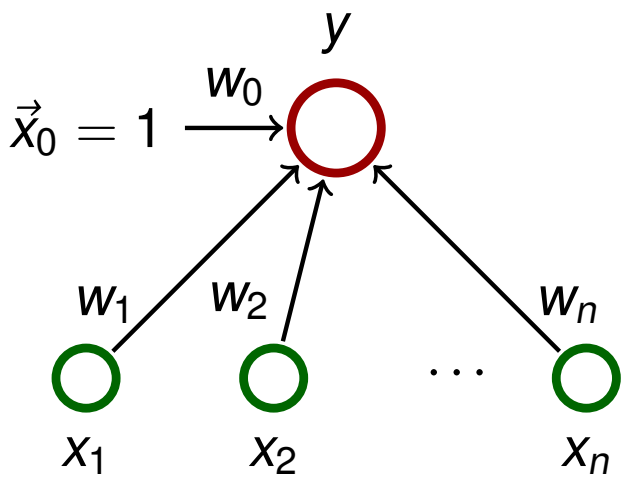
ADALINE

Organizační dynamika:



$\vec{w} = (w_0, w_1, \ldots, w_n)$ a $\vec{x} = (x_0, x_1, \ldots, x_n)$ kde $x_0 = 1$.

Aktivní dynamika:

funkce sítě: $y[\vec{w}](\vec{x}) = \vec{w} \cdot \vec{x}$

Poznámka: násobení vektorů

Pokud nebude řečeno jinak, budeme uvažovat všechny vektory jako *sloupcové*

$$\vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix} \qquad pak \quad \vec{a} = (a_0, a_1, \ldots, a_m)^\top$$

Skalární součin lze chápat jako součin matic:

$$\vec{a} \cdot \vec{b} = \sum_{k=0}^{m} a_k b_k = \vec{a}^\top \vec{b} = \vec{b}^\top \vec{a}$$

Libovolná dimenze - adaptivní dynamika - shrnutí

Tréninková množina:

$$\mathcal{T} = \left\{\left(\vec{x}_1, d_1\right), \left(\vec{x}_2, d_2\right), \ldots, \left(\vec{x}_p, d_p\right)\right\}$$

Zde $\vec{x}_k = (x_{k0}, x_{k1} \ldots, x_{kn})^\top \in \mathbb{R}^{n+1}$, $x_{k0} = 1$, je vstup k-tého vzoru a $d_k \in \mathbb{R}$ je očekávaný výstup.

Chybová funkce:

$$E(\vec{w}) = \frac{1}{2}\sum_{k=1}^{p}\left(\vec{w}^\top \vec{x}_k - d_k\right)^2 = \frac{1}{2}\sum_{k=1}^{p}\left(\sum_{i=0}^{n} w_i x_{ki} - d_k\right)^2$$

Gradient E:

$$\nabla E(\vec{w}) = \left(\frac{\partial E}{\partial w_0}(\vec{w}), \ldots, \frac{\partial E}{\partial w_n}(\vec{w})\right)^\top = \sum_{k=1}^{p}\left(\vec{w}^\top \vec{x}_k - d_k\right)\vec{x}_k$$

Minimalizace chyby

Nechť X je matice $p \times (n+1)$ (tj. p řádků, $n+1$ sloupců), jejíž k**-tý řádek** je tvořen vektorem $\vec{x}_k^\top$

Nechť $\vec{d} = (d_1, \ldots, d_p)^\top$

Pak

$$\begin{aligned} \nabla E(\vec{w}) = 0 \quad &\Leftrightarrow \quad (X^\top X)\vec{w} = X^\top \vec{d} \\ &\Leftrightarrow \quad X^\top (X\vec{w}) = X^\top \vec{d} \\ &\Leftrightarrow \quad \vec{w} = (X^\top X)^{-1} X^\top \vec{d} \end{aligned}$$

pokud $(X^\top X)^{-1}$ existuje.

(pak $(X^\top X)^{-1} X^\top$ je tzv. Moore-Penrose pseudoinverse matice X)

Všimněte si, že

$$(X^\top X)_{ij} = \sum_{k=1}^{p} x_{ki} x_{kj} \qquad \text{a} \qquad (X^\top \vec{d})_i = \sum_{k=1}^{p} x_{ki} d_k$$

(zde $(X^\top X)_{ij}$ je prvek matice $X^\top X$ na i-tém řádku a j-tém sloupci)

ADALINE - statistické učení

Uvažme následující modifikaci **adaptivní dynamiky**:

- Síti je předkládána (nekonečná) posloupnost tréninkových vzorů $\left(\vec{x}_1, d_1\right), \left(\vec{x}_2, d_2\right), \ldots$ kde
  - vstupy $\vec{x}_k = (x_{k0}, x_{k1}, \ldots, x_{kn})^\top \in \mathbb{R}^n$ jsou generovány náhodně s fixním rozdělením pravděpodobností
  - každý vstup $\vec{x}_k$ má přiřazen požadovaný výstup $d_k \in \mathbb{R}$.
- Pro danou konfiguraci $\vec{w}$ definujeme chybu:

$$E(\vec{w}) = \lim_{p\to\infty} \frac{1}{p} \sum_{k=1}^{p} \frac{1}{2} \left(\vec{w}^\top \vec{x}_k - d_k\right)^2$$

Poznámka: Podle zákona o velkých číslech je $E(\vec{w})$ střední hodnota proměnné, která vrací $\frac{1}{2}\left(\vec{w}^\top \vec{x}_k - d_k\right)^2$, tedy $E(\vec{w}) = \mathbb{E}\left[\frac{1}{2}\left(\vec{w}^\top \vec{x}_k - d_k\right)^2\right]$

- **Gradient E:**

$$\nabla E(\vec{w}) = \lim_{p\to\infty} \frac{1}{p} \sum_{k=1}^{p} \left(\vec{w}^\top \vec{x}_k - d_k\right) \vec{x}_k$$

ADALINE - statistické učení

Nechť A je matice $(n+1) \times (n+1)$ splňující

$$A_{ir} = \lim_{p \to \infty} \frac{1}{p} \sum_{k=1}^{p} x_{ki} x_{kr}$$

a $D \in \mathbb{R}^{n+1}$ vektor splňující

$$D_i = \lim_{p \to \infty} \frac{1}{p} \sum_{k=1}^{p} d_k x_{ki}$$

Pak $\nabla E(\vec{w}) = A\vec{w} - D$ a tedy

$$\begin{aligned} E(\vec{w}) = 0 &\Leftrightarrow A\vec{w} = D \\ &\Leftrightarrow \vec{w} = A^{-1} D \end{aligned}$$

Problémy:

1. neznáme A a D
2. A^{-1} nemusí existovat i přesto, že E má vždy jednoznačně určené minimum

ADALINE - statistické učení

Pokud lze odhadnout A_{ir} a D_i, můžeme použít gradientní sestup stejně jako v případě normálního ADALINE učení:

- váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- v kroku $t+1$ (zde $t = 0, 1, 2, \ldots$) je $\vec{w}^{(t+1)}$ vypočteno takto:

$$\begin{aligned} \vec{w}^{(t+1)} &= \vec{w}^{(t)} - \varepsilon \cdot \nabla E(\vec{w}^{(t)}) \\ &= \vec{w}^{(t)} - \varepsilon \cdot (A\vec{w}^{(t)} - D) \end{aligned}$$

Co když ani nelze odhadnout A_{ir} a D_i ?

ADALINE - statistické učení

Naštěstí funguje online algoritmus (Widrow & Hoff), který je úplně stejný jako v předchozí „nestatistické“ variantě!

- váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- v kroku $t+1$ (zde $t = 0, 1, 2, \ldots$) je $\vec{w}^{(t+1)}$ vypočteno takto:

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \varepsilon(t) \cdot \left(\vec{w}^{(t)} \cdot \vec{x}_{t+1} - d_{t+1}\right) \cdot \vec{x}_{t+1}$$

kde $0 < \varepsilon(t) \leq 1$ je rychlost učení v kroku $t+1$.

Věta (Widrow & Hoff)

Pro $\varepsilon(t) = \frac{1}{t}$ posloupnost $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \ldots$ konverguje ke globálnímu minimu chybové funkce $E(\vec{w})$.

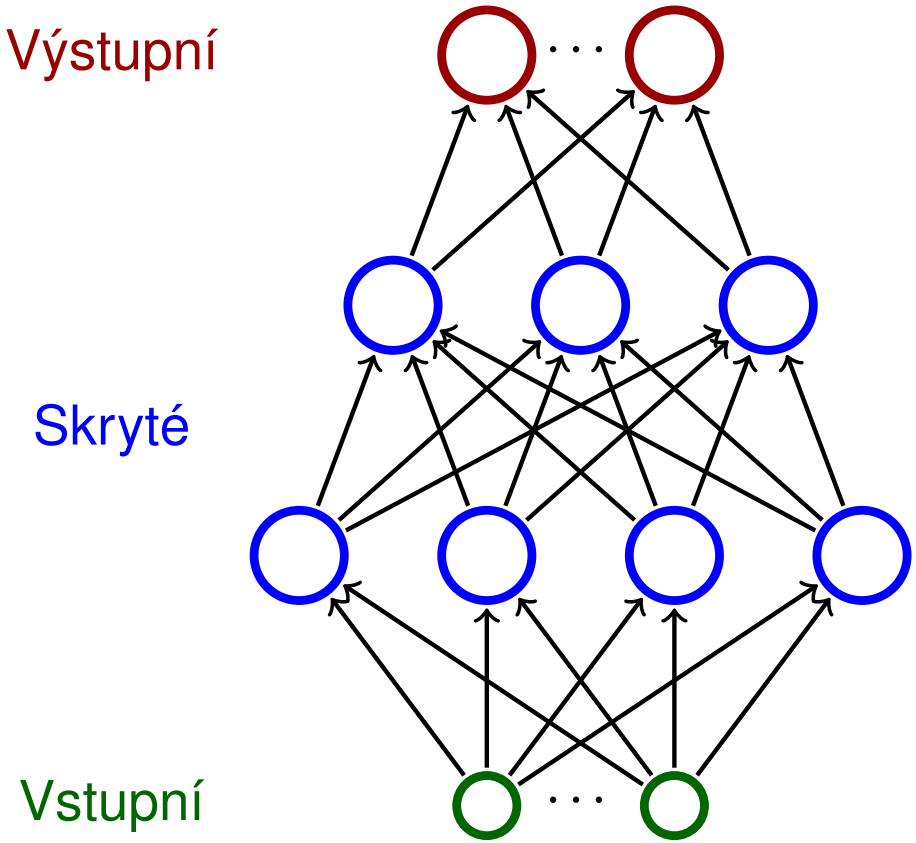
Matematická poznámka: zde se nejedná o konvergenci po složkách, ale o konvergenci průměru čtverců vzdálenosti (mean square convergence).

Vícevrstvá síť a zpětná propagace

- Vícevrstvé sítě - značení
- učící pravidlo zpětné propagace
- algoritmus zpětné propagace

Vícevrstvá síť

Organizační dynamika:



- Neurony jsou rozděleny do **vrstev** (vstupní a výstupní vrstva, obecně několik skrytých vrstev)
- Vrstvy číslujeme od 0; vstupní vrstva je nultá
  - Např. třívrstvá síť se skládá z jedné vstupní, dvou skrytých a jedné výstupní vrstvy.
- Neurony v ℓ-té vrstvě jsou spojeny se všemi neurony ve vrstvě $\ell + 1$.
- Vícevrstvou síť lze zadat počty neuronů v jednotlivých vrstvách (např. 2-4-3-2)

Vícevrstvá síť

Značení:

- Označme
  - X množinu vstupních neuronů
  - Y množinu výstupních neuronů
  - Z množinu všech neuronů (tedy $X, Y \subseteq Z$)
- jednotlivé neurony budeme značit indexy i, j apod.
- ξ_j je vnitřní potenciál neuronu j po skončení výpočtu
- y_j je stav (výstup) neuronu j po skončení výpočtu

  (zde definujeme $y_0 = 1$ jako hodnotu formálního jednotkového vstupu)
- w_{ji} je váha spoje **od** neuronu i **do** neuronu j

  (zejména w_{j0} je váha speciálního jednotkového vstupu, tj. $w_{j0} = -b_j$ kde b_j je bias neuronu j)
- $j_\leftarrow$ je množina všech neuronů, **z nichž** vede spoj do j (zejména $0 \in j_\leftarrow$)
- $j^\rightarrow$ je množina všech neuronů, **do nichž** vede spoj z j

Vícevrstvá síť

Aktivní dynamika:

- vnitřní potenciál neuronu j:

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

- aktivační funkce σ_j pro neuron j je libovolná diferencovatelná funkce
  [např. logistická sigmoida $\sigma_j(\xi) = \frac{1}{1+e^{-\lambda_j \xi}}$]
- Stav nevstupního neuronu $j \in Z \setminus X$ po skončení výpočtu je

$$y_j = \sigma_j(\xi_j)$$

  (y_j závisí na konfiguraci $\vec{w}$ a vstupu $\vec{x}$, proto budu občas psát $y_j(\vec{w}, \vec{x})$)

- Síť počítá funkci z $\mathbb{R}^{|X|}$ do $\mathbb{R}^{|Y|}$. Výpočet probíhá po vrstvách. Na začátku jsou hodnoty vstupních neuronů nastaveny na vstup sítě. V kroku ℓ jsou vyhodnoceny neurony z ℓ-té vrstvy.

Vícevrstvá síť

Adaptivní dynamika:

- Dána množina $\mathcal{T}$ **tréninkových vzorů** tvaru

$$\left\{ \left(\vec{x}_k, \vec{d}_k \right) \quad \middle| \quad k = 1, \ldots, p \right\}$$

kde každé $\vec{x}_k \in \mathbb{R}^{|X|}$ je vstupní vektor a každé $\vec{d}_k \in \mathbb{R}^{|Y|}$ je očekávaný výstup sítě. Pro každé $j \in Y$ označme d_{kj} očekávaný výstup neuronu j pro vstup $\vec{x}_k$
(vektor $\vec{d}_k$ lze tedy psát jako $\left(d_{kj} \right)_{j \in Y}$).

- **Chybová funkce:**

$$E(\vec{w}) = \sum_{k=1}^{p} E_k(\vec{w})$$

kde

$$E_k(\vec{w}) = \frac{1}{2} \sum_{j \in Y} \left(y_j(\vec{w}, \vec{x}_k) - d_{kj} \right)^2$$

Vícevrstvá síť - učící algoritmus

Dávkový algoritmus (gradientní sestup):

Algoritmus počítá posloupnost vektorů vah $\vec{w}^{(0)}, \vec{w}^{(1)}, \ldots$.

- váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- v kroku $t+1$ (zde $t = 0, 1, 2 \ldots$) je $\vec{w}^{(t+1)}$ vypočteno takto:

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

kde

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

je změna váhy w_{ji} v kroku $t+1$ a $0 < \varepsilon(t) \leq 1$ je rychlost učení v kroku $t+1$.

Všimněte si, že $\frac{\partial E}{\partial w_{ji}}(\vec{w}^{(t)})$ je komponenta gradientu ∇E, tedy změnu vah v kroku $t+1$ lze zapsat také takto: $\vec{w}^{(t+1)} = \vec{w}^{(t)} - \varepsilon(t) \cdot \nabla E(\vec{w}^{(t)})$.

Vícevrstvá síť - gradient chybové funkce

Pro každé w_{ji} máme

$$\frac{\partial E}{\partial w_{ji}} = \sum_{k=1}^{p} \frac{\partial E_k}{\partial w_{ji}}$$

kde pro každé $k = 1, \ldots, p$ platí

$$\frac{\partial E_k}{\partial w_{ji}} = \frac{\partial E_k}{\partial y_j} \cdot \sigma_j'(\xi_j) \cdot y_i$$

a pro každé $j \in Z \setminus X$ dostaneme

$$\frac{\partial E_k}{\partial y_j} = y_j - d_{kj} \qquad \text{pro } j \in Y$$

$$\frac{\partial E_k}{\partial y_j} = \sum_{r \in j^{\rightarrow}} \frac{\partial E_k}{\partial y_r} \cdot \sigma_r'(\xi_r) \cdot w_{rj} \qquad \text{pro } j \in Z \setminus (Y \cup X)$$

(Zde všechna y_j jsou ve skutečnosti $y_j(\vec{w}, \vec{x}_k)$).

Vícevrstvá síť - gradient chybové funkce

- Pokud $\sigma_j(\xi) = \frac{1}{1+e^{-\lambda_j \xi}}$ pro každé $j \in Z$ pak

$$\sigma_j'(\xi_j) = \lambda_j y_j (1 - y_j)$$

a dostaneme pro každé $j \in Z \setminus X$:

$$\frac{\partial E_k}{\partial y_j} = y_j - d_{kj} \qquad \text{pro } j \in Y$$

$$\frac{\partial E_k}{\partial y_j} = \sum_{r \in j^{\rightarrow}} \frac{\partial E_k}{\partial y_r} \cdot \lambda_r y_r (1 - y_r) \cdot w_{rj} \qquad \text{pro } j \in Z \setminus (Y \cup X)$$

- Pokud $\sigma_j(\xi) = a \cdot \tanh(b \cdot \xi_j)$ pro každé $j \in Z$ pak

$$\sigma_j'(\xi_j) = \frac{b}{a}(a - y_j)(a + y_j)$$

Vícevrstvá síť - výpočet gradientu

Algoritmicky lze $\frac{\partial E}{\partial w_{ji}} = \sum_{k=1}^{p} \frac{\partial E_k}{\partial w_{ji}}$ spočítat takto:

Polož $\mathcal{E}_{ji} := 0$ (na konci výpočtu bude $\mathcal{E}_{ji} = \frac{\partial E}{\partial w_{ji}}$)

Pro každé $k = 1, \ldots, p$ udělej následující

1. spočítej y_j pro každé $j \in Z$ a k-tý vzor, tedy $y_j = y_j(\vec{w}, \vec{x}_k)$
   (tj. vyhodnoť síť ve standardním aktivním režimu)
2. pro všechna $j \in Z$ spočítej $\frac{\partial E_k}{\partial y_j}$ pomocí zpětného šíření
   (viz. následující slajd!)
3. spočítej $\frac{\partial E_k}{\partial w_{ji}}$ pro všechna w_{ji} pomocí vzorce
   $$\frac{\partial E_k}{\partial w_{ji}} := \frac{\partial E_k}{\partial y_j} \cdot \sigma_j'(\xi_j) \cdot y_i$$
4. $\mathcal{E}_{ji} := \mathcal{E}_{ji} + \frac{\partial E_k}{\partial w_{ji}}$

Výsledné E_{ji} se rovná $\frac{\partial E}{\partial w_{ji}}$.

Vícevrstvá síť - zpětné šíření

$\frac{\partial E_k}{\partial y_j}$ lze spočítat pomocí zpětného šíření:

- spočítáme $\frac{\partial E_k}{\partial y_j}$ pro $j \in Y$ pomocí vzorce $\frac{\partial E_k}{\partial y_j} = y_j - d_{kj}$
- rekurzivně spočítáme zbylé $\frac{\partial E_k}{\partial y_j}$:

  Nechť j je v ℓ-té vrstvě a předpokládejme, že $\frac{\partial E_k}{\partial y_r}$ už máme spočítáno pro všechny neurony z vyšších vrstev (tedy vrstev $\ell + 1, \ell + 2, \ldots$).
  Pak lze $\frac{\partial E_k}{\partial y_j}$ spočítat pomocí vzorce

$$\frac{\partial E_k}{\partial y_j} = \sum_{r \in j^{\rightarrow}} \frac{\partial E_k}{\partial y_r} \cdot \sigma_r'(\xi_r) \cdot w_{rj}$$

  protože všechny neurony $r \in j^{\rightarrow}$ patří do vrstvy $\ell + 1$.

Složitost dávkového algoritmu

Výpočet hodnoty $\frac{\partial E}{\partial w_{ji}}(\vec{w}^{(t-1)})$ probíhá v lineárním čase vzhledem k velikosti sítě a počtu tréninkových vzorů

(za předpokladu jednotkové ceny operací včetně vyhodnocení $\sigma'_r(\xi_r)$ pro dané ξ_r)

Zdůvodnění: Algoritmus provede p krát následující

1. vyhodnocení sítě (tj. výpočet $y_j(\vec{w}, \vec{x}_k)$)
2. výpočet $\frac{\partial E_k}{\partial y_j}$ zpětným šířením
3. výpočet $\frac{\partial E_k}{\partial w_{ji}}$
4. přičtení $\frac{\partial E_k}{\partial w_{ji}}$ k $\mathcal{E}_{ji}$

Kroky 1. - 3. proběhnou v lineárním čase a krok 4. v konstantním vzhledem k velikosti sítě.

Počet iterací gradientního sestupu pro přiblížení se (lokálnímu) minimu může být velký ...

Vícevrstvá síť - učící algoritmus

Online algoritmus:

Algoritmus počítá posloupnost vektorů vah $\vec{w}^{(0)}, \vec{w}^{(1)}, \ldots$.

- váhy v $\vec{w}^{(0)}$ jsou inicializovány náhodně blízko 0
- v kroku $t+1$ (zde $t = 0, 1, 2, \ldots$) je $\vec{w}^{(t+1)}$ vypočteno takto:

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

kde

$$\Delta w_{ji}^{(t)} = -\varepsilon(t) \cdot \frac{\partial E_k}{\partial w_{ji}}(w_{ji}^{(t)})$$

kde $k = (t \mod p) + 1$ a $0 < \varepsilon(t) \leq 1$ je rychlost učení v kroku $t+1$.

Lze použít i **stochastickou verzi** tohoto algoritmu, v níž je k voleno náhodně z $\{1, \ldots, p\}$.