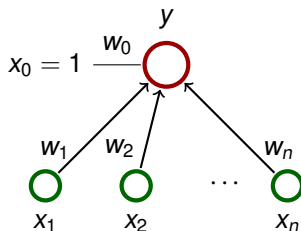


## Organizační dynamika:



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

## Aktivní dynamika:

- ▶ vnitřní potenciál:  $\xi = w_0 + \sum_{i=1}^n w_i x_i = \sum_{i=0}^n w_i x_i = \vec{w} \cdot \vec{X}$
- ▶ aktivační funkce:  $\sigma(\xi) = \xi$
- ▶ funkce sítě:  $y[\vec{w}](x_1, \dots, x_n) = \sigma(\xi) = \vec{w} \cdot \vec{X}$

**Adaptivní dynamika:**

- ▶ Dána množina **tréninkových vzorů**

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

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

Intuice: chceme, aby síť počítala afinní aproximaci funkce, jejíž (některé) hodnoty nám předepisuje tréninková množina.

- ▶ Označme:  $\vec{X}_k = (x_{k0}, x_{k1}, \dots, x_{kn}) \in \mathbb{R}^{n+1}$  kde  $x_{k0} = 1$ .
- ▶ **Chybová funkce:**

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

- ▶ Cílem je nalézt  $\vec{w}$ , které minimalizuje  $E(\vec{w})$ .

## Dávkový algoritmus (gradientní sestup):

- ▶ váhy v  $\vec{w}^{(0)}$  jsou inicializovány náhodně blízko 0
- ▶ v  $t$ -tém kroku je  $\vec{w}^{(t)}$  vypočteno takto:

$$\begin{aligned}\vec{w}^{(t)} &= \vec{w}^{(t-1)} - \varepsilon \cdot \nabla E(\vec{w}^{(t-1)}) \\ &= \vec{w}^{(t-1)} - \varepsilon \cdot \left( \frac{\partial E}{\partial w_0}(\vec{w}), \dots, \frac{\partial E}{\partial w_n}(\vec{w}) \right) \\ &= \vec{w}^{(t-1)} - \varepsilon \cdot \sum_{k=1}^p (\vec{w}^{(t-1)} \cdot \vec{X}_k - d_k) \cdot \vec{X}_k\end{aligned}$$

Zde  $0 < \varepsilon \leq 1$  je rychlost učení.

## Tvrzení

*Pro dostatečně malé  $\varepsilon > 0$  posloupnost  $\vec{w}^{(0)}, \vec{w}^{(1)}, \vec{w}^{(2)}, \dots$  konverguje (po složkách) ke globálnímu minimu funkce  $E$  (tedy k vektoru  $\vec{w}$ , který splňuje  $\nabla E(\vec{w}) = \vec{0}$ ).*

Uvažme následující modifikaci **adaptivní dynamiky**:

- ▶ Síť je předkládána (nekonečná) posloupnost tréninkových vzorů  $(\vec{x}_1, d_1), (\vec{x}_2, d_2), \dots$  kde
  - ▶ vstupy  $\vec{x}_k = (x_{k1}, \dots, x_{kn}) \in \mathbb{R}^n$  jsou generovány náhodně s daným rozdělením pravděpodobností
  - ▶ každý vstup  $\vec{x}_k$  má přiřazen požadovaný výstup  $d_k \in \mathbb{R}$ .
- ▶ Označme  $\vec{X}_k = (x_{k0}, x_{k1}, \dots, x_{kn})$  kde  $x_{k0} = 1$ .  
Výstup sítě pro  $k$ -tý vzor je potom  $\vec{w} \cdot \vec{X}_k$ .
- ▶ Pro danou konfiguraci  $\vec{w}$  definujeme chybu:

$$E(\vec{w}) = \lim_{p \rightarrow \infty} \frac{1}{p} \cdot \sum_{k=1}^p \frac{1}{2} (\vec{w} \cdot \vec{X}_k - d_k)^2$$

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

# ADALINE - statistické učení

Vypočteme gradient  $\nabla E(\vec{w}) = \left( \frac{\partial E}{\partial w_0}(\vec{w}), \dots, \frac{\partial E}{\partial w_n}(\vec{w}) \right)$  :

$$\begin{aligned} \frac{\partial E}{\partial w_i}(\vec{w}) &= \lim_{p \rightarrow \infty} \frac{1}{p} \cdot \sum_{k=1}^p (\vec{w} \cdot \vec{X}_k - d_k) \cdot x_{ki} \\ &= \sum_{r=0}^n w_r \cdot A_{ri} - C_i \end{aligned}$$

kde

$$A_{ri} = \lim_{p \rightarrow \infty} \frac{1}{p} \cdot \sum_{k=1}^p x_{kr} \cdot x_{ki}$$

$$C_i = \lim_{p \rightarrow \infty} \frac{1}{p} \cdot \sum_{k=1}^p d_k \cdot x_{ki}$$

Hledáme minimum  $E(\vec{w})$ , tedy  $\vec{w}^*$  takové, že  $\nabla E(\vec{w}^*) = 0$ .

Pokud dokážeme statisticky odhadnout  $A_{ri}$  a  $C_i$  pak můžeme rovnou položit:

$$0 = \frac{\partial E}{\partial w_i}(\vec{w}) = \sum_{r=0}^n w_r \cdot A_{ri} - C_i$$

a dostat  $\vec{w}^*$  jako řešení systému lineárních rovnic.

## Problémy:

1. řešení nemusí existovat  
(přestože minimum funkce  $E$  vždy existuje)
2. nemusíme mít odhad pro  $A_{ri}$  a  $C_i$

# ADALINE - statistické učení

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

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

$$\begin{aligned}\vec{w}^{(t)} &= \vec{w}^{(t-1)} - \varepsilon \cdot \nabla E(\vec{w}^{(t-1)}) \\ &= \vec{w}^{(t-1)} - \varepsilon \cdot \lim_{p \rightarrow \infty} \frac{1}{p} \cdot \sum_{k=1}^p (\vec{w}^{(t-1)} \cdot \vec{X}_k - d_k) \cdot \vec{X}_k \\ &= \vec{w}^{(t-1)} - \varepsilon \cdot \left( \sum_{r=0}^n \vec{w}_r^{(t-1)} \cdot A_{ri} - C_i \right)\end{aligned}$$

Co když ani nelze odhadnout  $A_{ri}$  a  $C_i$  ?

# 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  $t$ -tém kroku (zde  $t = 1, 2, \dots$ ) je  $\vec{w}^{(t)}$  vypočteno takto:

$$\vec{w}^{(t)} = \vec{w}^{(t-1)} - \varepsilon(t) \cdot (\vec{w}^{(t-1)} \cdot \vec{X}_t - d_t) \cdot \vec{X}_t$$

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

## Věta (Widrow & Hoff)

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

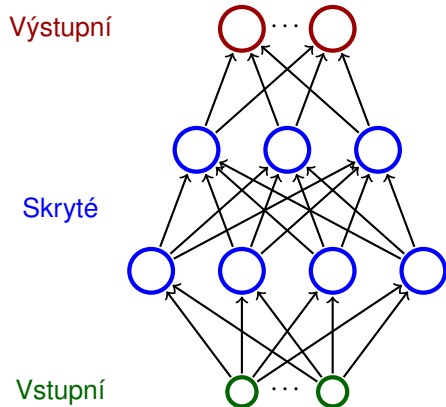
Matematická poznámka: zde se nejedná o konvergenci po složkách, ale o konvergenci průměru čtverců vzdálenosti (mean square convergence).



# Vícevrstvá síť a zpětná propagace

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

## 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  $\ell$ -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)

## 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.
- ▶  $\xi_j$  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  $j$

## Aktivní dynamika:

- ▶ vnitřní potenciál neuronu  $j$ :

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

- ▶ aktivační funkce  $\sigma_j$  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  $\ell$  jsou vyhodnoceny neurony z  $\ell$ -té vrstvy.

## Adaptivní dynamika:

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

$$\{(\vec{x}_k, \vec{d}_k) \mid k = 1, \dots, p\}$$

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

- ▶ **Chybová funkce:**

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

kde

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

## Dávkový algoritmus (gradientní sestup):

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

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

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

kde

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

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

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

# 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  $k = 1, \dots, p$  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

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

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

(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  $j \in Z \setminus X$ :

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

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

- ▶ 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  $k = 1, \dots, p$  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  $\mathcal{E}_{ji}$  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  $\ell$ -té vrstvě a předpokládejme, že  $\frac{\partial E_k}{\partial y_r}$  už máme spočítáno pro všechny neurony z vyšších vrstev (tedy vrstev  $\ell + 1, \ell + 2, \dots$ ).

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  $\xi_r$ )

**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 ...

## Online algoritmus:

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

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

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

kde

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

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

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