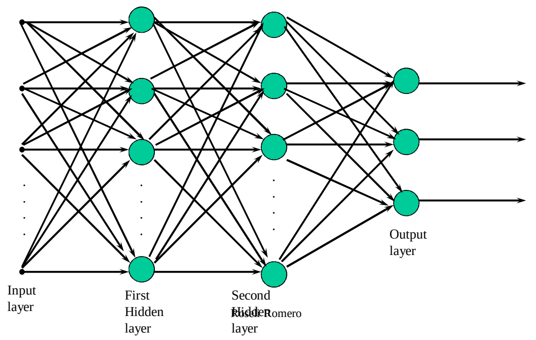
SCE 5809 – REDES NEURAIS

REDE NEURAL DO TIPO MULTI-CAMADAS

Profa. Roseli Romero

Modelo de Rede Neural com Multiplas Camadas



II - Algoritmo Back-Propagation

$$\text{Out}(x) = g\left(\Sigma_j\, W_j\, g(\Sigma_i\, w_i\, x_i) \right)$$

Isto é uma função não-linear
de uma combinação linear
de funções não lineares
de combinações lineares das entradas

II - Algoritmo BackPropagation

OBJETIVO

- Encontrar um conjunto de pesos $\{W_j\},\{w_{jk}\}$, para
  MINIMIZAR $\Sigma_i\ (y_i - \text{Out}(\underline{x}_i))^2$

pelo metodo do “gradiente descent”.

OBS: Convergência para um MINIMO global não é garantida.

Na prática: não é problema!!!

II - Algoritmo Back-Propagation

$$\Delta w_{jk} = -\eta \delta_j{}^p out_k{}^p$$

- Se o neurônio está na camada de saída

$$\delta_{pj} = (y_j{}^p - out_j{}^p)\phi'(net_j{}^p) \qquad net_j^p = \sum_k w_{jk} out_k{}^p$$

- Se o neurônio está na camada oculta

$$\delta_{pj} = \phi'(net_j{}^p)\sum_k \delta_k{}^p w_{kj}$$

Obtenção do Alg.Backpropagation

$$\Delta w_{jk} = -\frac{\partial E}{\partial w_{jk}}$$

$$E(n) = 0.5\sum_{j \in C} e_j^2(n)$$

onde conjunto C inclui todos os neuronios na camada de saída. Seja N denotar o no. total de padrões no conj. Treinamento. Então:

$$E_{av} = \frac{1}{N}\sum_{n=1}^{N} E(n)$$

Obtenção do alg. BP

$$v_j = \sum_{i=0}^{p} w_{ji} y_i(n) \qquad e_j(n) = d_j(n) - y_j(n)$$

Onde p é o no. total de entradas. O sinal de saída é:

$$y_j(n) = \varphi(v_j(n))$$

Acordo com a regra da cadeia, obtemos:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

Obtenção do Alg. BP

$$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n) \qquad \frac{\partial e_j(n)}{\partial y_j(n)} = -1 \qquad \frac{\partial e_j(n)}{\partial y_j(n)} = \varphi'(v_j(n))$$

$$\frac{\partial v_j(n)}{\partial w_{ij}(n)} = y_i(n)$$

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n).\varphi'(v_j(n))y_i(n)$$

Portanto:

$$\Delta w_{ij} = -\frac{\partial E(n)}{\partial w_{ji}(n)} \Rightarrow \Delta w_{ij} = \eta \delta_j(n) y_i(n);$$

$$\delta_j(n) = e_j(n)\varphi'(v_j(n)) \qquad (*)$$

Obtenção do Alg. BP

- Se o neuronio j está na camada de saída:

  O calculo de delta é direto

- Se o neuronio j está na camada intermediária

$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial E(n)}{\partial y_j(n)}\varphi'(v_j(n)) \qquad (0)$$

$$E(n) = 0.5\sum_{k\in C} e_k^2(n)$$

Em algum evento, diferenciando a ultima equação em rel. a y_j

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_k e_k.\frac{\partial e_k(n)}{\partial y_j(n)} = \sum_k e_k.\frac{\partial e_k(n)}{\partial v_k(n)}.\frac{\partial v_k(n)}{\partial y_j(n)} \qquad (1)$$

Roseli Romero

Obtenção do Alg. BP

Lembrando que como k está na saída, o erro e_k

$$e_k(n) = d_k(n) - y_k(n) = d_k(n) - \varphi(v_k(n))$$

Então:

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi'(v_k(n)) \qquad (2)$$

Como:

$$v_k = \sum_{j=0}^{p} w_{kj}\, y_j(n)$$

Tem-se:

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n) \qquad (3)$$

Obtenção do Alg. BP

Substituindo (2) e (3) em (1), obtém-se:

$$\frac{\partial E(n)}{\partial y_j(n)} = -\sum_k e_k.\varphi'(v_k(n)).w_{kj}(n)$$

Pela equação (*), tem-se:

$$\frac{\partial E(n)}{\partial y_j(n)} = -\sum_k \delta(n) w_{kj}(n)$$

Finalmente, substituindo esta eq. Na eq. (0), obtém-se:

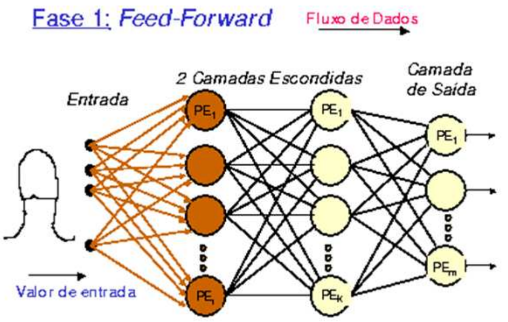
$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)}\varphi'(v_j(n)) = \varphi'(v_j(n))\sum_k \delta_k(n) w_{kj}(n)$$

MULTI-LAYER PERCEPTRON

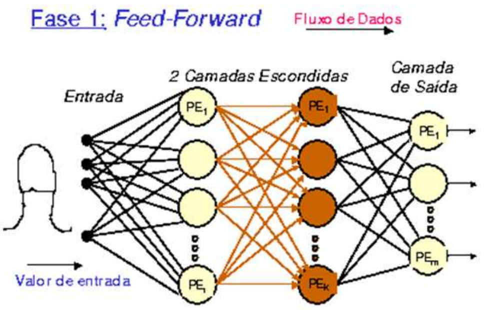
- Redes de apenas uma camada só representam funções linearmente separáveis
- Redes de múltiplas camadas solucionam essa restrição
- O desenvolvimento do algoritmo Back-Propagation foi um dos motivos para o ressurgimento da área de redes neurais em 1986 por Rumelhart et.

Roseli Romero

PROCESSO DE APRENDIZADO

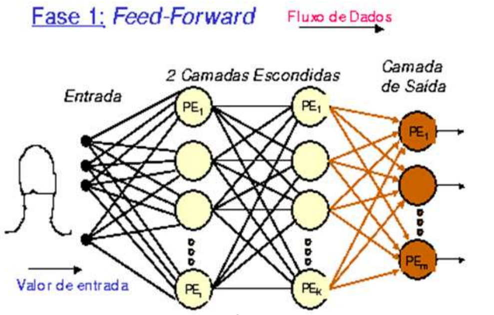


PROCESSO DE APRENDIZADO



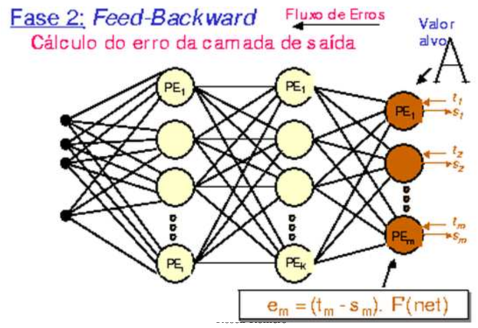
Roseli Romero

PROCESSO DE APRENDIZADO

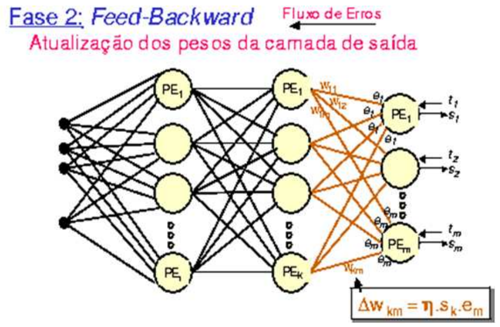


Roseli Romero

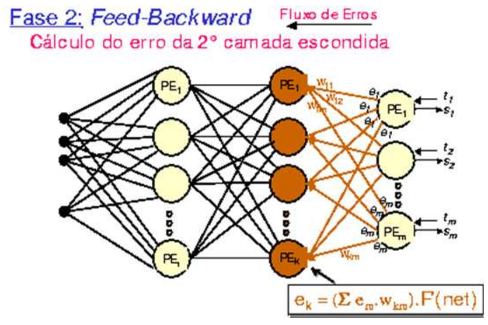
PROCESSO DE APRENDIZADO



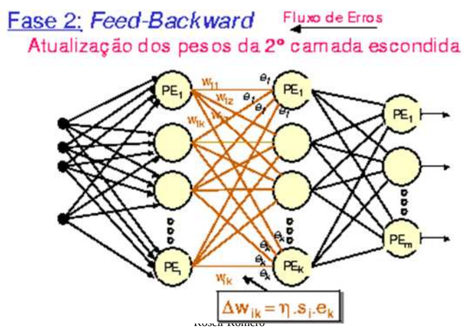
PROCESSO DE APRENDIZADO



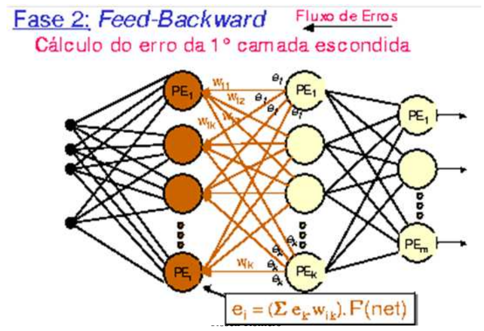
PROCESSO DE APRENDIZADO



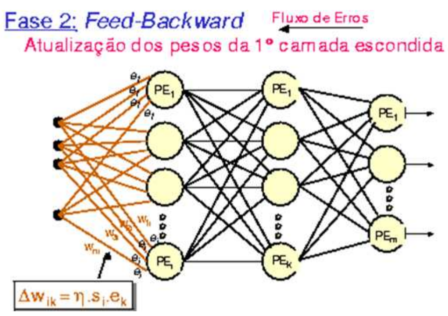
PROCESSO DE APRENDIZADO



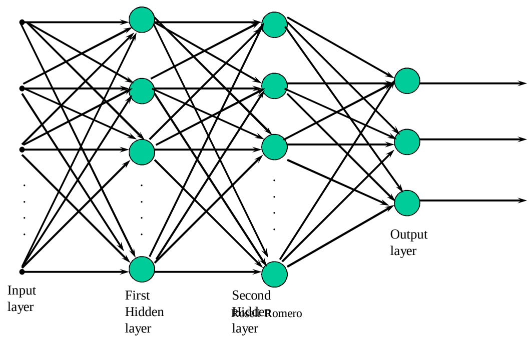
PROCESSO DE APRENDIZADO



PROCESSO DE APRENDIZADO



Modelo de Rede Neural com Multiplas Camadas



ALGORITMO

Este procedimento de aprendizado é repetido diversas vezes, até que *para todos processadores de camada de saída e para todos padrões de treinamento*, o erro seja menor do que o especificado.

ALGORITMO

Inicialização: pesos iniciados com valores aleatórios e pequenos ([-1,1])

Treinamento

Repita

Considere um novo padrão de entrada x_i e seu respectivo vetor de saída t_i desejado do conj. de treinamento;

Repita

- Apresentar o par (x_i t_i) **(modo padrão)**
- calcular as saídas dos processadores, começando da primeira camada escondida até a camada de saída;
- calcular o erro na camada de saída
- atualizar os pesos de cada processador, começando pela camada de saída, até a camada de entrada;

até que erro quadrático médio (para esse padrão) seja<= tol1.

até que o erro quadrático médio seja <= tol2 para todos os padrões de conjunto de treinamento

Exercício

- Faça uma iteração do algoritmo BP (ida e volta) para treinar uma rede neural multicamadas a aprender a função OU-EXCLUSIVO