (tzv. vítěz); aktivita zbylých neuronů je potlačena (snížena).

Je pozoruhodné, že tato laterální inhibice (tj. postranní či příčné potlačení) funguje jako metoda pomocí níž si vrstva sama stanoví, který neuron má nejvyšší odezvu na předložený vstup. Není zapotřebí mít přehled o všech neuronech. Postačuje pouze znalost, jak každý neuron reaguje na svou lokální stimulaci. Každá jednotka funguje nezávisle, bez kontroly, která by měřila výkonnost vzhledem k nějaké maximální hodnotě ve vrstvě.

U biologických systémů byly obdobně účinkující laterální inhibiční schémata objevena v několika podsystémech mozku, např. u vizuálního systému.

Implementace laterální inhibice je náročná kvůli složitosti propojení ve vrstvě. Simulátor lze jednoduše realizovat aplikací funkce **maximum**: Jakmile je jednou Kohonenova vrstva stabilizována, a je znám vítěz, výstup vrstvy je jednoduše binární +1 od vítěze a nic od ostatních. V důsledku representuje vítěz kategorii, do níž náleží vstupní obrazec.

Pro trénování sítě je klíčem stanovení vítěze. Na rozdíl od ostatních typů neuronových sítí, Kohonenova mapa modifikuje váhy pouze u vítěze a jeho blízkého okolí (za předpokladu, že vzdálenost je dána jako parametr). Ostatní jednotky se neučí.

Trénovací pravidlo:

$$\Delta w_i = \beta \left(x_i - w_i^{(stará)} \right) \qquad 0.0 \le \beta \le 1.0$$

Zde β je tzv. učící konstanta (někdy zvaná jako „zisk"), x_i je hodnota vstupního signálu i-tého váhovaného spoje. Obvykle $\beta < 0.2$. Uvedené trénovací pravidlo je velmi jednoduché.

Soubor vah pro daný neuron lze považovat za složky n-rozměrného vektoru vah a odpovídající vstupní signály jako složky n-rozměrného vstupního vektoru. Kohonenovo učící pravidlo pouze pohybuje vektorem vah tak, aby se lépe vyrovnal se vstupním vektorem $\vec{x}$.

Jinými slovy, vítěz je ten neuron, jehož vektor je nejbližší vstupnímu vektoru a výsledkem jednoho trénovacího kroku je „popostrčení" váhového vektoru ke vstupnímu vektoru, přičemž velikost tohoto „popostrčení" závisí na zisku β. Kromě toho vítězovi sousedé (fyzicky nejbližší) rovněž upraví své váhy pomocí téhož pravidla pro trénování.

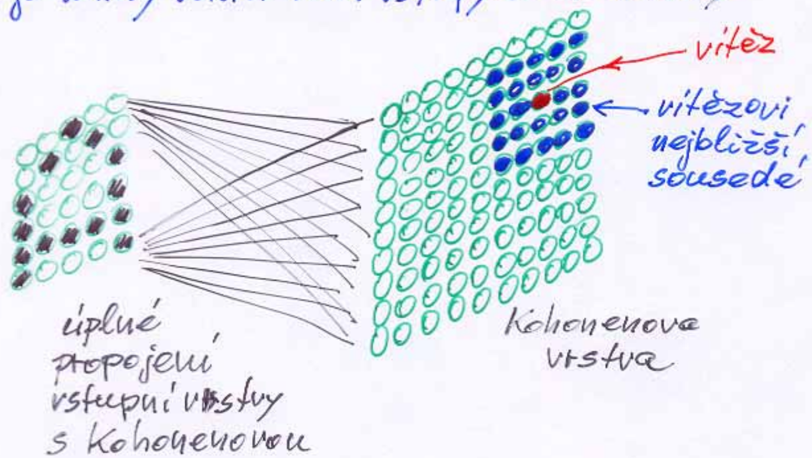
Jednoduchá verze Kohonenovy mapy:

$$y = \begin{cases} 1 & \text{když } I = \sum_{i=1}^{n} w_i x_i \text{ je největší ve vrstvě,} \\ 0 & \text{v ostatních případech} \end{cases}$$

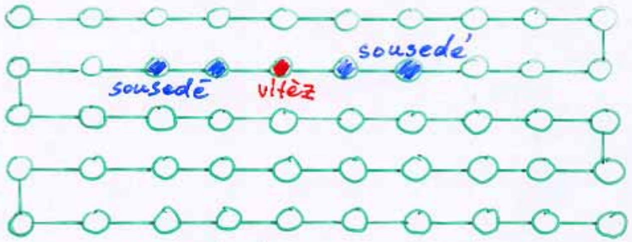
$\Delta w_i = \beta \left(x_i - w_i^{(stará)} \right)$ obvykle $0.0 \le \beta \le 0.25$

$\vec{x}$ je vektor vstupů

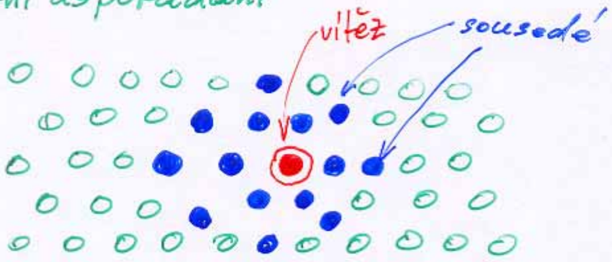
$\vec{w}$ je váhový vektor mezi vstupy a K. neurony



Fyzické sousedství vítěze je složeno z neuronů, které jsou ve fyzické blízkosti vítěze. Co to přesně znamená, záleží na tom, jak je síť navržena. Existují různé varianty:



Lineární uspořádání



Hexagonální uspořádání

V lineárním uspořádání má každý neuron jednoho souseda na každé straně. Sousedství vítěze může sestávat z neuronů na 2 či 3 pozicích na každé straně. Obvyklejší bývá šestiúhelníkovité uspořádání. Možné je jakékoliv uspořádání, které se hodí.

Obecně lze říci, že trénování Kohonenovy vrstvy začíná s poměrně rozsáhlým sousedstvím, které je postupně zmenšováno, jak trénování postupuje. Podobně zisk β je zpočátku velký a postupně je snižován.

Figure 11.4 The self-organizing network at the start of the iterations.

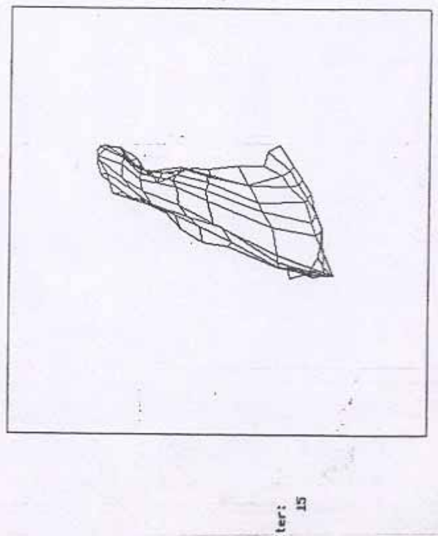


Figure 11.5 The self-organizing network as it begins to unravel itself.

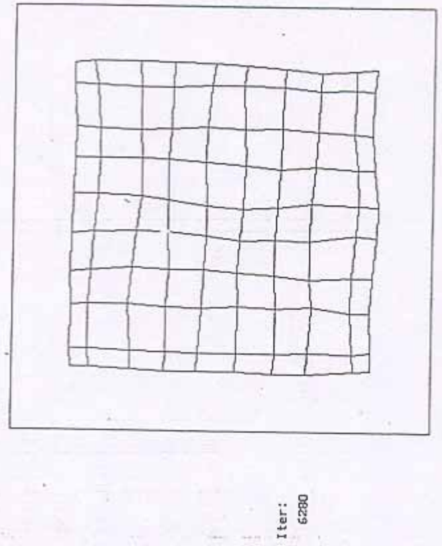


Figure 11.6 The 10 × 10 self-organizing network trained with uniformly distributed random input.

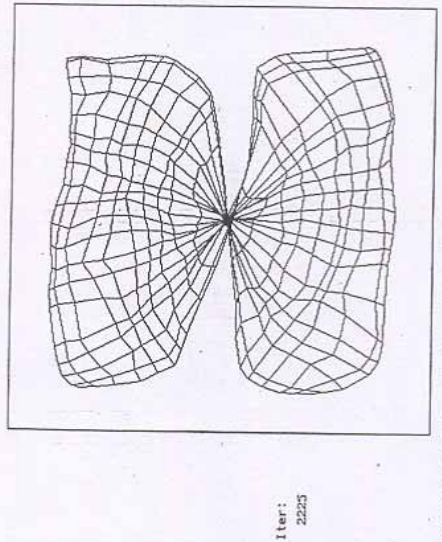


Figure 11.7 Self-organization with a twist. Too small an adapting neighborhood size caused this network to tie itself in a knot.