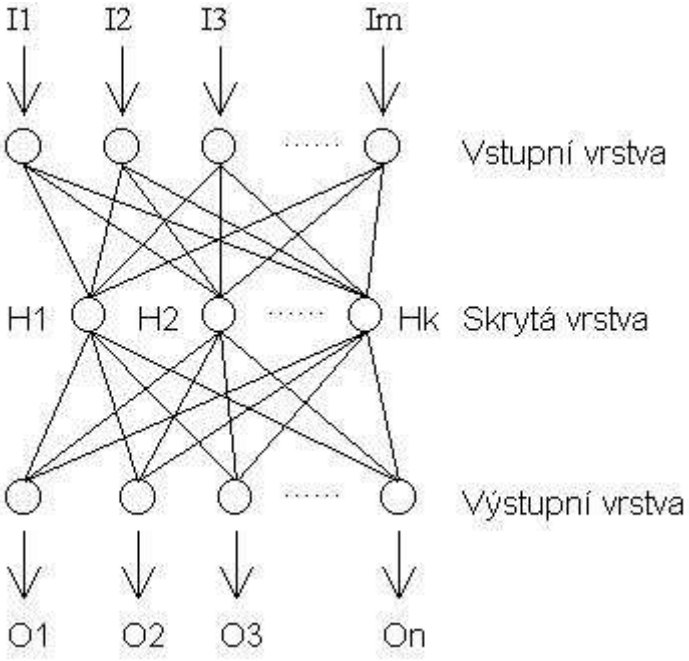
Tomáš Janík
172787

Jednotlivé historické modely neuronových sítí

Vícevrstevná perceptronová síť – opakování

Teoretický model obsahuje tři vrstvy perceptronů; každý neuron první vrstvy je spojen s každým neuronem z vrstvy druhé, obdobně mezi druhou a třetí vrstvou. Síť je dopředná – signál je posílán každým spojením (synapsí) z první vrstvy, druhá vrstva signály „sbírá“ a vyhodnocuje aktivační funkcí a opět posílá signál vrstvě třetí. Neuronům v první vrstvě se říká vstupní, neuronům v třetí výstupní.



Protože síť perceptronů je nejznámější a nejvíce „historická“, další modely na ni více či méně navazovaly (nebo si ji alespoň uvědomovaly), uvedu k ní příslušnou matematickou realizaci. Dalším důvodem pro to je fakt, že učící algoritmus, který používá, se objevuje s různými obměnami v mnoha dalších modelech.

Vstupním neuronům zadáváme číselně reprezentovaný problém – vektor **x**, každá synapse je číselně ohodnocena – vektor všech synaptických vah **w**. Všechna čísla jsou reálné hodnoty. Neuron j v další vrstvě pak počítá svůj tzv. vnitřní potenciál:

$$\xi_j = \sum_{i \in j_{\leftarrow}} w_{ji} y_i$$

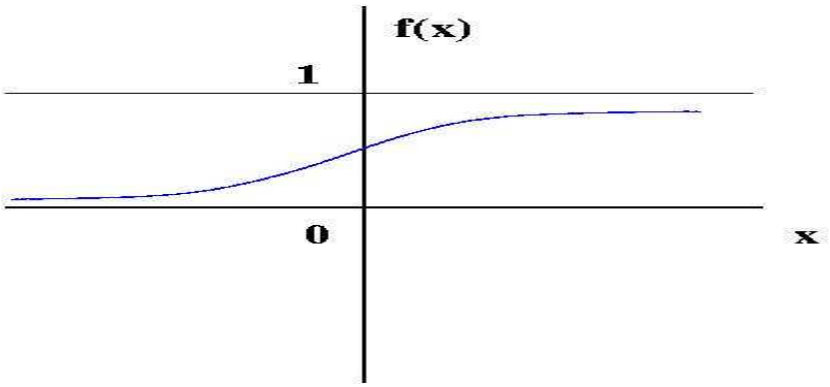
Každý neuron si potenciál „zpracuje“ a získá tak hodnotu y, kterou posílá neuronům v další vrstvě.

$$y_j = \sigma(\xi_j)$$

$$\sigma_j(\xi) = \frac{1}{1+e^{-\lambda_j \xi}}$$ třeba si všimnout, že $\sigma : R \to (0,1)$

Tomáš Janík
172787

Zatímco pro obyčejnou perceptronovou síť je použitá σ ostrá nelinearita, my potřebujeme (neboť to vyžaduje algoritmus backpropagation) funkci spojitou a diferencovatelnou.



V předpisu funkce je použita proměnná λ_j, který má efekt na strmosti funkce. Tento parametr se dá interpretovat jako míru rozhodnosti sítě (všimněme si, že pokud půjde $\lambda \to \infty$, bude se funkce limitně podobat ostré nelinearitě – de facto rozhodování ano/ne).

„Zpětné šíření“

Algoritmus backpropagation (zpětné šíření) nejprve vyjádří chybu sítě jako součet všech odchylek od požadovaných výsledků:

$$E(w) = \frac{1}{2}\sum_{k=1}^{p}\sum_{j\in Y}(y_j(w, x_k) - d_{kj})$$

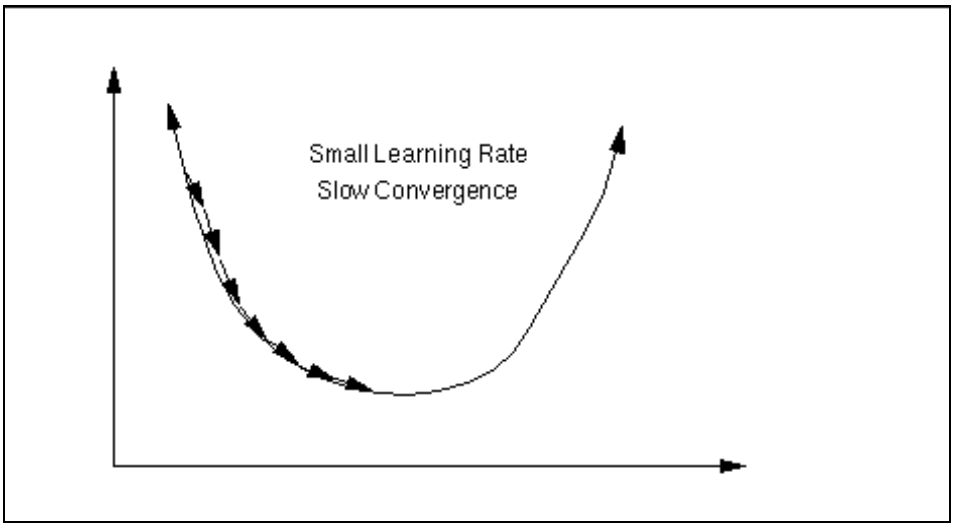
Tato funkce je spojitou funkcí váhového vektoru a jelikož chceme, aby chyba sítě byla co nejmenší, celý problém se redukuje na hledání lokálního minima funkce. V každém kroku učení je jsou předloženy všechny „učící pomůcky“ a upraven váhový vektor:

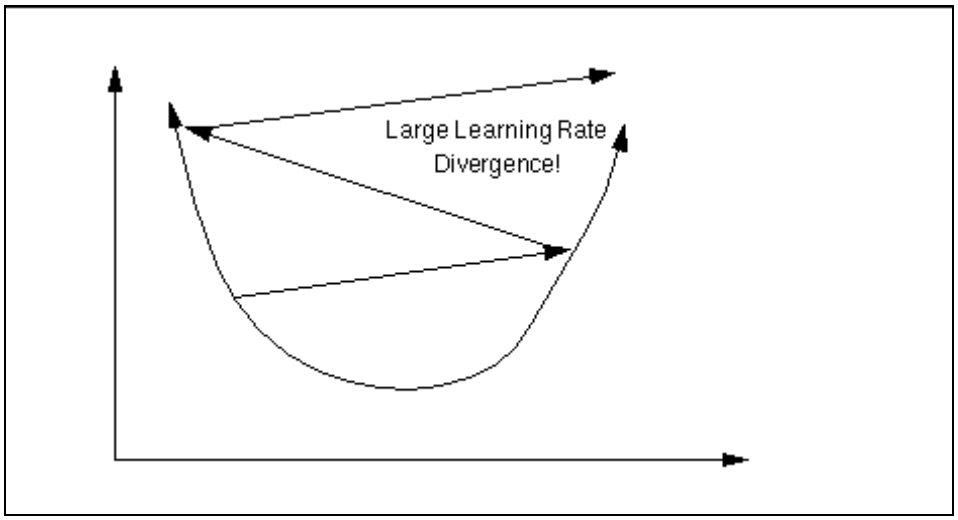
$$w_{ji}^{(t)} = w_{ji}^{(t-1)} + \Delta w_{ji}^{(t)}$$

Funkce je poté minimalizována neustálým odečítáním záporného gradientu, epsilon zde představuje rychlost učení

$$\Delta w_{ji}^{(t)} = -\varepsilon \frac{\partial E}{\partial w_{ji}}(w^{(t-1)})$$

Následující obrázky ukazují, co se stane, pokud je epsilon zvoleno příliš malé nebo příliš velké:





Zbytek práce je výpočet derivace funkce E(w), který je vzhledem k tomuto referátu příliš složitý, abychom se jím zabývali.

„Nejslavnější perceptronová síť“ – Mark I Perceptron

V roce 1958 manifestoval Frank Rosenblatt výpočetní sílu sítě perceptronů (jež byl tvůrce) na neuropočítači Mark I Perceptron. Byl navržen na rozpoznávání znaků (původním odborným záměrem Rosenblatta bylo rozpoznávání obrazců). Znak byl promítán na tabuli, ze které byl snímán polem 20x20 fotovodičů. Intenzita 400 obrazových bodů byla vstupem neuronové sítě. Mark I Perceptron měl 512 adaptovatelných váhových parametrů (**w**), které byly realizovány polem 8x8x8 potenciometrů. Hodnota odporu u každého potenciometru byla nastavována samostatným motorem. Perceptronovou síť implementoval analogový obvod, který řídil běh motorů. Rosenblatt k této prezentaci sezval mnoho důležitých postav tehdejšího informačního průmyslu. Poté, co se Mark I Perceptron naučil znaky rozpoznávat, dovolil divákům, aby obvody mezi potenciometry libovolně zpřeházeli, a ukázal, že jeho počítač je přesto schopen se na novou situaci adaptovat. Proběhlo nové učení a Mark I Perceptron opět demonstroval

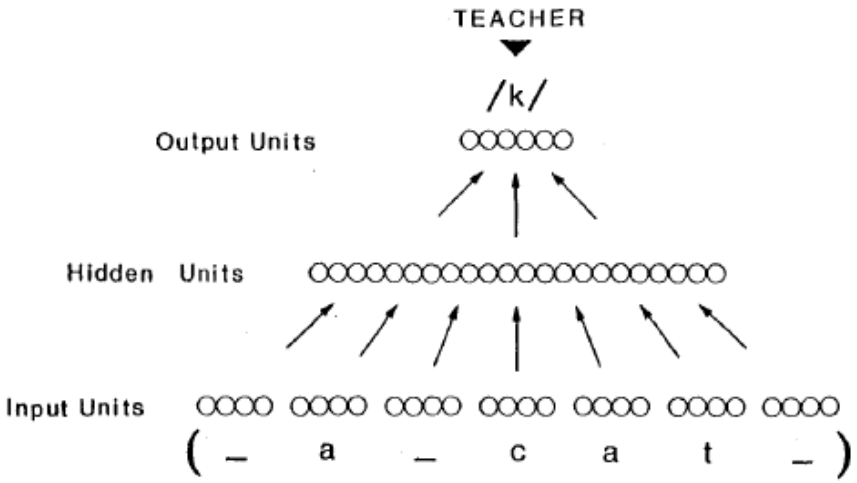
správný běh. Právě zásluhou Franka Rosenblatta se stala oblast neurovýpočtů na jistou dobu nejpozorovanějším a nejfinancovanějším odvětvím informačních technologií. Tento fakt měl také vědecké opodstatnění – přestože už v době, kdy Rosenblatt vynalezl perceptron, existovaly jiné modely sítí, Rosenblatt pro svou síť matematicky dokázal, že je schopna se v konečném času naučit jakýkoli problém, pokud řešení takového problému existuje. To jsou důvody, proč je Rosenblatt považován za zakladatele neurovýpočtů i přes značné zásluhy jeho předchůdců.

Obecné využití sítě perceptronů

Mark I Perceptron byl pouze jednovrstevný model, proto mu stačila standardní ostrá nelinearita při rozhodování. Je zřejmé, že jednovrstevná síť perceptronů nikdy nebude schopna řešit všechny druhy funkcí, typickým příkladem je funkce XOR, na kterou je třeba alespoň jedné vrstvy mezi vstupem a výstupem. Přestože se v padesátých letech vědělo, že vytvořením takové vícevrstvé sítě by se funkce XOR dala simulovat, nebyl znám žádný učící algoritmus pro takovou síť. Autoři z toho nesprávně vyvodili, že takový algoritmus vzhledem ke komplikovanosti funkce, kterou vícevrstvá síť představuje, ani není možný. Jejich tvrzení se přejalo a považovalo se za přesvědčivé.

Je dokázáno, že vícevrstvá (alespoň jedna vrstva mezi vstupem a výstupem – více se v klasické perceptronové síti nepoužívá) síť dokáže aproximovat libovolnou spojitou funkci s libovolnou přesností. Přesnost je určena volbou počtu neuronů ve skryté vrstvě. Vzhledem k tomu, že síti zadáváme (většinou) empirická data, je velký počet neuronů chyba. V takovém případě se síť snaží naučit všechny odchylky od nějakého průměru, místo aby hledala optimální generalizaci (přeučení sítě), naopak volba malého počtu neuronů vede ke generalizaci přílišné a nežádoucí.

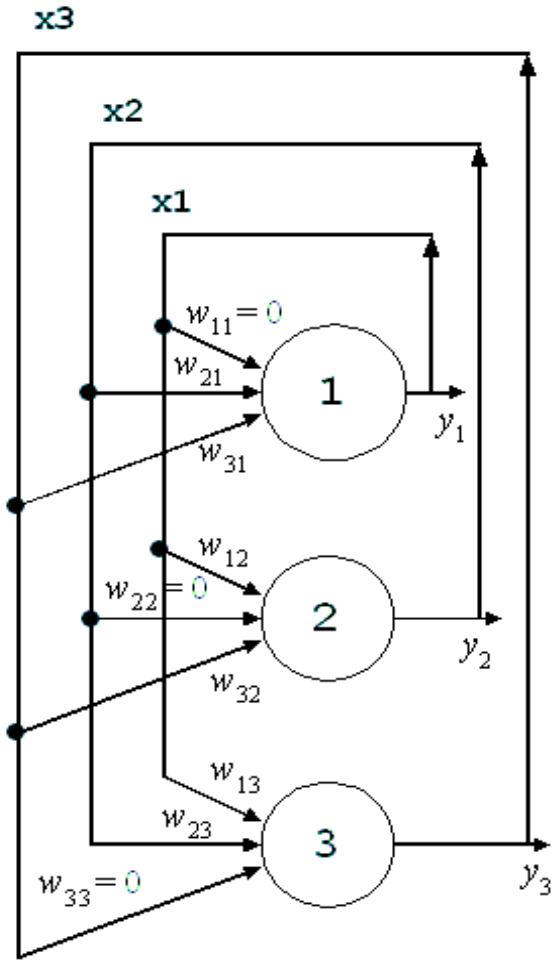
Velmi slavný příklad vícevrstvé perceptronové sítě s algoritmem backpropagation je systém NETtalk vyvinutým Terrencem Sejnowskim a Charlesem Rosenbergem. Jde o převod anglicky psaného textu na mluvený signál. Síť má 7x29 vstupních neuronů pro zakódování kontextu sedmi písmen psaného textu (každé možnosti odpovídá jeden neuron – 26 pro písmena + jeden pro tečku, čárku a mezeru). Síť obsahuje 80 skrytých neuronů a 26 výstupních pro všechny anglické fonémy. Tréninkovou množinu tvoří zadávání věty – zadává se každé písmeno s šesti znaky, který tvoří kontext. NETtalk zaznamenal velký úspěch a vynutil pozornost dalšímu výzkumu neuronových sítí.



Tomáš Janík
172787

Hopfieldova síť

Architektura Hopfieldovy sítě je úplný K_n graf, kde n je počet neuronů. Takto vypadá pro n=3:



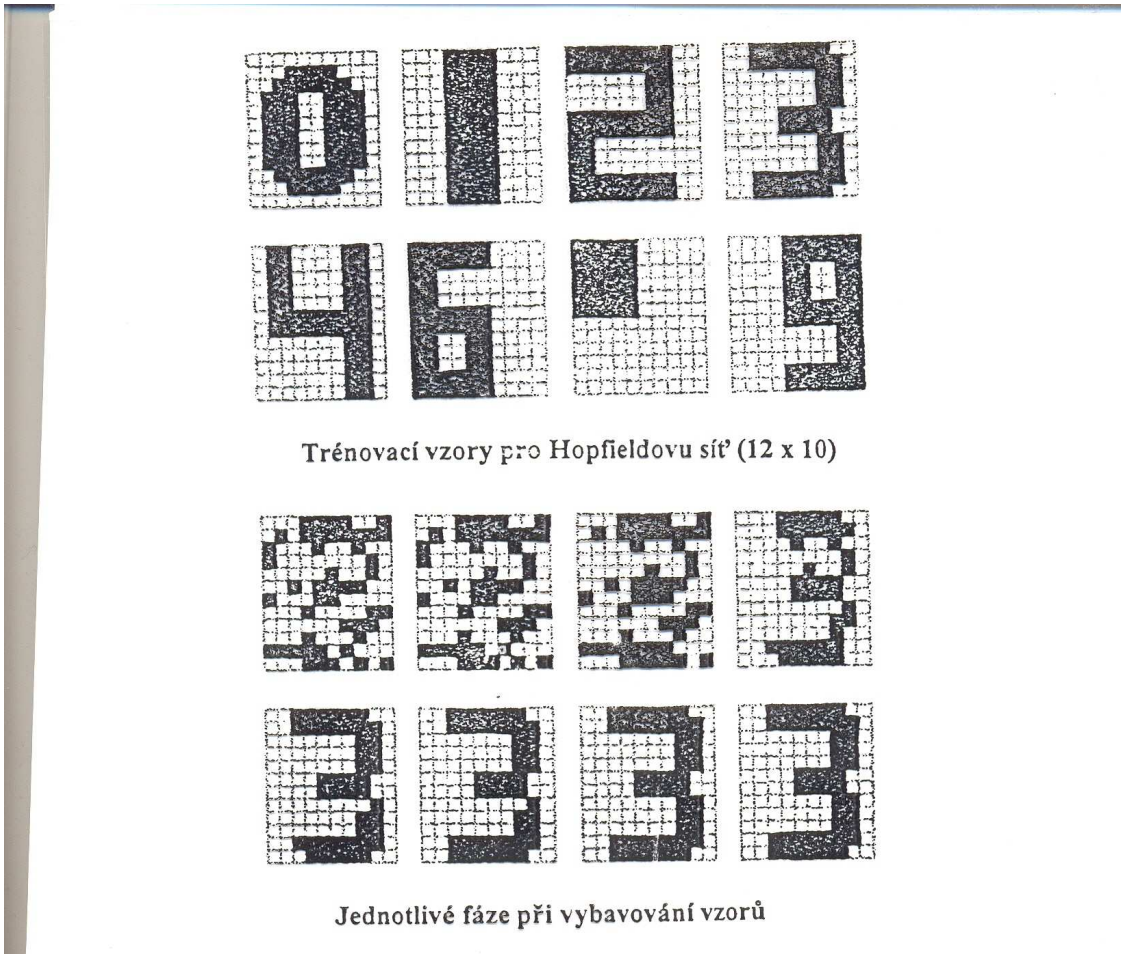
Hopfieldova síť je jedním z modelů tzv. asociativních pamětí (vedle lineární asociativní sítě). Autoasociativita je vlastnost pozorovaná u člověka, je to např. schopnost vybavit si celé jméno člověka, pokud nám někdo připomene křestní jméno nebo třeba jen počáteční písmeno jeho příjmení (Hopfieldova síť může být případně použita v databázových systémech), schopnost na základě části obrazu představit si, co je vně. Nečastěji se využívá při ostření obrazu. Model navrhl už McCulloch a Pitts, ale teprve díky Hopfieldovi, který při analýze stability této sítě využil analogii s fyzikální teorií magnetických materiálů, se tento model stal všeobecně známým.

Výpočetní princip je proti perceptronům obrácený. Síť se velice rychle a snadno adaptuje (naučí) – vytváří si tzv. stabilní stavy podle tréninkových vzorů. Náročnější je samotný běh. Nejprve se vyjádří tzv. energetická funkce:

$$E(y) = -\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} w_{ji} y_j y_i$$

Tomáš Janík
172787

Opět podobně jako tomu bylo u perceptronů se hledá lokální minimum (místo s nejnižší energií) – to je pak výpočet sítě. Příklad použití Hopfieldovy sítě:



Trénovací vzory pro Hopfieldovu síť (12 x 10)

Jednotlivé fáze při vybavování vzorů

Hopfieldova síť se dá použít jako heuristika při řešení problému obchodního cestujícího, problém je stálu v řádu NP úplnosti, ale tato metoda může nabízet určité zefektivnění:

Máme dáno **N** měst (**N > 2**) a funkci **d** definující vzdálenost měst pro každou dvojici. Použijeme síť s **n = N x N** neurony, jejichž stavy indexujeme $\mathbf{y_{ru}}$ (**r** označuje číslo města a **u** jeho možné pořadí). Neurony v ní uspořádáme maticově, jak ukazuje tabulka 1.

	zastávky			
města				

Řádky tabulky představují města, která chceme navštívit a sloupce určují pořadí zastávek v těchto městech.

- **Cestující navštíví každé město právě jednou, tedy v každém řádku tabulky je aktivní jeden neuron**.
  To dosáhneme minimalizací výrazu

$$E_A = \frac{A}{2}\sum_{r=1}^{N}\sum_{u=1}^{N}\sum_{\substack{v=1 \\ v\neq u}}^{N} y_{ru}y_{rv}$$

- kde **A** je konstanta určující vliv této podmínky.
- **Cestující je při každé zastávce pouze v jednom městě, tedy v každém sloupci tabulky je aktivní jeden neuron.**
  Výraz k minimalizaci je

$$E_B = \frac{B}{2}\sum_{u=1}^{N}\sum_{r=1}^{N}\sum_{\substack{s=1 \\ s\neq r}}^{N} y_{ru}y_{su}$$

- kde **B** má podobnou úlohu jako **A** v minulém výrazu.
- **Cestující navštíví právě N měst, tedy v tabulce je aktivních právě N neuronů.**

$$E_C = \frac{C}{2}\left(N - \sum_{r=1}^{N}\sum_{u=1}^{N} y_{ru}\right)^2$$

- **Cestující vykoná co nejkratší cestu, řešení bude optimální.**

$$E_D = \frac{D}{2}\sum_{r=1}^{N}\sum_{s=1}^{N}\sum_{u=1}^{N} d(r,s)\,y_{ru}\,(y_{s,u-1} + y_{s,u+1})$$

- Výrazy s proměnnou **u** je nutné brát modulo počtem měst.

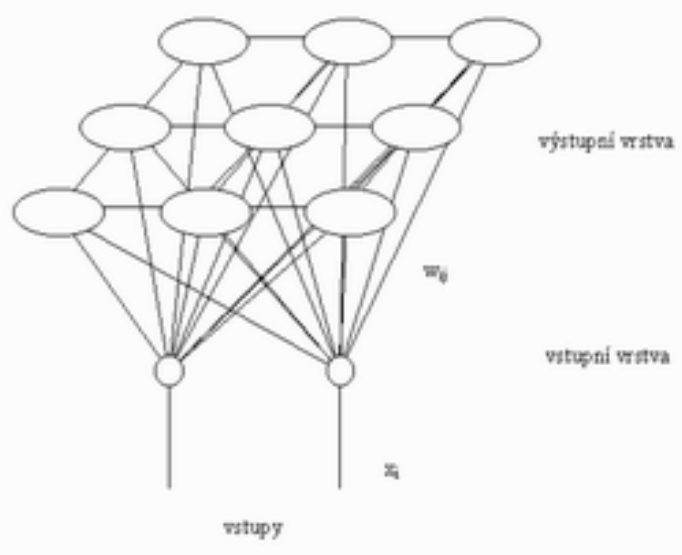
Výsledná funkce energie sítě, jejíž minimalizací získáme řešení, je součtem uvedených výrazů:

$$E = E_A + E_B + E_C + E_D$$

Učení bez učitele (Unsupervised Learning)

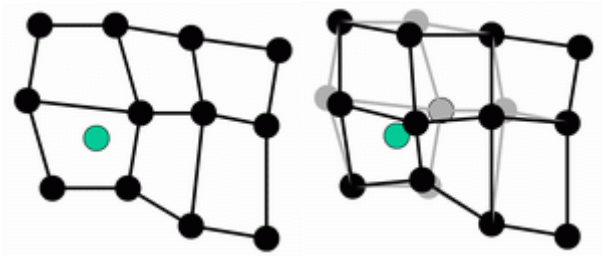
Učení bez učitele nemá žádné kritérium správnosti, algoritmus je navržen tak, že hledá na vstupních datech určité vzorky se společnými vlastnostmi – proto se učení bez učitele také nazývá samoorganizace. Společným principem těchto modelů je, že výstupní neurony sítě spolu soutěží o to, který z nich bude aktivní. Na rozdíl od předešlých učících principů je tedy v určitém čase aktivní pouze jeden neuron. Samoorganizačním sítím (SOM = Self-Organizing Map nebo Kohonenovy mapy po svém stvořiteli) se zadávají vstupní hodnoty a neurčuje se

jim, jak má vypadat výsledek. Hledají společné vlastnosti a generalizace, nezkoumají, co je správně, ale hledají v daném problému něco, co se dá použít k vnitřní organizaci – inspirace lidským učením. Nejčastějším použitím je zpracování řeči, detekce osob podle fotografie, odstranění rušení (podobné jako Hopfield), automatické třídění všeho druhu.



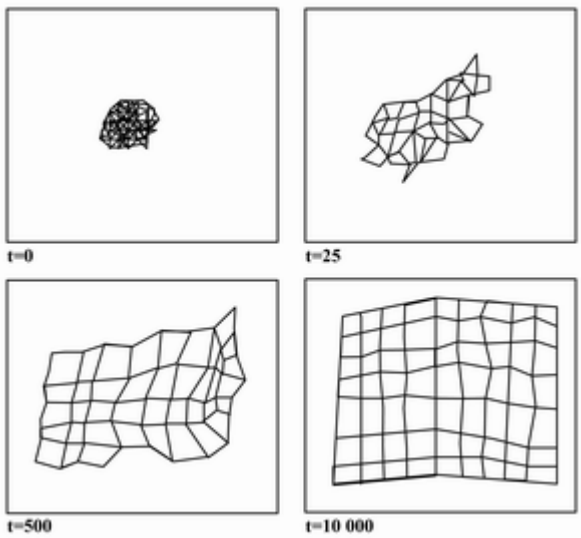
Příklad jednoduché fyzické struktury Kohonenovy mapy

Neuron, který má k předloženému vstupu nejblíže se označí jako vítězný a ten jediný posune svůj váhový vektor ještě blíže ke vzoru. Interpretace Kohonenova učení pak souvisí s výše uvedeným soutěžením. Na následujícím obrázku je tento efekt geometricky zobrazen v trojrozměrném poli (mřížka na výstupu je dvojrozměrná).



Vítězný neuron oslabuje ostatní neurony a sám svoji pozici upevňuje. Na mřížce výstupních neuronů (viz obrázek) tak vznikají silní reprezentanti určitých trendů – akt třízení, neurony kolem těchto silných reprezentantů jsou okamžitě utlumeny, vzdálenější neurony se postupně přidávají (pokud tedy nejsou sami reprezentanty) k některým silným jedincům a jsou utlumeny. Tomuto systému se někdy také říká „k-means clustering“ – hledání k středů.
V aktivním režimu je pak zadanému vzoru přisouzen ten neuron z množiny vítězů v průběhu učení, který dovede zadaný vzor nejlépe reprezentovat.

Tomáš Janík
172787



Rozložení neuronů ve vstupním datovém prostoru po různém počtu kroků učení