

3 - Formas de Aplicação do Treinamento BP

Processo em

Batelada – Batch

Lotes

Regra Delta - para treinamento “on line”, dinâmico

Regra delta com momento

Batelada - Algoritmo

Inicializar $n = 1$; $\alpha = .1$; $w_i(1) \in [+.2, -.2]$ *randômicos*

Até que o critério de parada seja satisfeito

para cada par $(\underline{x}^p, \underline{y}^p)$ $p = 1, \dots, P$

calcular o acréscimo Δw_{ij}^p nas sinapses devido ao par p:
(ver algoritmo na sessão 2 Treinamento BP)

$$\underline{\tilde{y}}^p = \varphi(\underline{x}^p)$$

$$\epsilon_m^p = y_m^p - \tilde{y}_m^p \quad \text{e} \quad \frac{\partial \tilde{y}_m^p}{\partial w_{ij}} \quad \forall m, i, j$$

$$\Delta w_{ij}^p = -\alpha \frac{\partial \epsilon^p}{\partial w_{ij}} = 2\alpha \sum_{m=1}^M \epsilon_m^p \frac{\partial y_m^p}{\partial w_{ij}} \quad \forall i, j$$

outro par

calcular os acréscimos e atualizar as sinapses

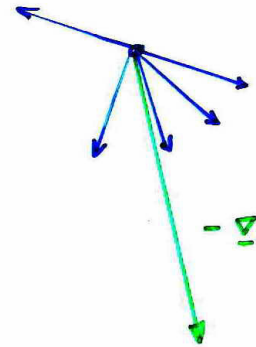
$$\Delta w_{ij}(n) = -\alpha E_p \Delta w_{ij}^p \quad w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \quad \forall i, j$$

$n = n+1$

fim do algoritmo

se $P \gg (P > \sim 200)$

$\Rightarrow$ muito lento



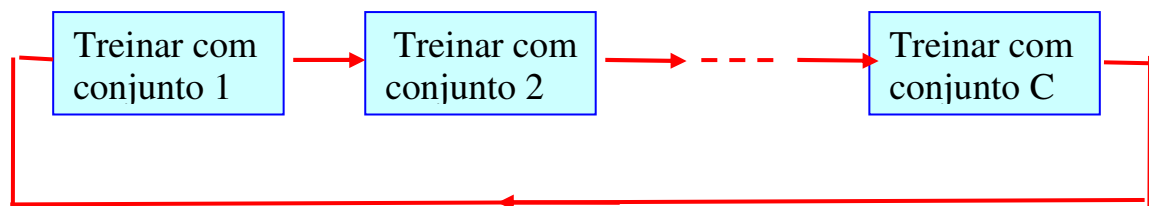
Processo por Lotes

Se $P > \sim 200$

Alocar os pares entrada-saída aleatoriamente em C conjuntos,

$$C \approx P / 200$$

Algoritmo



Lotes - Algoritmo

Inicializar $n = 1$; $\alpha = .1$; $w_i(1) \in [+.2, -.2]$ *randômicos*

Até que o critério de parada seja satisfeito

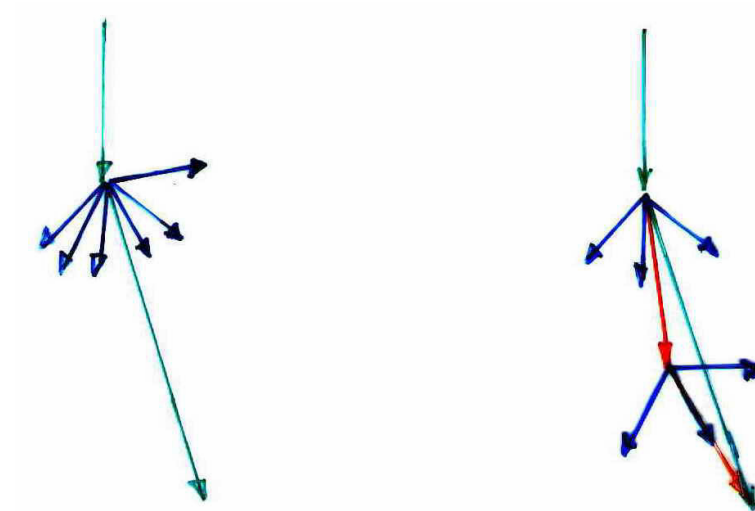
Treinar (em batelada, sem reinicializar) usando os pares do conjunto 1

Treinar (em batelada, sem reinicializar) usando os pares do conjunto 2

...

Treinar (em batelada, sem reinicializar) usando os pares do conjunto C

fim do algoritmo



Reduz o tempo

Preserva a generalização

diminui (quase nada) a estabilidade do treinamento

Processo Regra Delta - uso “on line”

Atualiza os parâmetros a cada par entrada-saída apresentado

o acréscimo nas sinapses é aplicado após cada par entrada-saída ser apresentado - é como usar lotes com um único par.

Treinamento adaptativo para sistemas variantes no tempo

Regra Delta - Algoritmo

Inicializar $n = 1$; $\alpha = .1$; $w_i(1) \in [+.2, -.2]$ *randômicos*

Até que o critério de parada seja satisfeito

para cada par $(\underline{x}^p, y^p)$ $p = 1, \dots, P$

calcular o acréscimo Δw_{ij}^p nas sinapses devido ao par p:
(ver algoritmo no cap 2 Treinamento BP)

$$\tilde{y}^p = \phi(\underline{x}^p)$$

$$\varepsilon_m^p = y_m^p - \tilde{y}_m^p \quad \text{e} \quad \frac{\partial \tilde{y}_m^p}{\partial w_{ij}} \quad \forall m, i, j$$

$$\Delta w_{ij}^p = -\alpha \frac{\partial \varepsilon^{p2}}{\partial w_{ij}} = 2\alpha \sum_{m=1}^M \varepsilon_m^p \frac{\partial y_m^p}{\partial w_{ij}} \quad \forall i, j$$

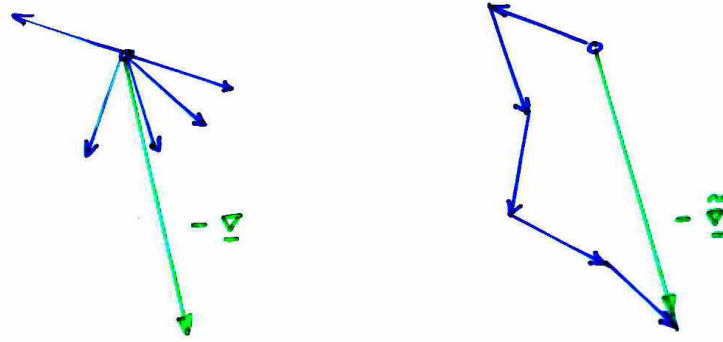
atualizar as sinapses

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}^p \quad \forall i, j$$

$n = n+1$

outro par

fim do algoritmo



Muito rápido, mas F oscila !

Normalmente usado com momento

Treinamento com Momento

Regra delta:
$$\Delta w_{i_D}(n) = 2\alpha \sum_{l=1}^m \varepsilon_l \frac{\partial \tilde{y}_l}{\partial w_i}$$

Momento
$$\Delta w_i(n) = \beta \Delta w_i(n-1) + (1-\beta) \Delta w_{i_D}(n)$$

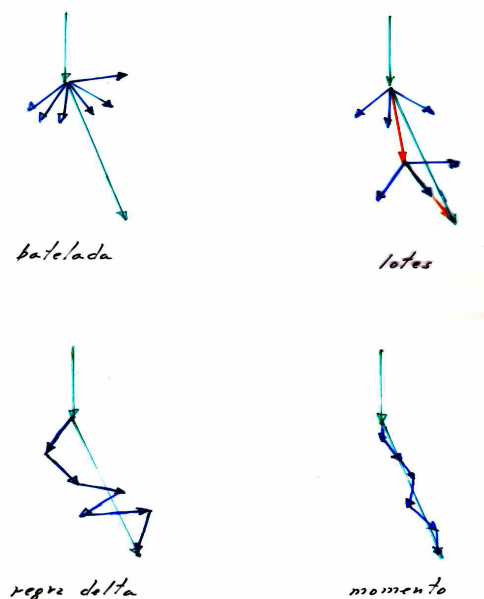
$$\beta_{\text{típico}} \approx .9$$

Intermediário entre treinamento regra delta e treinamento por lotes.

Cada “lote” é constituído pelos últimos

$$N_0 \approx \frac{4}{1-\beta} \text{ pares aplicados.}$$

Compensa a “instabilidade” do processo regra delta



Regra Delta com Momento - Algoritmo

Inicialização $n=1$; $\alpha = .1$ $\beta = .9$ $\Delta w_i^{RD}(0) = 0 \quad \forall i$

para as $i = 1, 2, \dots, V$ variáveis $w_i(1) \in [+.2, -.2]$ randômicos;

Até que o critério de parada seja satisfeito

para cada par $(\underline{x}^p, y^p)$ $p = 1, \dots, P$

calcular o acréscimo por regra delta (BP)

(ver algoritmo anterior)

$$\tilde{y}^p = \phi(\underline{x}^p)$$

$$\varepsilon_m^p = y_m^p - \tilde{y}_m^p \quad e \quad \frac{\partial \tilde{y}_m^p}{\partial w_i} \quad \forall m, i$$

$$\Delta w_i^{RD}(n) = 2\alpha \sum_{m=1}^M \varepsilon_m^p \frac{\partial y_m^p}{\partial w_i} \quad \forall i$$

calcular e aplicar o acréscimo com momento

$$\Delta w_i(n) = \beta \Delta w_i(n-1) + (1 - \beta) \Delta w_i^{RD}(n) \quad \forall i$$

$$w_i(n+1) = w_i(n) + \Delta w_i(n) \quad \forall i$$

$n=n+1$

outro par

fim do algoritmo

Detalhes do treinamento Regra Delta com momento

$$n = 1 \quad \Delta w_D(1)$$

$$\begin{aligned} \Delta w(1) &= \beta \Delta w(0) + (1-\beta) \Delta w_D(1) \\ &= (1-\beta) \Delta w_D(1) \end{aligned}$$

$$n = 2 \quad \Delta w_D(2)$$

$$\begin{aligned} \Delta w(2) &= \beta \Delta w(1) + (1-\beta) \Delta w_D(2) = \\ &= \beta(1-\beta) \Delta w_D(1) + (1-\beta) \Delta w_D(2) \end{aligned}$$

$$n = 3 \quad \Delta w_D(3)$$

$$\begin{aligned} \Delta w(3) &= \beta \Delta w(2) + (1-\beta) \Delta w_D(3) = \\ &= \beta^2(1-\beta) \Delta w_D(1) + \beta(1-\beta) \Delta w_D(2) + (1-\beta) \Delta w_D(3) \end{aligned}$$

$$n = 4 \quad \Delta w_D(4)$$

$$\Delta w(4) = \beta^3(1-\beta) \Delta w_D(1) + \beta^2(1-\beta) \Delta w_D(2) + \beta(1-\beta) \Delta w_D(3) + (1-\beta) \Delta w_D(4)$$

...

$$n \quad \Delta w_D(n)$$

$$\begin{aligned} \Delta w(n) &= \beta^{n-1}(1-\beta) \Delta w_D(1) + \beta^{n-2}(1-\beta) \Delta w_D(2) + \dots \\ &\quad \dots + \beta^{n-1}(1-\beta) \Delta w_D(i) + \dots + (1-\beta) \Delta w_D(n) \end{aligned}$$

$$\Delta w(n) = (1-\beta) \sum_{i=1}^n \Delta w_D(i) \beta^{n-i}$$

Contribuições em cada passo:

Média ponderada exponencialmente pelo atraso

$n - i$ = atraso da i -ésima entrada em relação ao instante atual n

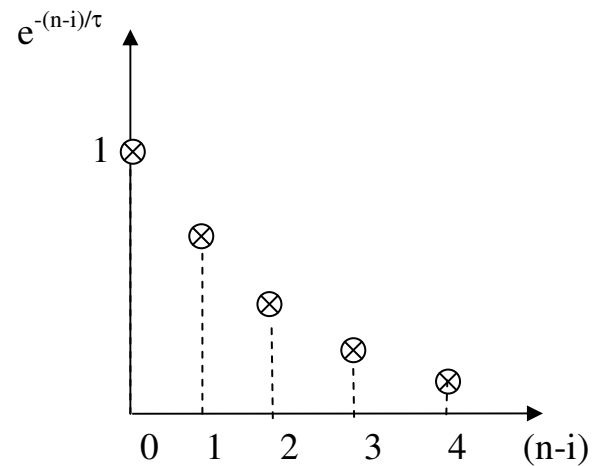
No passo n

$$\Delta w(n) = (1-\beta) \sum_{i=1}^n \Delta w_D(i) \beta^{n-i}$$

$$\beta^{n-i} = e^{(n-i) \ln \beta} = e^{-\frac{(n-i)}{\tau}}$$

$$\tau = -\frac{1}{\ln \beta} \approx \frac{1}{1-\beta} \quad \text{se } \beta \approx 1$$

$$\Delta w(n) = (1-\beta) \sum_{i=1}^n \Delta w_D(i) e^{-(n-i)/\tau}$$

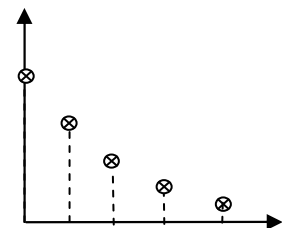


média ponderada exponencialmente pelo atraso

Contribuições significativas: $\text{peso} > .02 \implies n-i < 4 \tau$

Máximo atraso significativo (lote significativo):

$$N_0 \cong 4\tau \cong \frac{4}{1-\beta}$$



Contribuição total de $\Delta w_D(n)$:

$$\begin{aligned} \Delta w_D(n) (1-\beta) (1 + \beta + \beta^2 + \dots) &= \\ &= \Delta w_D(n) (1-\beta) \frac{1}{1-\beta} = \Delta w_D(n) \end{aligned}$$