Temas da 3ª semana – PSI3471-2020 – Prof Emilio

#5 (16/março – 2ªf) Foco da semana: aprendizado da Rede Neural MLP – O Gradiente descendente e a otimização de pesos sinápticos com base no conjunto de treino e EBP; dedução das fórmulas do EBP, em sala de aula em conjunto com os alunos:

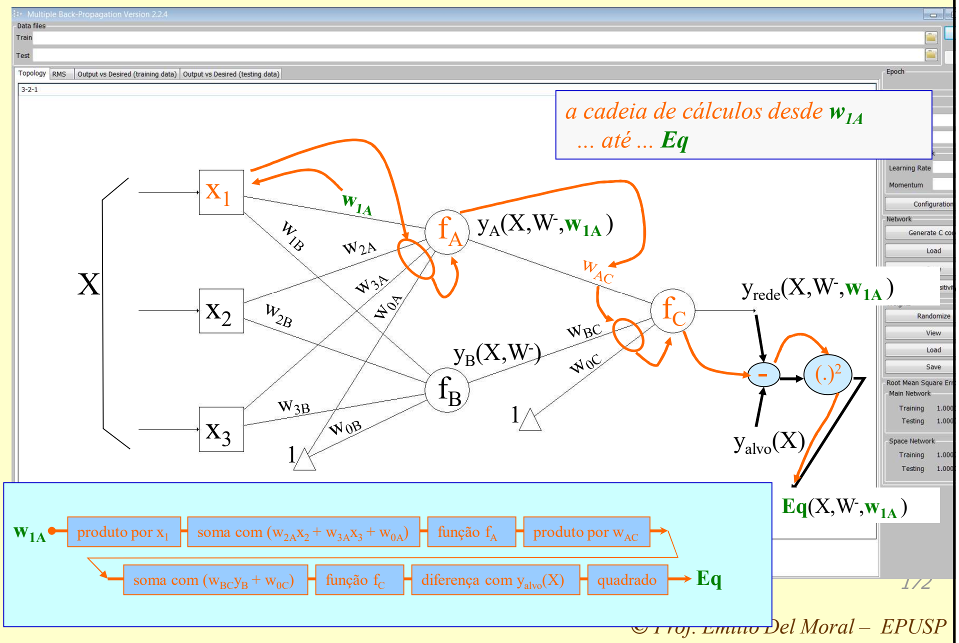
#6 (18/março – 4ªf) ... Discussão das extensões das deduções já feitas (para um peso no EBP) para os demais pesos sinápticos; redundâncias nos cálculos dos diversos pesos da rede neural e otimização do esforço computacional. Regra "Delta" de aprendizado de Widrow, para neurônio isolado; Aprendizado por EBP recursivo, camada a camada.

Ao comparamos as expressões das derivadas parciais de TODOS pesos da rede (Ex6 de treino), encontramos notáveis redundâncias:

Redundâncias do tipo 1: A cadeia de cálculos que é relevante ao cálculo da derivada parcial de Eqm com relação ao peso w_{1A} é muito semelhante à cadeia de cálculos que é relevante ao cálculo da derivada parcial de Eqm com relação ao peso w_{2A}. E o mesmo pode ser dito para a cadeia relevante para w_{3A}, e o mesmo também vale para w_{0A}.

Isto faz que as fórmulas de todas essas derivadas parciais sejam muito parecidas, e que seja possível reduzir computação na fase de aprendizado de pesos, aproveitando que boa parte dos cálculos são comum a todas essas cadeias. Temos redundância portanto, quando comparamos fórmulas para aqueles pesos que compõem a somatória ponderada de um dado nó neural.

O que muda na cadeia abaixo se em vez de w1A nós focarmos em w2A?
.... Apenas o comecinho da cadeia!!!



172

Ao comparamos as expressões das derivadas parciais de TODOS pesos da rede (Ex6 de treino), encontramos notáveis redundâncias:

"Temos redundância portanto, quando comparamos fórmulas para aqueles pesos que compõem a somatória ponderada de um dado nó neural." ....

E note que essa redundância é similar à redundância já detectada nos inícios das redes neurais, no aprendizado de um neurônio isolado pela "regra delta" de Widrow ...

173

"Regra Delta" (Widrow) para o aprendizado em nó simples:

Gradiente Descendente = Regra Δ p/ aprendizado de um nó:

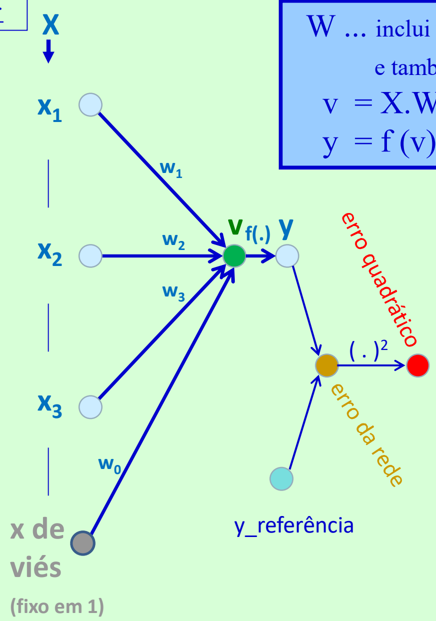
$\Delta w_i = - x_i . f'(v).2.\text{erro}$

... ou, vetorialmente:

$\Delta W = - X.f'(v).2.\text{erro}$

(ΔW = *correções de W*)

... em loop ...



W ... inclui w_1 a w_3

e também o viés (w_0)

$v = X.W = \Sigma\, w_i\, x_i$

$y = f(v)$

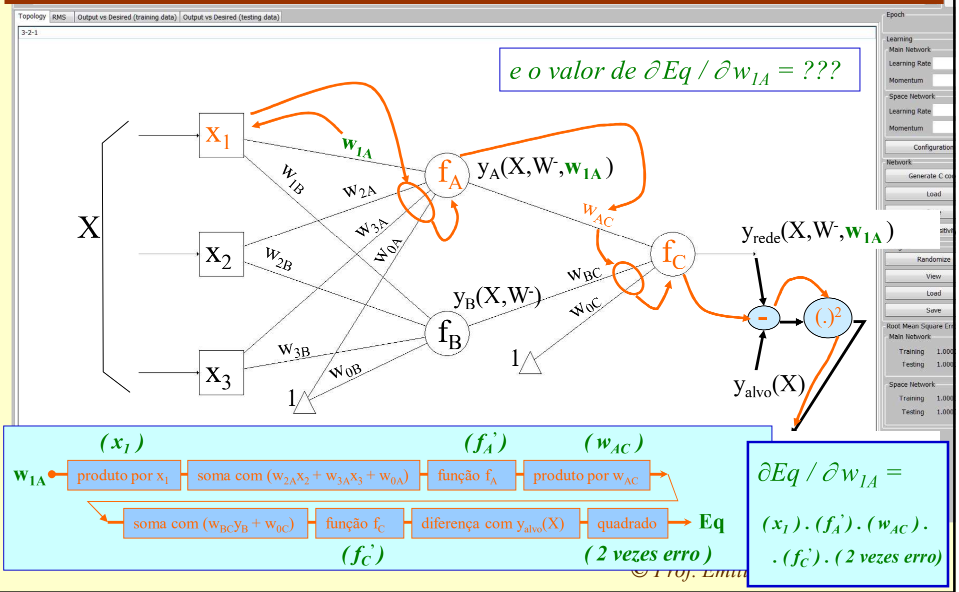
Grafo de cálculos para um único nó neural ...

Ao comparamos as expressões das derivadas parciais de TODOS pesos da rede (Ex6 de treino), encontramos notáveis redundâncias:

Redundâncias do tipo 2) A cadeia de cálculos que é relevante ao cálculo da derivada parcial de Eqm com relação ao peso w_{1A} tem uma parcela muito semelhante à cadeia de cálculos que é relevante ao cálculo da derivada parcial de Eqm com relação ao peso w_{AC}.

Temos aqui redundância de cálculos quando vamos de uma cama para outra.

A expressão p/ $\partial Eq / \partial w_{AC}$ (peso da 2ª camada) é similar a boa parte da expressão adiante, obtida para $\partial Eq / \partial w_{1A}$

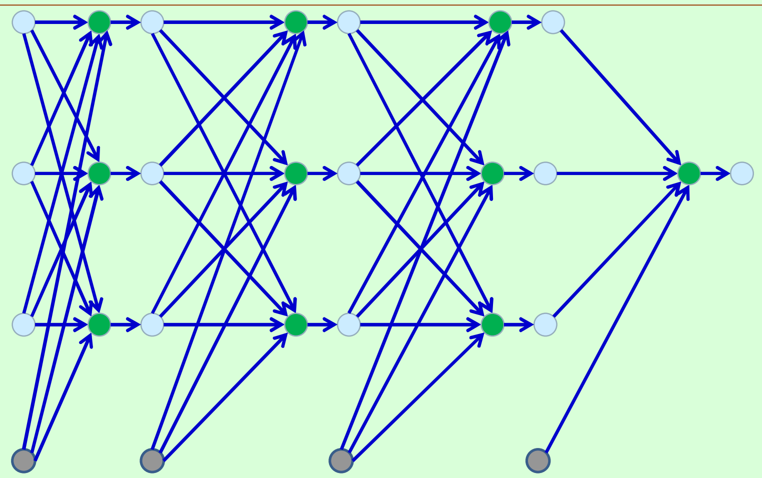


Ao comparamos as expressões das derivadas parciais de TODOS pesos da rede (Ex6 de treino), encontramos notáveis redundâncias:

Os dois tipos de redundâncias nos cálculos das adaptações de pesos sinápticos destacadas nos slides anteriores, são as bases para a estratégia de cálculo recursivo do Error Back Propagation, conforme detalhado nos slides mais adiante

Tal estratégia de cálculo é extremamente importante em redes neurais com muitas camadas: Os cálculos das adaptações dos pesos sinápticos são feitos de forma recursiva, primeiro realizando-se os cálculos para os pesos da camada de saída, depois são feitos os cálculos para os pesos da camada anterior (com reaproveitamento parcial de cálculos já feitos anteriormente), e assim sucessivamente, até chegarmos aos cálculos para os pesos da primeira camada.

Rede Neural com muitas camadas



Prof. Emilio Del Moral Hernandez

178

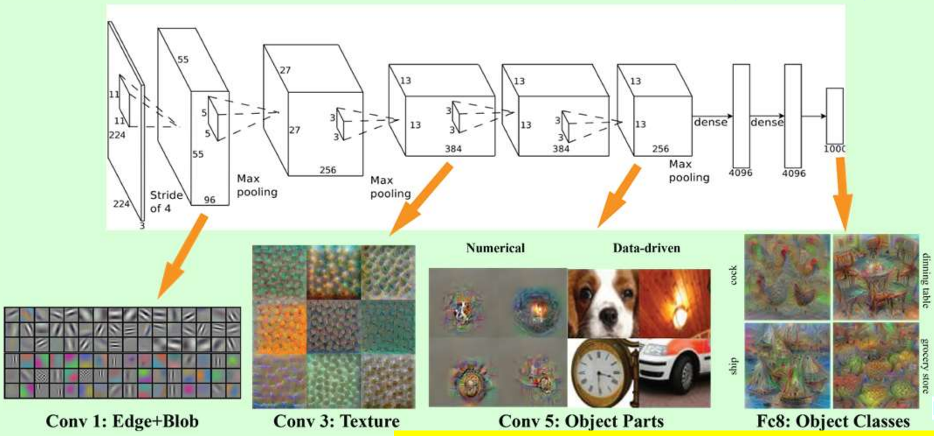
Redes Neurais Convolucionais / Profundas são exemplos importantes de estruturas com um grande número de camadas

Classificação em Redes Neurais Convolucionais

(slide elaborado por Mark Cappello Ferreira de Sousa)

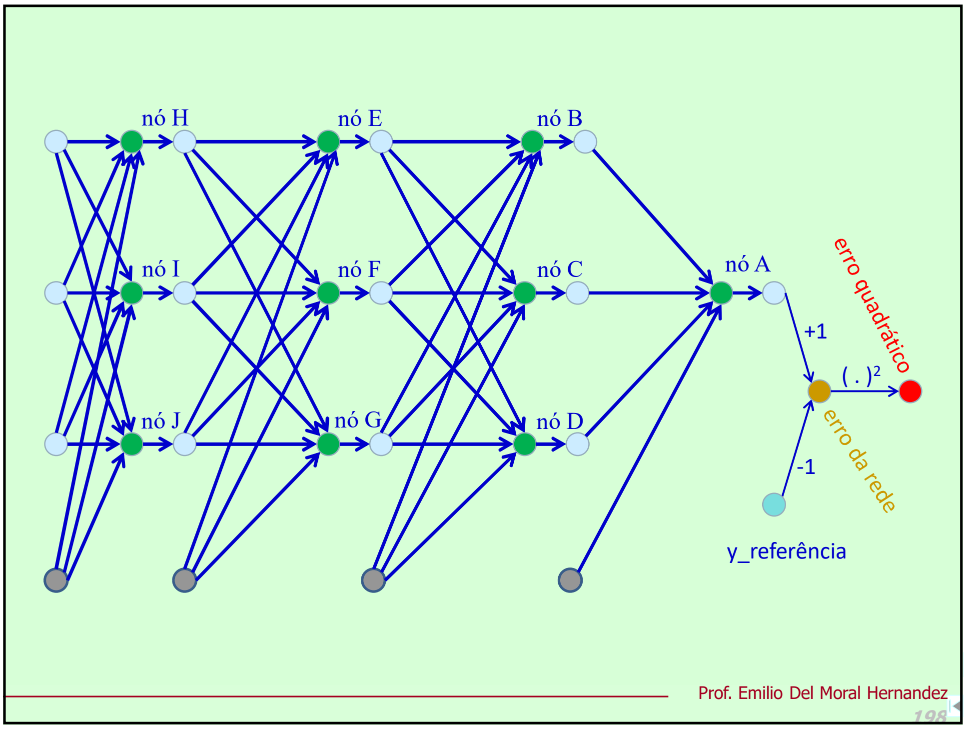
Convoluções
(Extração de características)

Fully Connected
(MLP)

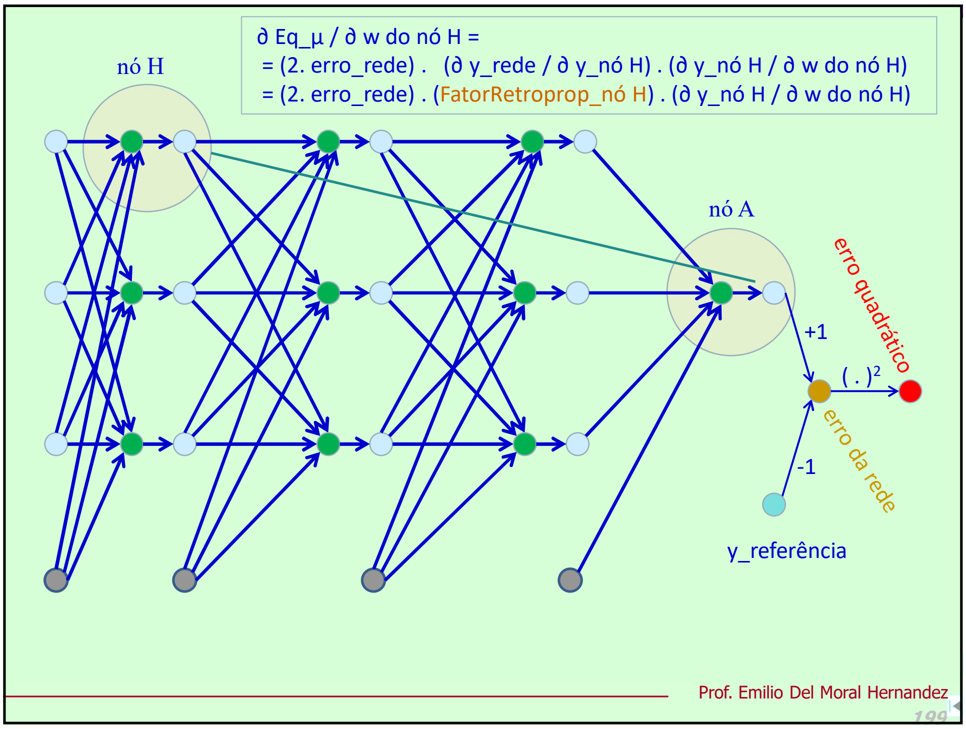


figuras ilustrativas – extraídas da internet

181

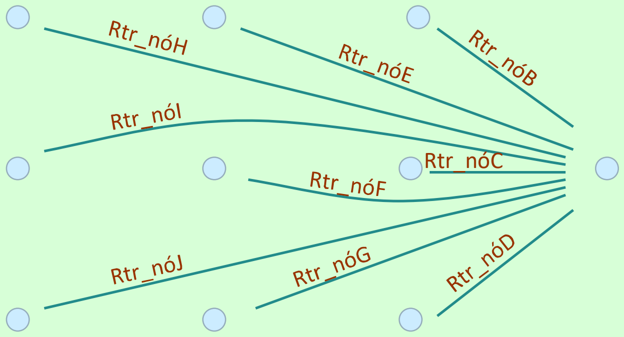

nó H
nó E
nó B
nó I
nó F
nó C
nó A
nó J
nó G
nó D
+1
-1
(.)²
erro quadrático
erro da rede
y_referência
Prof. Emilio Del Moral Hernandez
198

198


∂ Eq_μ / ∂ w do nó H =
= (2. erro_rede) . (∂ y_rede / ∂ y_nó H) . (∂ y_nó H / ∂ w do nó H)
= (2. erro_rede) . (FatorRetroprop_nó H) . (∂ y_nó H / ∂ w do nó H)
nó H
nó A
+1
-1
(.)²
erro quadrático
erro da rede
y_referência
Prof. Emilio Del Moral Hernandez
199

199

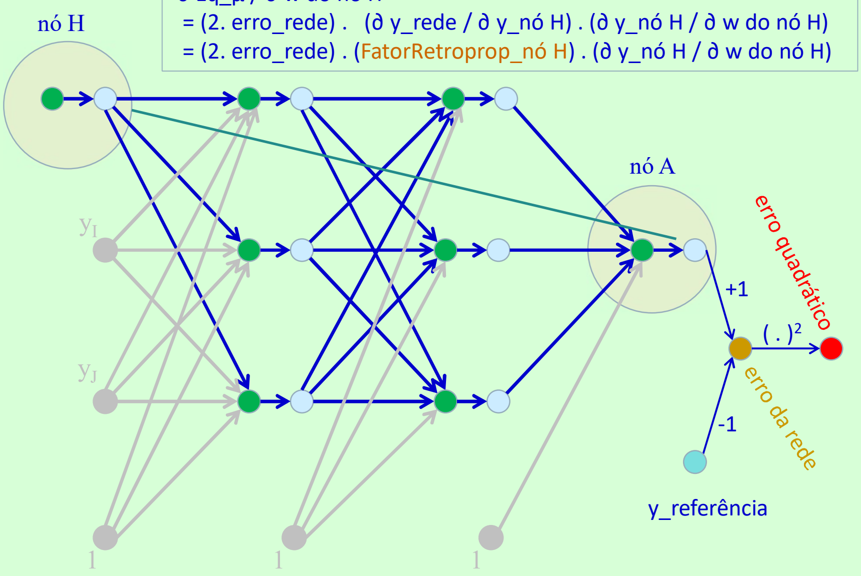
Temos um retropropagador para cada no´ da rede, a excessao do no´ de saida “nó A”: o nó de saída não necessita do retropropagador, já que o erro da saída da rede é o próprio erro desse nó; aliás, se tentar calcular (∂ y_rede / ∂ y_nó) para ele, chega-se obviamente a Rtr_nóA = 1.



∂ Eq_μ / ∂ w para w nó *KK* =
= (2. erro_rede) . (∂ y_rede / ∂ y_nó *KK*) . (∂ y_nó *KK* / ∂ w)
= (2. erro_rede) . (FatorRetroprop_nó *KK*) . (∂ y_nó *KK* / ∂ w)

200

∂ Eq_μ / ∂ w do nó H =
= (2. erro_rede) . (∂ y_rede / ∂ y_nó H) . (∂ y_nó H / ∂ w do nó H)
= (2. erro_rede) . (FatorRetroprop_nó H) . (∂ y_nó H / ∂ w do nó H)

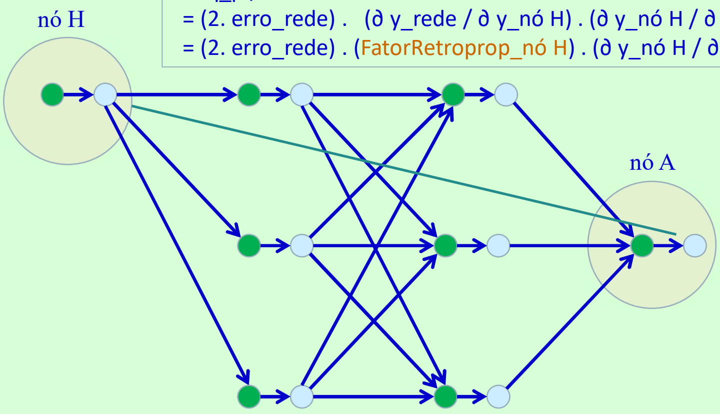


201

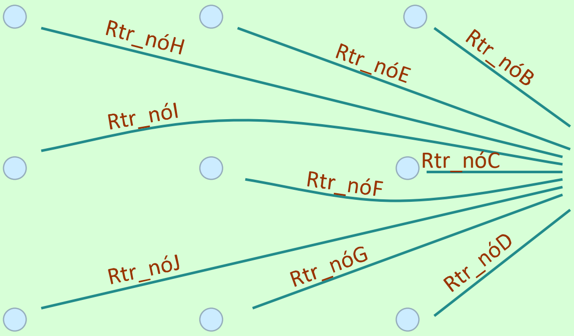
∂ Eq_μ / ∂ w do nó H =

= (2. erro_rede) . (∂ y_rede / ∂ y_nó H) . (∂ y_nó H / ∂ w do nó H)

= (2. erro_rede) . (FatorRetroprop_nó H) . (∂ y_nó H / ∂ w do nó H)



Temos um retropropagador para cada no´ da rede, a excessao do no´ de saida "nó D": o nó de saída não necessita do retropropagador, já que o erro da saída da rede é o próprio errodesse nó; aliás, se tentar calcular (∂ y_rede / ∂ y_nó) *para ele, chega-se obviamente a Rtr_nóD = 1.*



∂ Eq_μ / ∂ w para w nó NN =

= (2. erro_rede) . (∂ y_rede / ∂ y_nó NN) . (∂ y_nó NN / ∂ w)

= (2. erro_rede) . (FatorRetroprop_nó NN) . (∂ y_nó NN / ∂ w)

Definindo novas variáveis: uma nova variável auxiliar associada a cada nó, chamada erro do nó

... Revisitemos a expressão para ∂ Eq_μ / ∂ w , reagrupando termos

∂ Eq_μ / ∂ w =

= (2. erro_rede). (∂ y_rede / ∂ y_nó). (∂ y_nó / ∂ w)

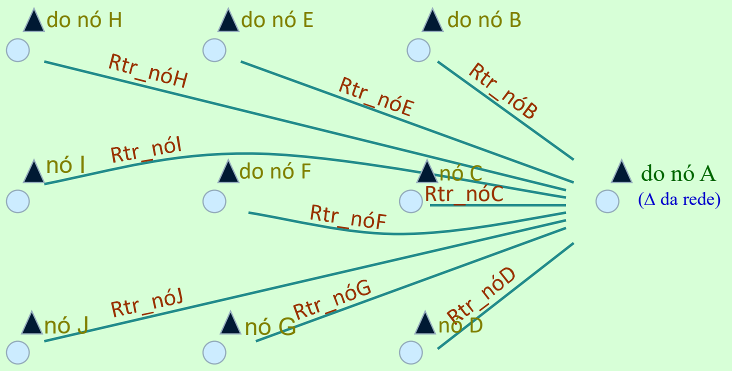
definição

= (2. erro_rede). (FatorRetroprop_nó). (∂ y_nó / ∂ w)

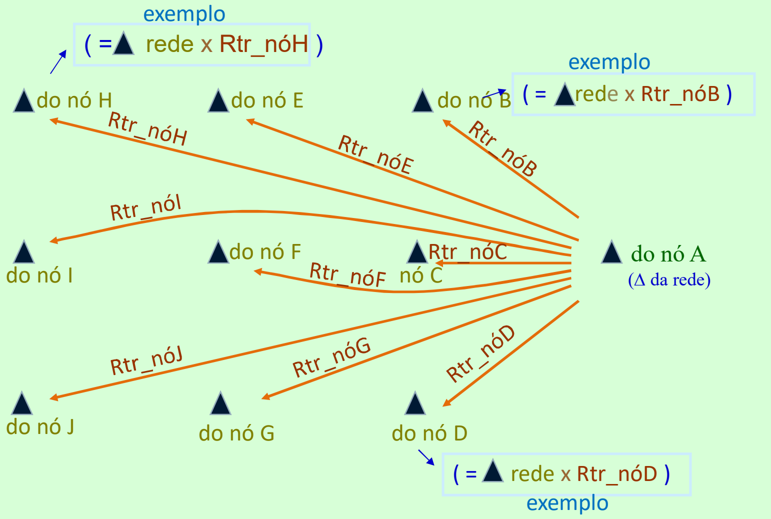
= (2. [erro_rede . FatorRetroprop_nó] . (∂ y_nó / ∂ w))

Definição de ***erro de nó***

(= erro de rede retroprogado ao nó)



Os triângulos representam os erros associados a cada nó

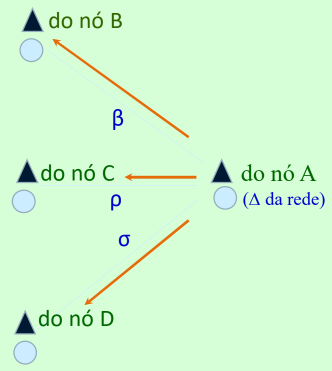


Prof. Emilio Del Moral Hernandez

206

206

<u>*Retropogação de erro aos nós da penúltima camada*</u>



definição de β, ρ e σ:

$\beta : \partial y_A / \partial y_B = (w_{BA} \cdot f_A')$

$\rho : \partial y_A / \partial y_C = (w_{CA} \cdot f_A')$

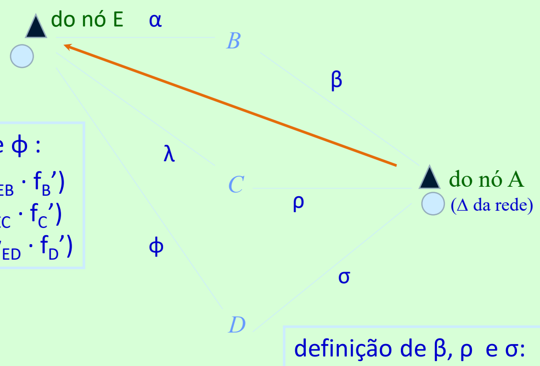
$\sigma : \partial y_A / \partial y_D = (w_{DA} \cdot f_A')$

Prof. Emilio Del Moral Hernandez

207

207

Retropogação de erro envolvendo cadeia com trifurcação



definição de α, λ e ϕ :

$\alpha : \partial y_B / \partial y_E = (w_{EB} \cdot f_B')$

$\lambda : \partial y_C / \partial y_E = (w_{EC} \cdot f_C')$

$\phi : \partial y_D / \partial y_E = (w_{ED} \cdot f_D')$

definição de β, ρ e σ:

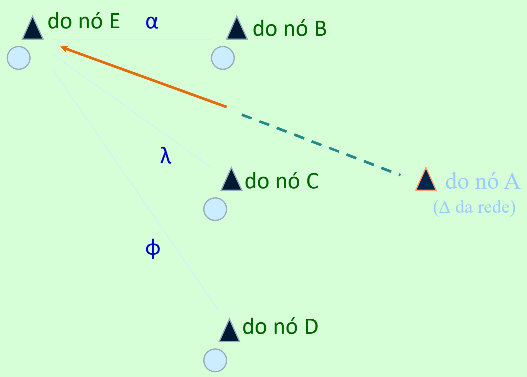
$\beta : \partial y_A / \partial y_B = (w_{BA} \cdot f_A')$

$\rho : \partial y_A / \partial y_C = (w_{CA} \cdot f_A')$

$\sigma : \partial y_A / \partial y_D = (w_{DA} \cdot f_A')$

erro E = (α· erro da rede · β) + (λ· erro da rede . ρ) +(ϕ·erro da rede . σ)

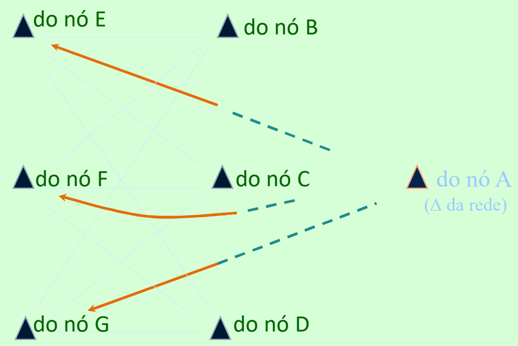
Retropogação de erro explorando recursão



erro E = α·(erro da rede · β) + λ·(erro da rede . ρ) + ϕ·(erro da rede . σ)
= α·(erro B) + λ·(erro C) + ϕ·(erro D)

erro E = (α· erro da rede · β) + (λ· erro da rede . ρ) +(ϕ·erro da rede . σ)

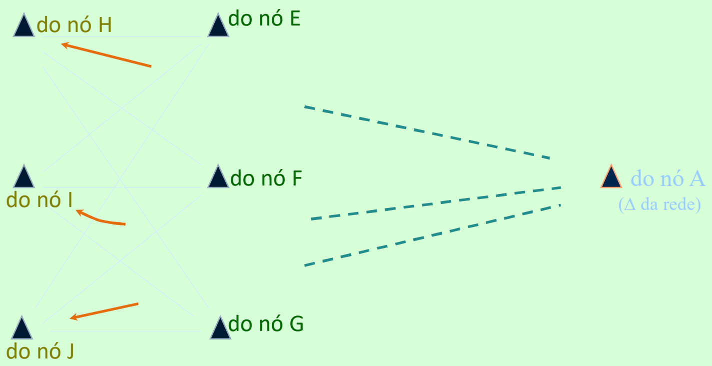
Retropogação de erro explorando recursão



Prof. Emilio Del Moral Hernandez

210

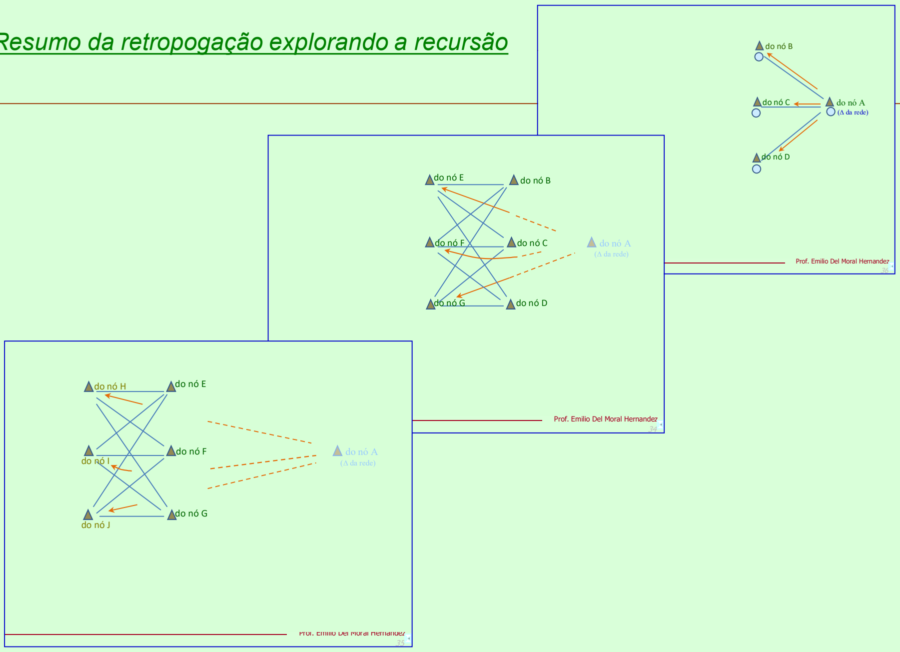
Retropogação de erro explorando recursão



Prof. Emilio Del Moral Hernandez

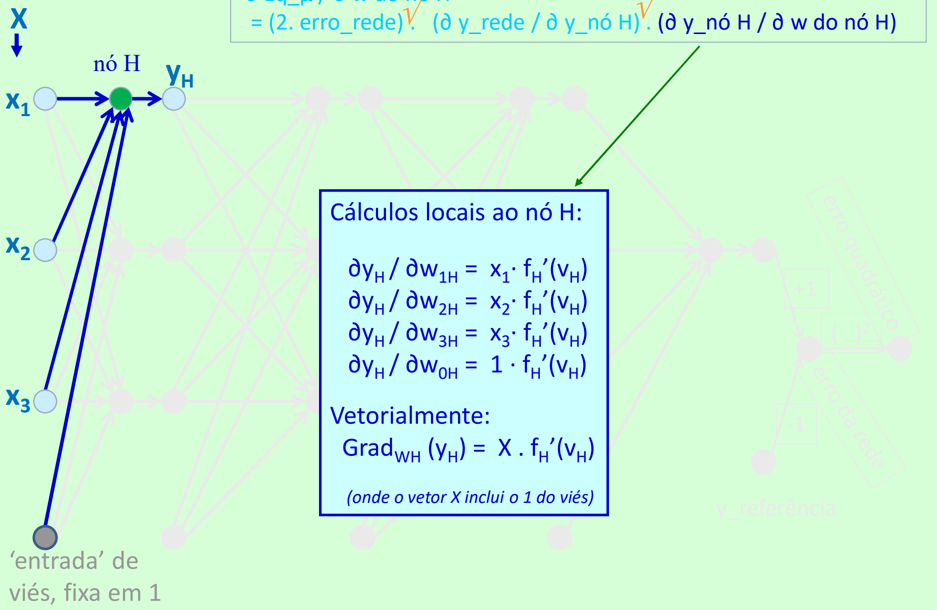
211

Resumo da retropogação explorando a recursão



213

∂ Eq_µ / ∂ w do nó H =
= (2. erro_rede) √ . (∂ y_rede / ∂ y_nó H) √ . (∂ y_nó H / ∂ w do nó H)



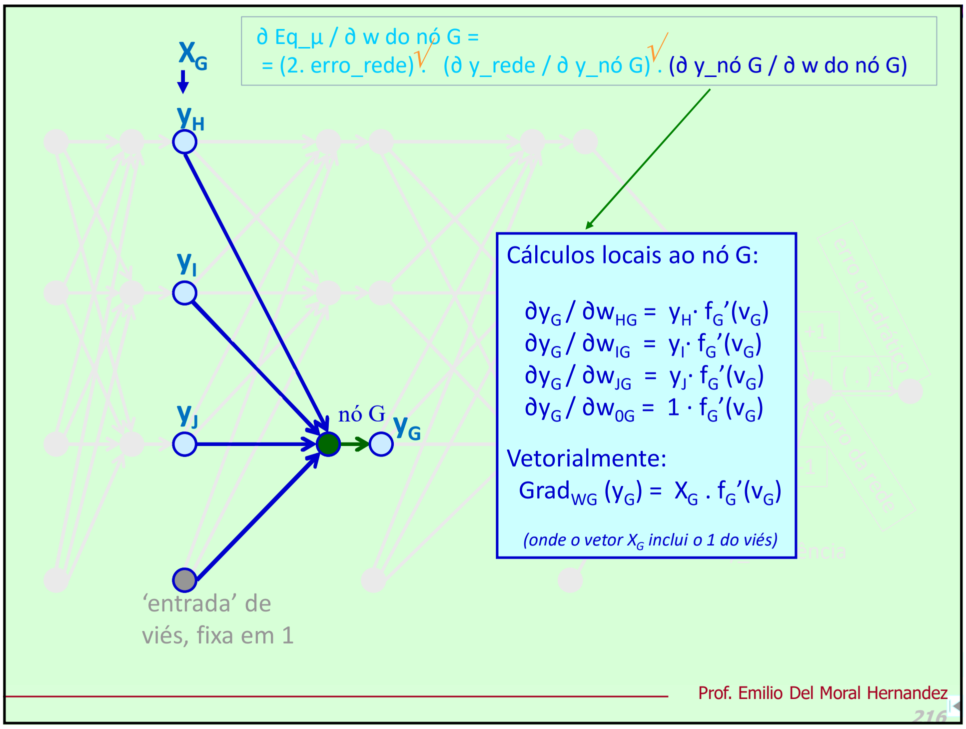
Cálculos locais ao nó H:

$\partial y_H / \partial w_{1H} = x_1 \cdot f_H'(v_H)$
$\partial y_H / \partial w_{2H} = x_2 \cdot f_H'(v_H)$
$\partial y_H / \partial w_{3H} = x_3 \cdot f_H'(v_H)$
$\partial y_H / \partial w_{0H} = 1 \cdot f_H'(v_H)$

Vetorialmente:
$Grad_{WH}(y_H) = X \, . \, f_H'(v_H)$

(onde o vetor X inclui o 1 do viés)

215



∂ Eq_μ / ∂ w do nó G =
= (2. erro_rede)√. (∂ y_rede / ∂ y_nó G)√. (∂ y_nó G / ∂ w do nó G)
X_G
y_H
y_I
y_J
nó G
y_G
'entrada' de viés, fixa em 1
Cálculos locais ao nó G:
$\partial y_G / \partial w_{HG} = y_H \cdot f_G'(v_G)$
$\partial y_G / \partial w_{IG} = y_I \cdot f_G'(v_G)$
$\partial y_G / \partial w_{JG} = y_J \cdot f_G'(v_G)$
$\partial y_G / \partial w_{0G} = 1 \cdot f_G'(v_G)$
Vetorialmente:
$Grad_{WG} (y_G) = X_G \,.\, f_G'(v_G)$
(onde o vetor X_G inclui o 1 do viés)
Prof. Emilio Del Moral Hernandez
216

216