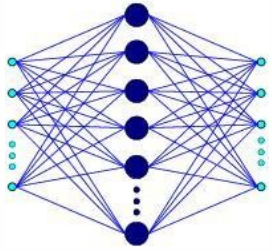
Ciência da Computação

REDE NEURAIS

Semestre: 2010/1

AULA 13



Professor: Max Pereira

http://paginas.unisul.br/max.pereira

UNISUL

Aqui seu futuro acontece

Conteúdo

- Redes MLP (Backpropagation)
- Exercícios

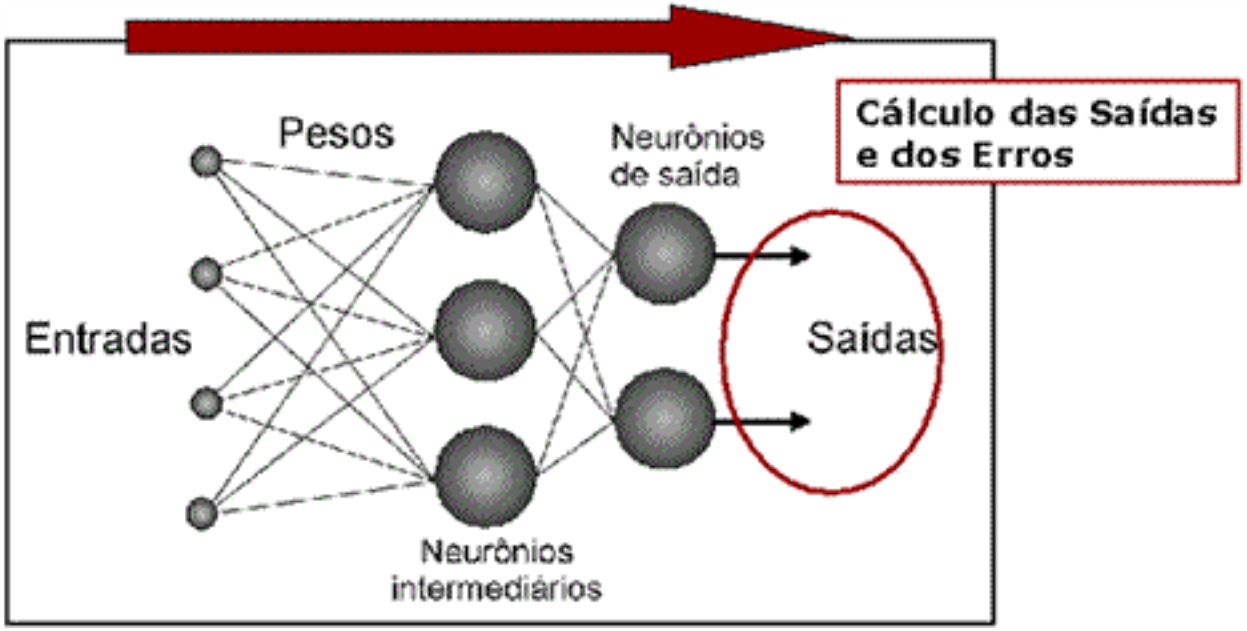
Redes Backpropagation

- Desenvolvida de forma independente por vários pesquisadores
- Em 1974, Paul Werbos desenvolveu o algoritmo durante sua tese de doutorado.
- Técnica de aprendizado supervisionado mais utilizada para redes neurais
- Rede direta multi-camada
- Solução para classificação de padrões não-lineares.

Redes Backpropagation

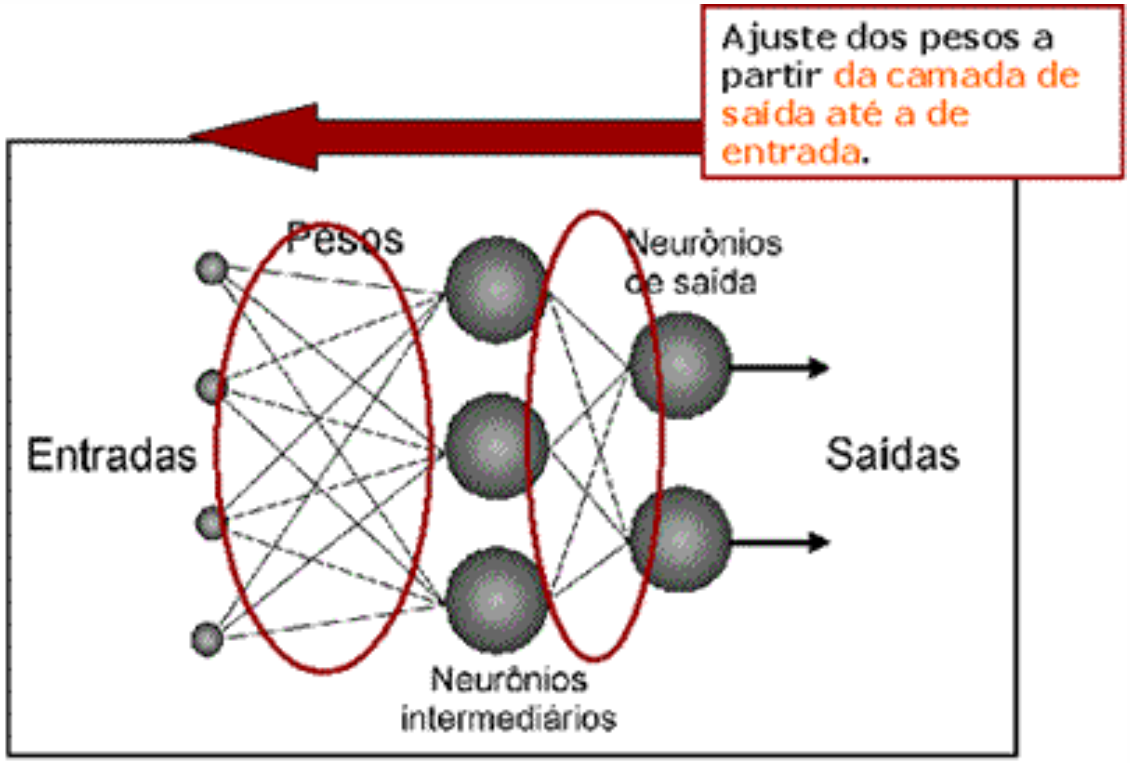
- O treinamento da rede envolve três estágios: o estímulo dos padrões de entrada (propagação), o cálculo e retropropagação do erro e o ajuste dos pesos.
- Após o treinamento, a aplicação da rede envolve apenas o processamento da fase de propagação.

Treinamento



Propagação

Treinamento



Retropropagação

Arquitetura

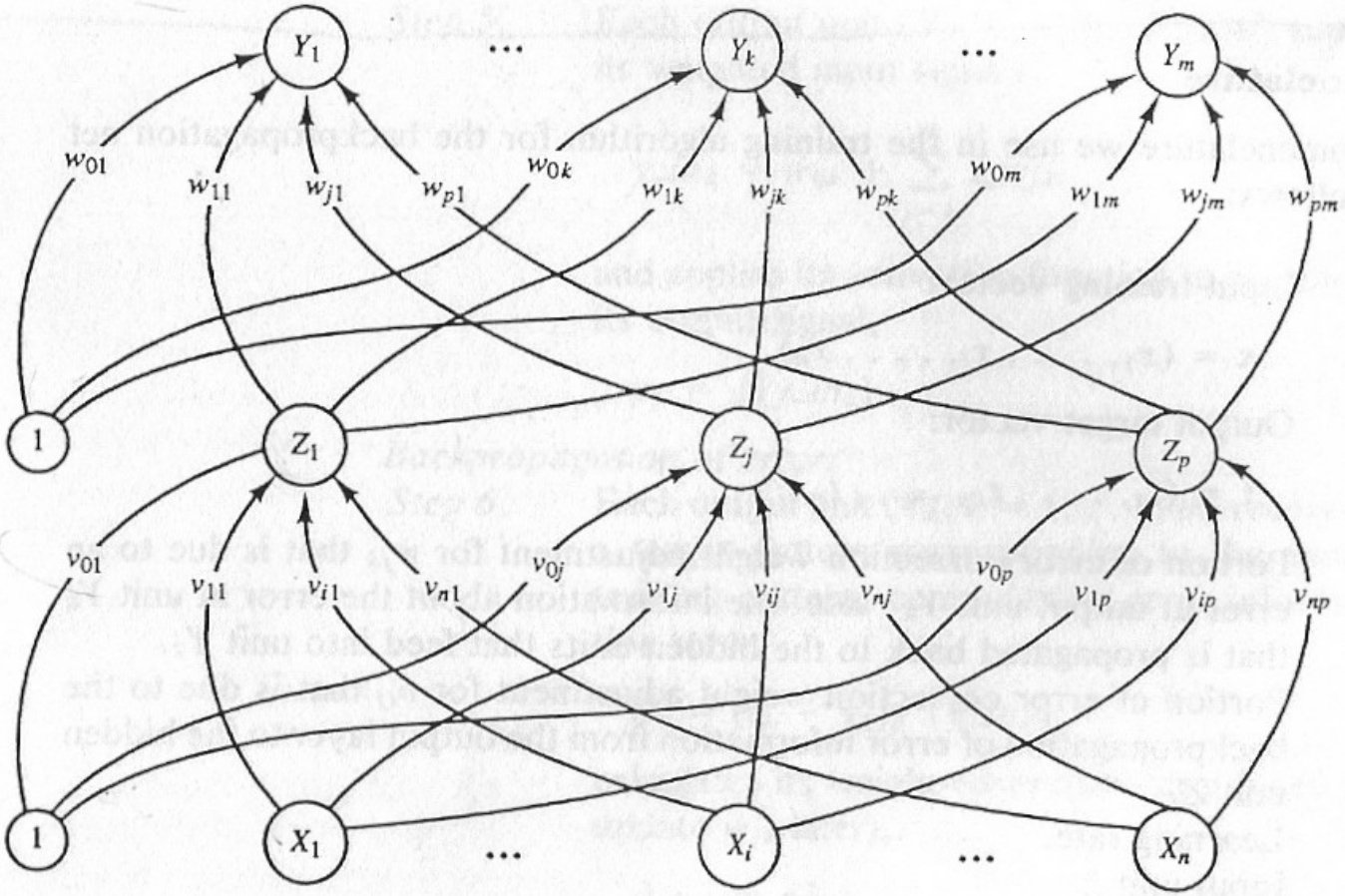


Figure 6.1 Backpropagation neural network with one hidden layer.

Algoritmo

- Durante o estágio de propagação, cada unidade (x_i) recebe um sinal de entrada e envia o sinal para cada unidade da camada intermediária (z_j).
- Cada unidade da camada intermediária calcula sua ativação e manda seu sinal para cada unidade da camada de saída (y_k).

Algoritmo

- Durante o treinamento cada unidade da camada de saída compara sua própria ativação (y_k) com o seu valor alvo (t_k) para determinar o erro associado.
- Com base nesse erro, o fator δ_k é calculado. δ_k é usado para distribuir o erro da unidade de saída (y_k) para todas as unidades da camada anterior (camada intermediária).

Algoritmo

- O δ_k também é usado, para posteriormente, ajustar os pesos entre as camadas de saída e intermediária.
- Da mesma forma, o fator δ_j é calculado para cada unidade da camada intermediária (z_j) e é utilizado para ajustar os pesos entre a camada intermidária e a camada de entrada.

Algoritmo

- Após todos os fatores δ serem calculados, os pesos para todas as camadas são ajustados simultaneamente.
- Os ajuste dos pesos w_{jk} usa o fator δ_k e a ativação z_j. O ajuste dos pesos v_{ij} usa o fator δ_j e a ativação x_i.

Função de Ativação

- Uma das mais típicas funções de ativação para as redes backpropagation é a função sigmóide binária:

$$f(x) = \frac{1}{1 + \exp(-x)}$$

com,

$$f'(x) = f(x)[1 - f(x)]$$

Função de Ativação

- Outra função comum é a função sigmóide bipolar:

$$f(x) = \frac{2}{1 + \exp(-x)} - 1$$

com,

$$f'(x) = \frac{1}{2}[1 + f(x)][1 - f(x)]$$

Step 0. Initialize weights.
(Set to small random values).

Step 1. While stopping condition is false, do Steps 2–9.

Step 2. For each training pair, do Steps 3–8.

Feedforward:

Step 3. Each input unit (X_i, $i = 1, \ldots, n$) receives input signal x_i and broadcasts this signal to all units in the layer above (the hidden units).

Step 4. Each hidden unit (Z_j, $j = 1, \ldots, p$) sums its weighted input signals,

Bias

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij},$$

applies its activation function to compute its output signal,

$$z_j = f(z\_in_j),$$

and sends this signal to all units in the layer above (output units).

Step 5. Each output unit (Y_k, $k = 1, \ldots, m$) sums its weighted input signals,

Bias

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

and applies its activation function to compute its output signal,

$$y_k = f(y\_in_k).$$

Backpropagation of error:

Step 6. Each output unit (Y_k, $k = 1, \ldots, m$) receives a target pattern corresponding to the input training pattern, computes its error information term,

$$\delta_k = (t_k - y_k) f'(y\_in_k),$$

calculates its weight correction term (used to update w_{jk} later),

$$\Delta w_{jk} = \alpha \delta_k z_j,$$

calculates its bias correction term (used to update w_{0k} later),

$$\Delta w_{0k} = \alpha \delta_k,$$

and sends δ_k to units in the layer below.

Step 7. Each hidden unit ($Z_j, j = 1, \ldots, p$) sums its delta inputs (from units in the layer above),

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk},$$

multiplies by the derivative of its activation function to calculate its error information term,

$$\delta_j = \delta\_in_j \, f'(z\_in_j),$$

calculates its weight correction term (used to update v_{ij} later),

$$\Delta v_{ij} = \alpha \delta_j x_i,$$

and calculates its bias correction term (used to update v_{0j} later),

$$\Delta v_{0j} = \alpha \delta_j.$$

Update weights and biases:

Step 8. Each output unit ($Y_k, k = 1, \ldots, m$) updates its bias and weights ($j = 0, \ldots, p$):

$$w_{jk}(\text{new}) = w_{jk}(\text{old}) + \Delta w_{jk}.$$

Each hidden unit ($Z_j, j = 1, \ldots, p$) updates its bias and weights ($i = 0, \ldots, n$):

$$v_{ij}(\text{new}) = v_{ij}(\text{old}) + \Delta v_{ij}.$$

ep 9. Test stopping condition.