

# FUNDAMENTOS DA INTELIGÊNCIA ARTIFICIAL

*Aula 05 - Backpropagation  
(Multilayer Perceptrons - MLPs)*

Prof. Rafael G. Mantovani



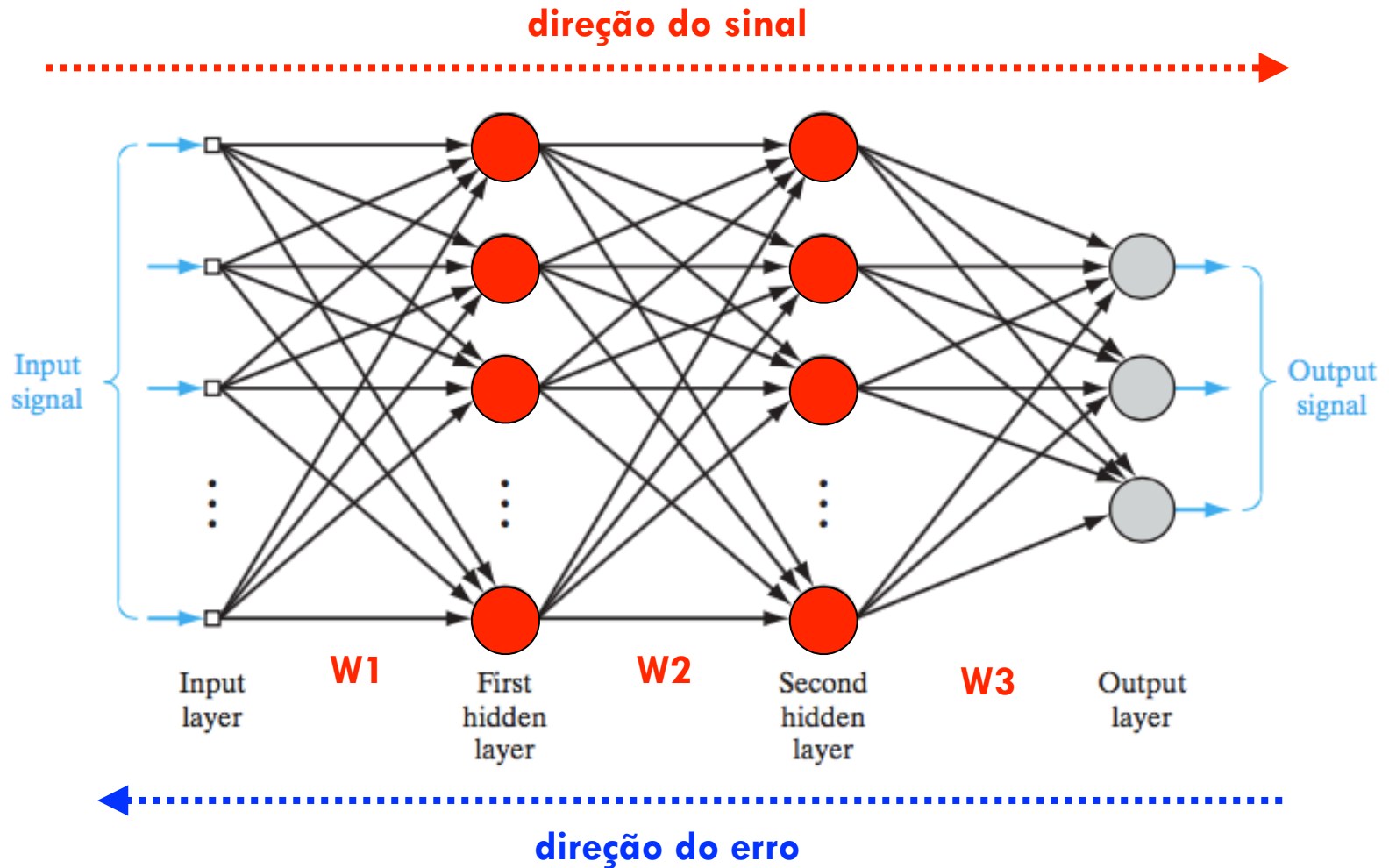
# Roteiro

- 1 Introdução
- 2 *Backpropagation*
- 3 Algoritmo Resumido
- 4 Exemplo / Exercício
- 5 Síntese / Próximas Aulas
- 6 Referências

# Roteiro

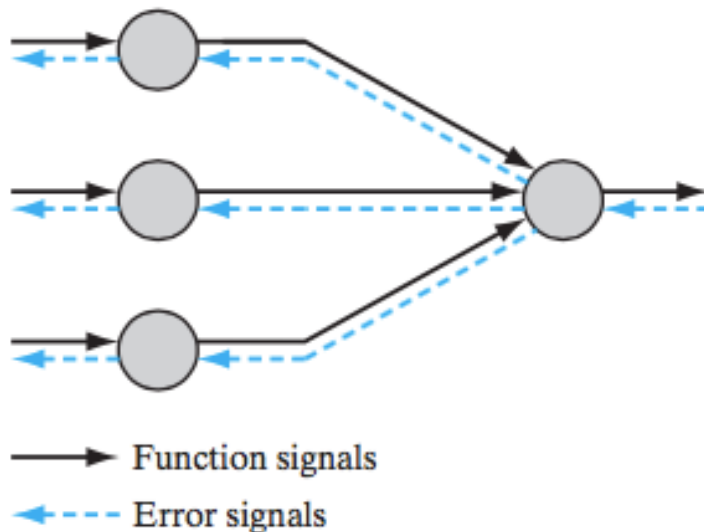
- 1** Introdução
- 2** *Backpropagation*
- 3** Algoritmo Resumido
- 4** Exemplo / Exercício
- 5** Síntese / Próximas Aulas
- 6** Referências

# Introdução



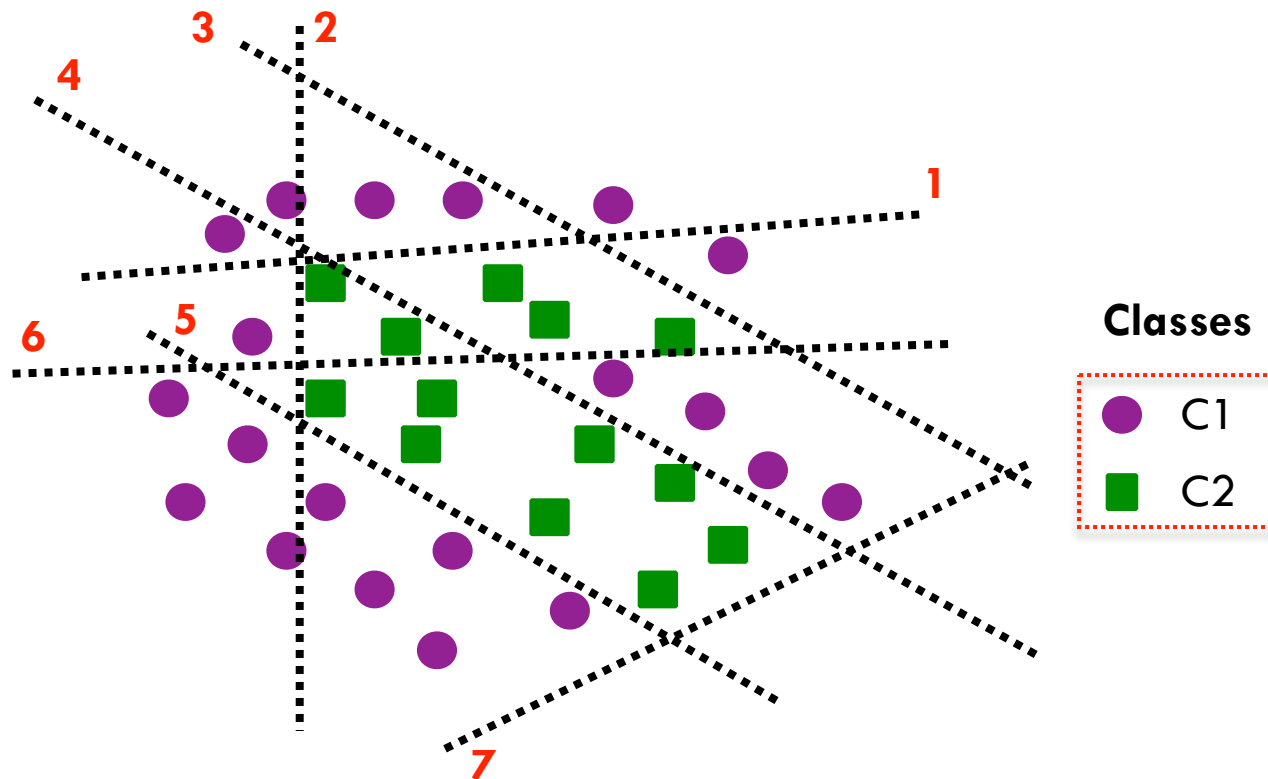
# Introdução

- Sinais de função e sinais de erro



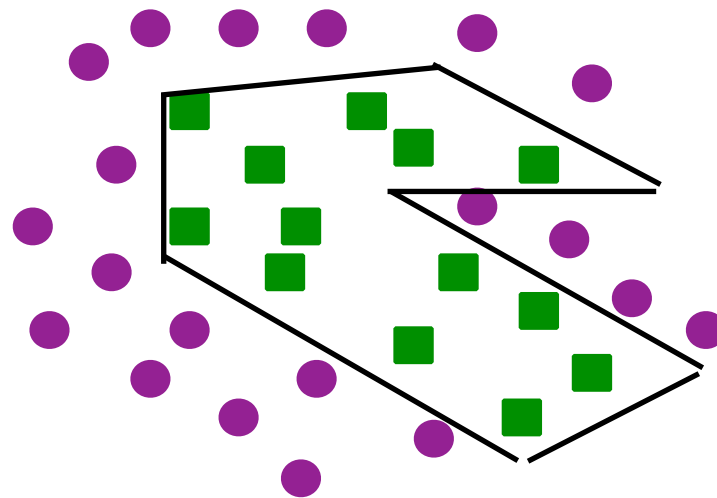
**FIGURE 4.2** Illustration of the directions of two basic signal flows in a multilayer perceptron: forward propagation of function signals and back propagation of error signals.

# Introdução

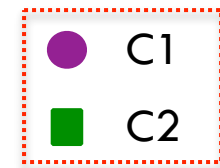


**7 hiperplanos = 7 neurônios**

# Introdução



**Classes**



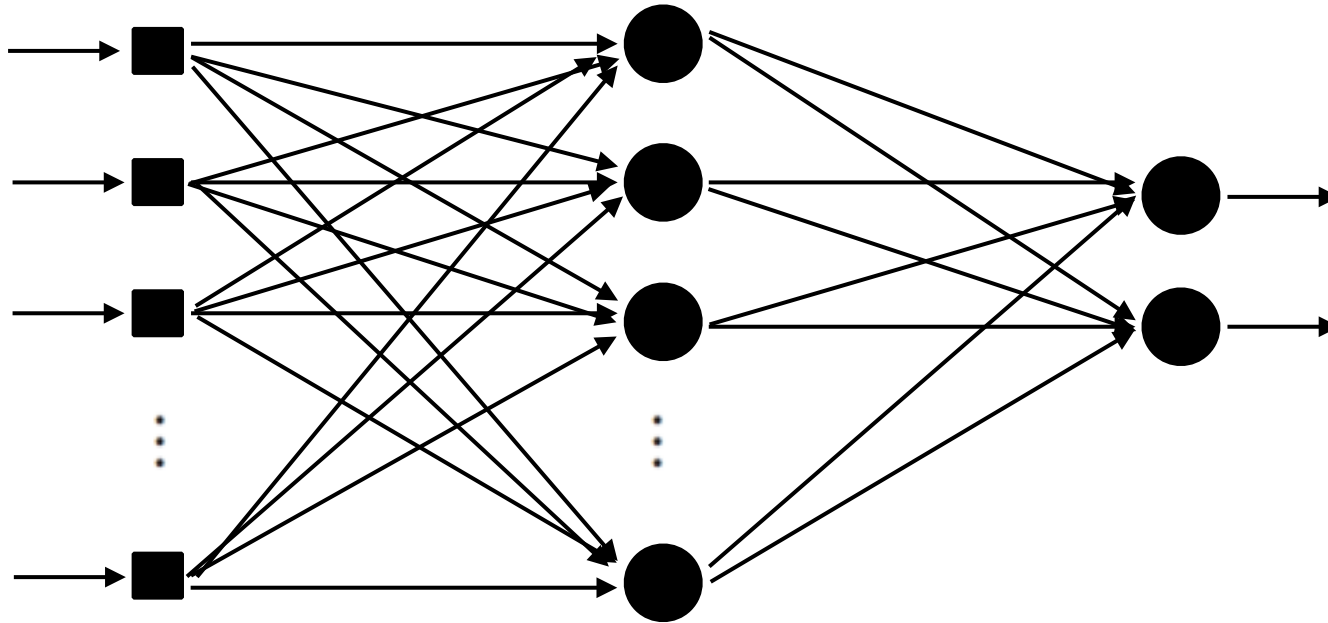
**7 hiperplanos = 1 região convexa**

# Roteiro

- 1 Introdução
- 2 ***Backpropagation***
- 3 Algoritmo Resumido
- 4 Exemplo / Exercício
- 5 Síntese / Próximas Aulas
- 6 Referências



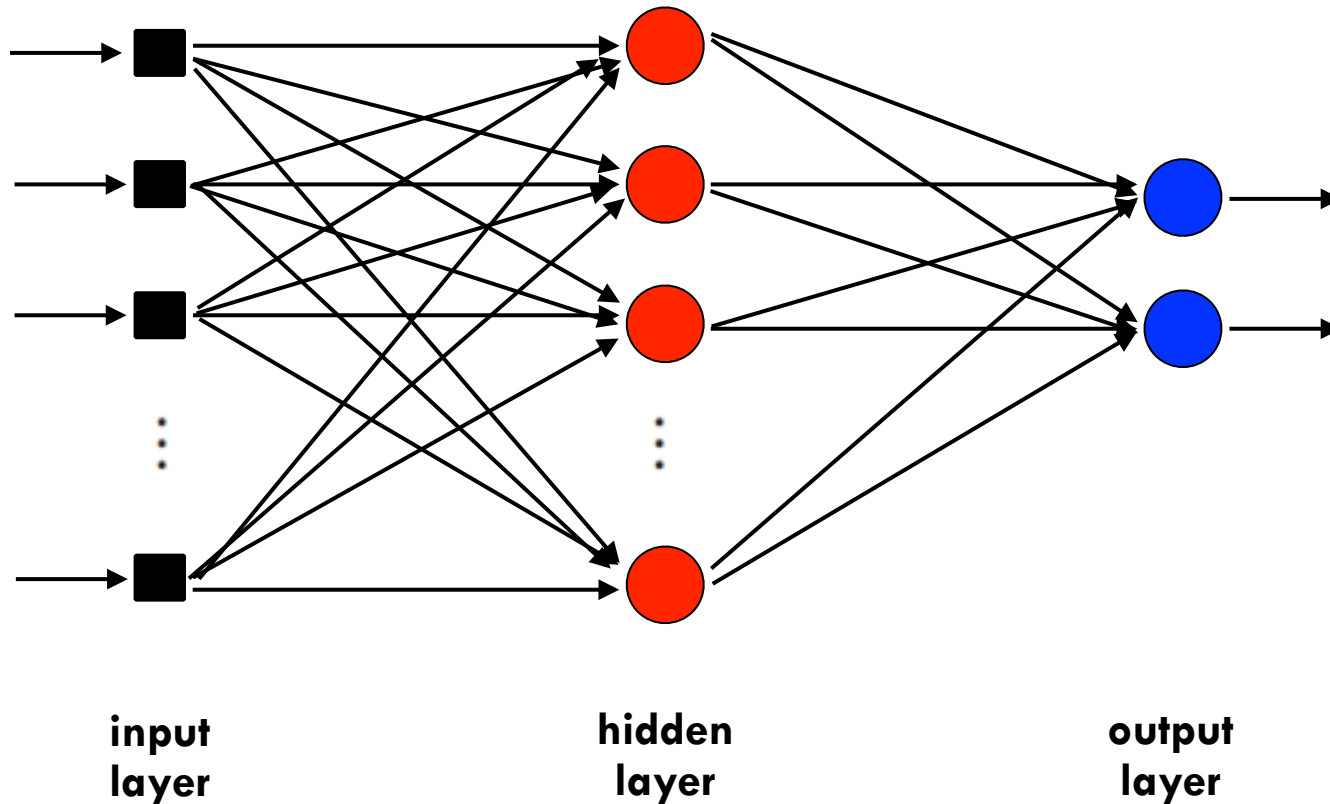
# Backpropagation



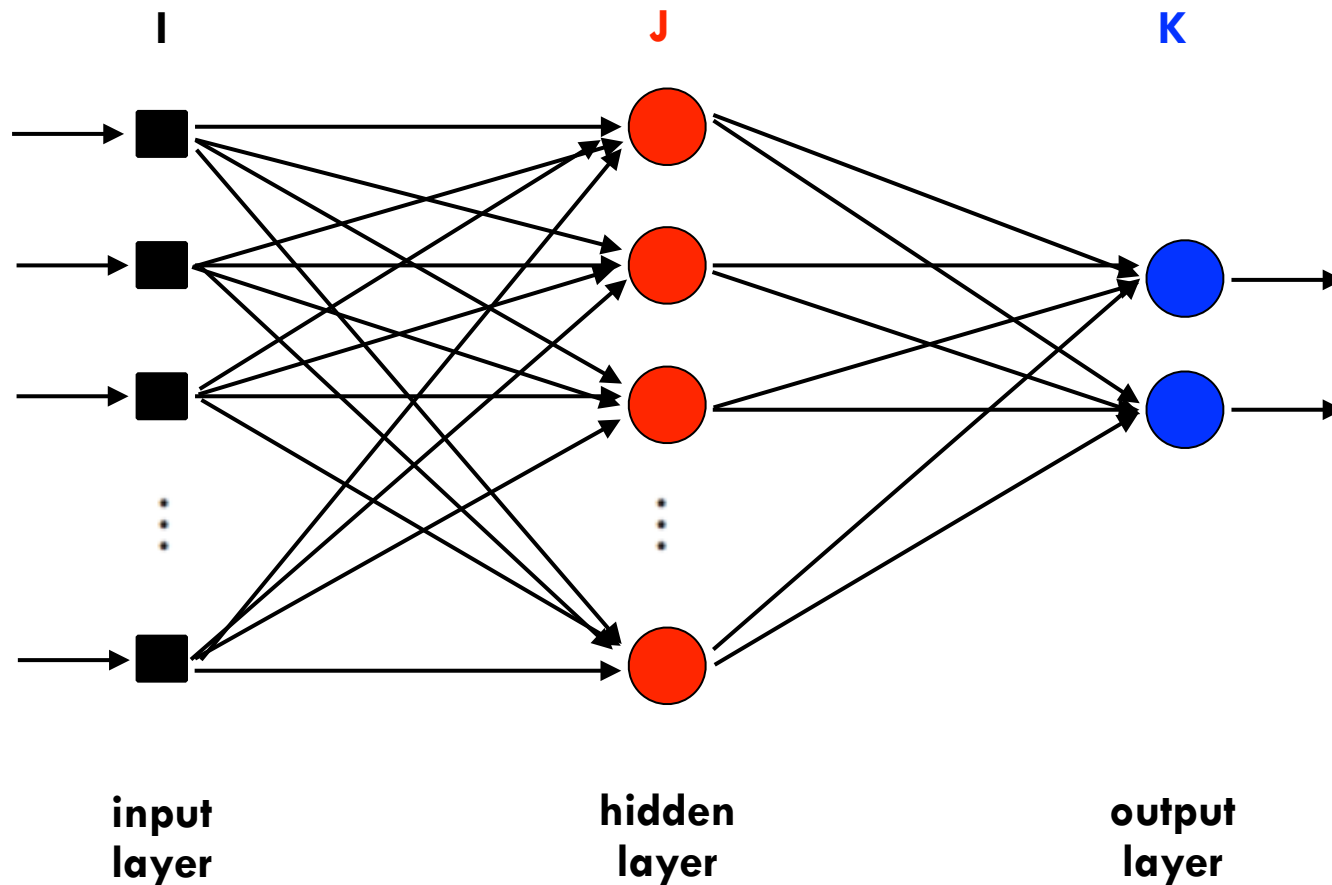
# Backpropagation

1. Aplicar um vetor de entrada para a rede ( $X$ ) e calcular os valores de saída
2. Comparar as saídas atuais com as saídas desejadas e obter uma medida de erro
3. Determinar em qual direção (+ ou -) e modificar os pesos para minimizar o erro
4. Determinar a quantidade para se modificar cada peso
5. Aplicar as correções aos pesos
6. Repetir de 1 a 5 com todos os exemplos de treinamento, até que uma margem de erro de treinamento seja atingida

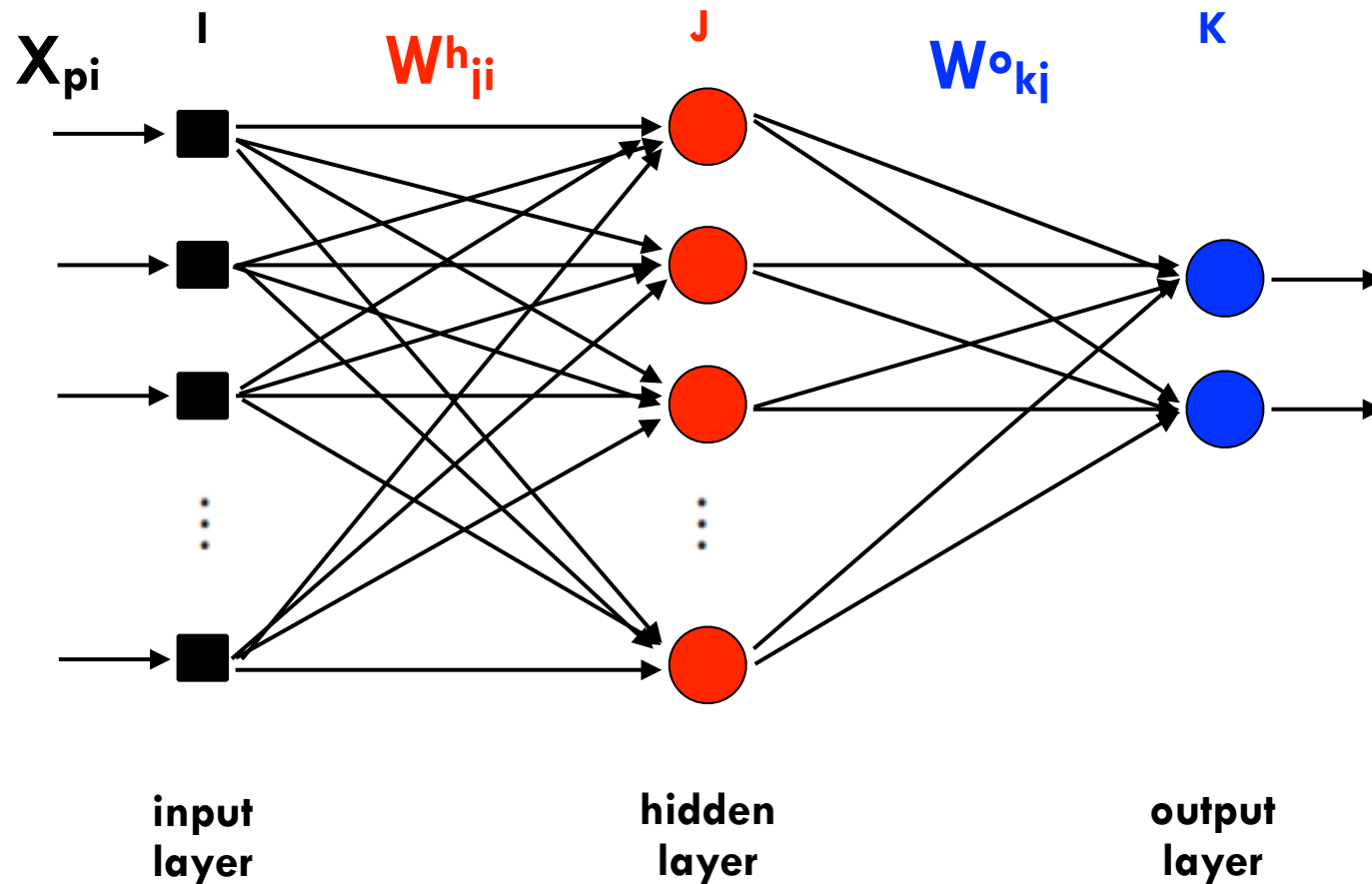
# Backpropagation



# Backpropagation



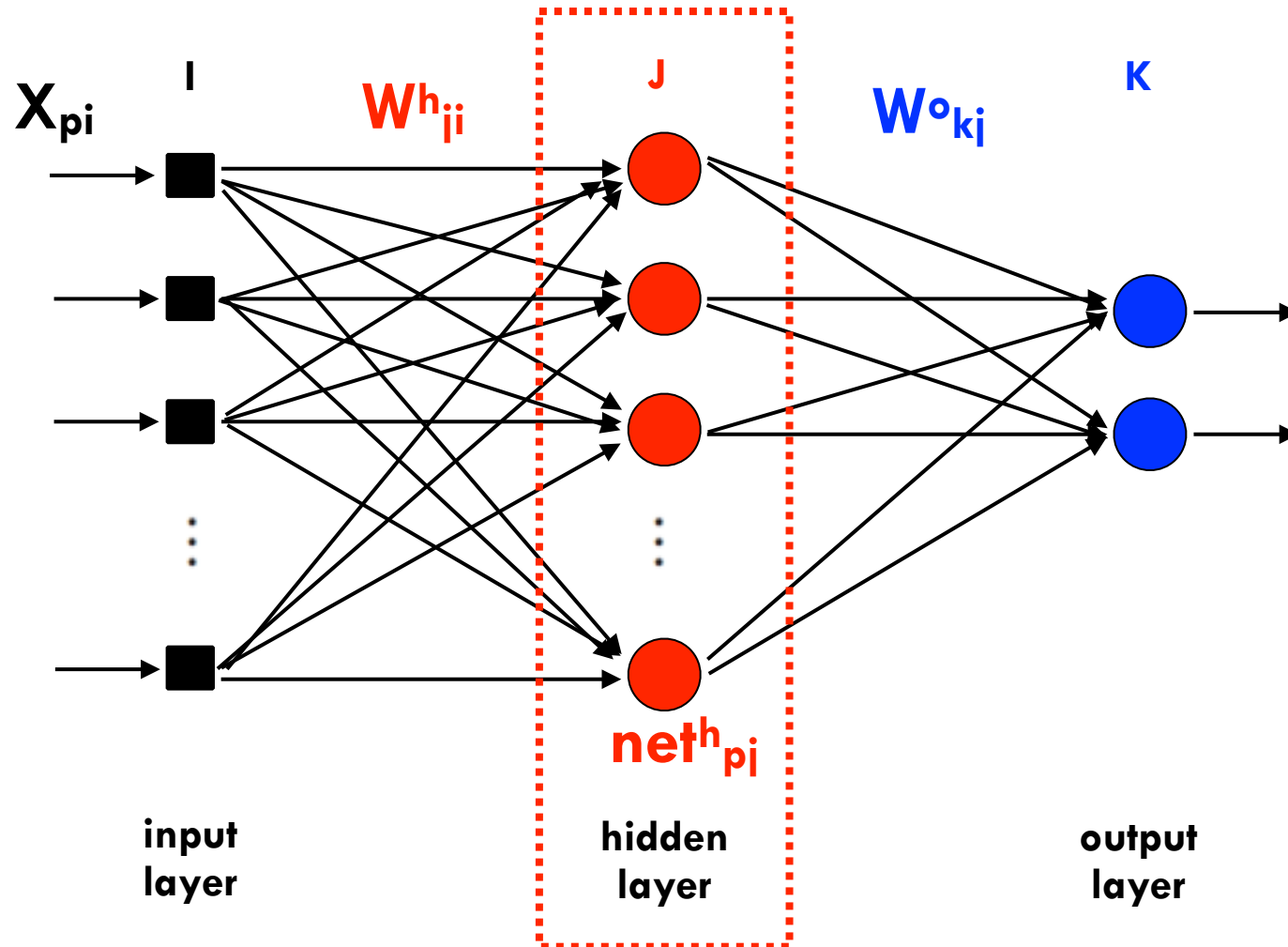
# Backpropagation



# Backpropagation

- 1. Aplicar um vetor de entrada para a rede (X) e calcular os valores de saída**
2. Comparar as saídas atuais com as saídas desejadas e obter uma medida de erro
3. Determinar em qual direção (+ ou -) e modificar os pesos para minimizar o erro
4. Determinar a quantidade para se modificar cada peso
5. Aplicar as correções aos pesos
6. Repetir de 1 a 5 com todos os exemplos de treinamento, até que uma margem de erro de treinamento seja atingida

# Backpropagation



# Backpropagation

- Sinal no neurônio de índice J na camada escondida:

$$\text{net}_{pj}^h = \sum_{i=1}^N w_{ji}^h x_{pi} + \theta_j^h$$

$N \rightarrow ?$

$w_{ji}^h \rightarrow ?$

$x_{pi} \rightarrow ?$

$\theta_j^h \rightarrow ?$



# Backpropagation

- Sinal no neurônio de índice J na camada escondida:

$$\text{net}_{pj}^h = \sum_{i=1}^N w_{ji}^h x_{pi} + \theta_j^h$$

$N$  → número de neurônios na camada de entrada

$w_{ji}^h$  → peso da conexão com o neurônio de entrada  $i$

$x_{pi}$  → padrão inserido na entrada da rede

$\theta_j^h$  → bias do neurônio

# Backpropagation

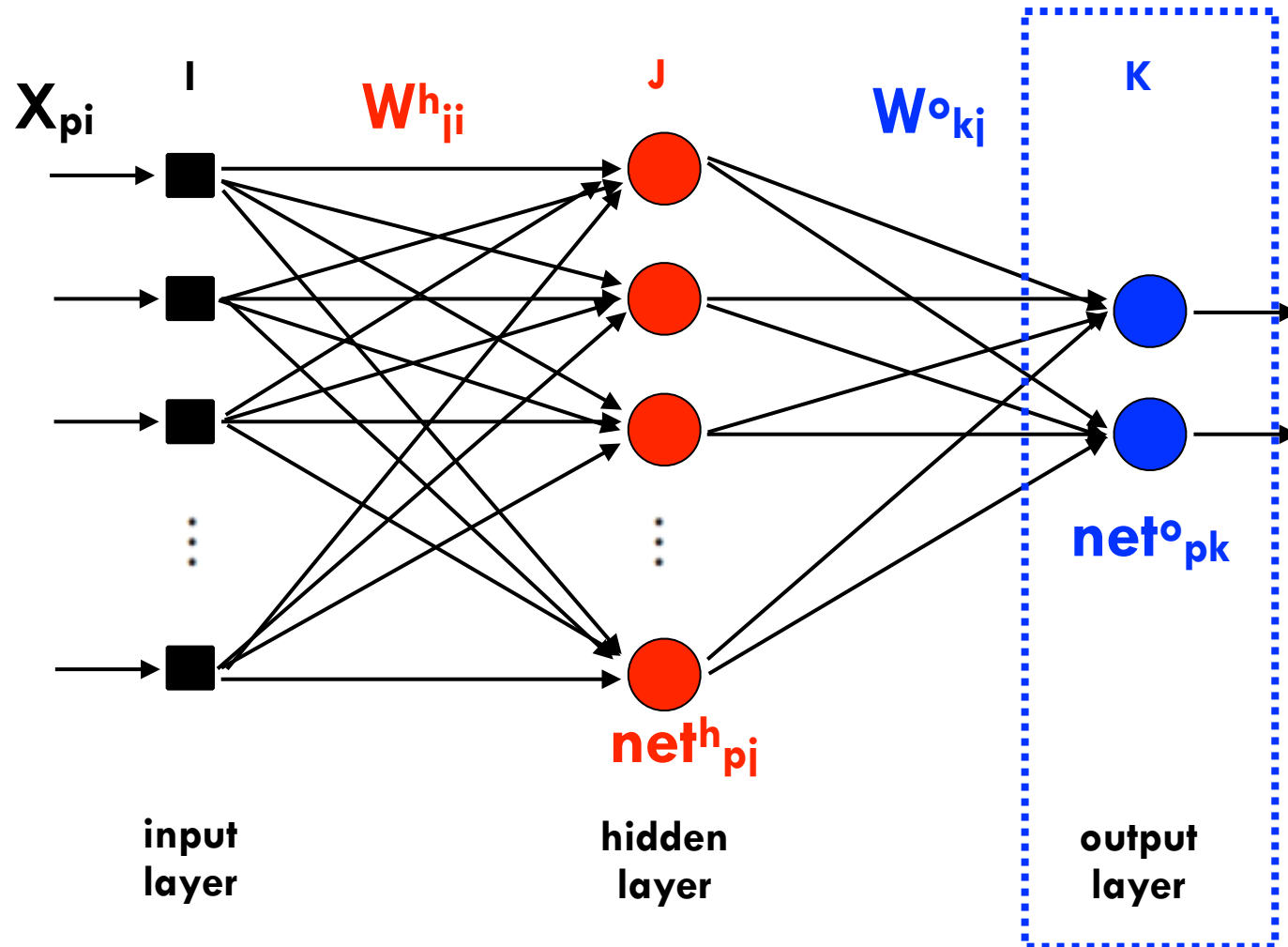
- Sinal no neurônio de índice J na camada escondida:

$$\text{net}_{pj}^h = \sum_{i=1}^N w_{ji}^h x_{pi} + \theta_j^h$$

- Ativação desse neurônio é igual a:

$$i_{pi} = f_i^h(\text{net}_{pi}^h)$$

# Backpropagation



# Backpropagation

- Sinal no neurônio de índice K na camada de saída:

$$\text{net}_{pk}^o = \sum_{j=1}^L w_{kj}^o i_{pj} + \theta_k^o$$

$L \rightarrow$

$w_{pj}^o \rightarrow$

$i_{pj} \rightarrow$

$\theta_k^o \rightarrow$

# Backpropagation

- Sinal no neurônio de índice K na camada de saída:

$$\text{net}_{pk}^o = \sum_{j=1}^L w_{kj}^o i_{pj} + \theta_k^o$$

$L$  → número de neurônios na camada de saída

$w_{pj}^o$  → peso da conexão com o neurônio j da camada escondida

$i_{pj}$  → valor de ativação do neurônio j da camada escondida

$\theta_k^o$  → bias do neurônio

# Backpropagation

- Sinal no neurônio de índice K na camada de saída:

$$\text{net}_{pk}^o = \sum_{j=1}^L w_{kj}^o i_{pj} + \theta_k^o$$

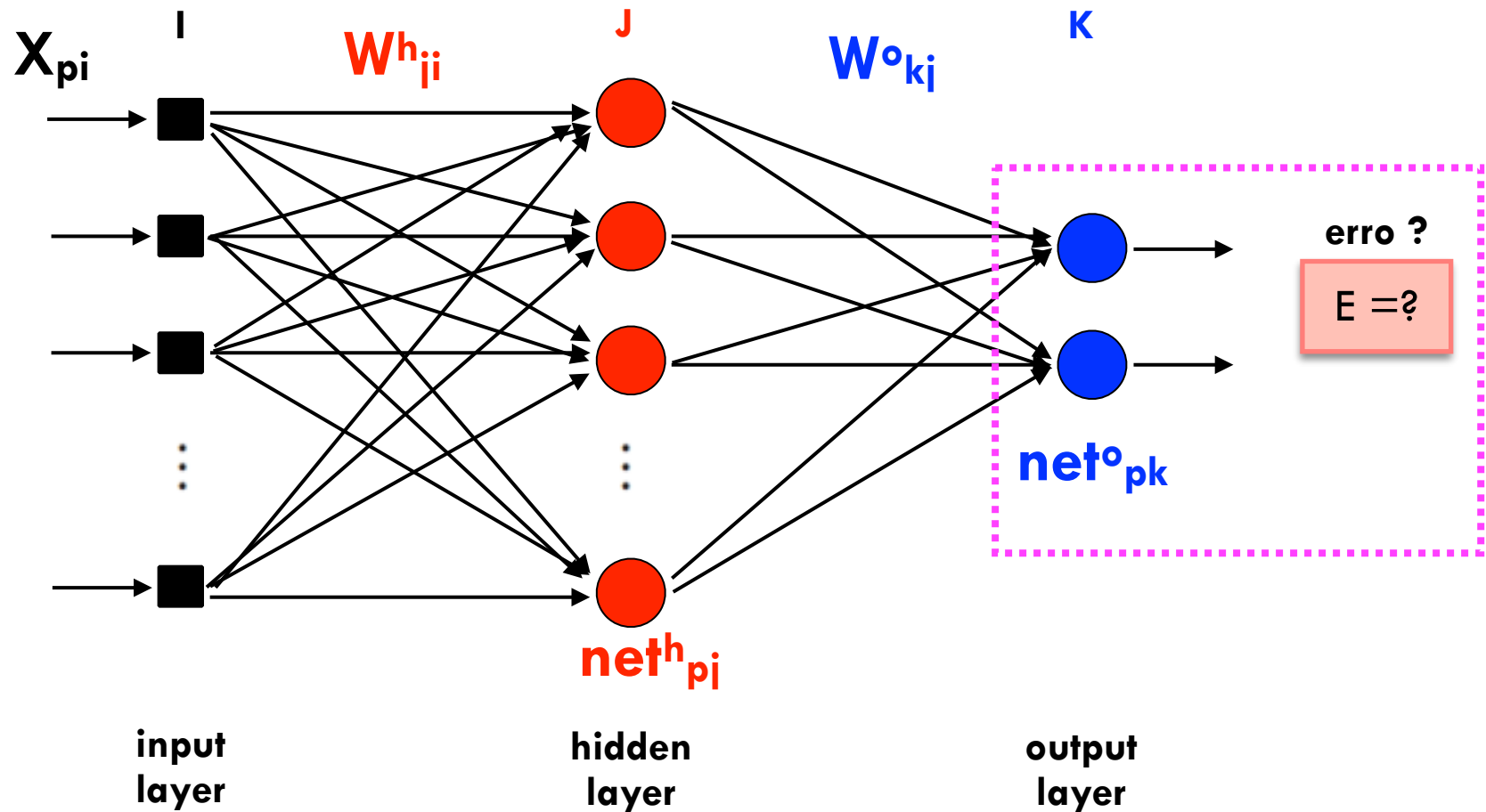
- Saída desse neurônio é igual a:

$$o_{pk} = f_k^o(\text{net}_{pk}^o)$$

# Backpropagation

1. Aplicar um vetor de entrada para a rede ( $X$ ) e calcular os valores de saída
- 2. Comparar as saídas atuais com as saídas desejadas e obter uma medida de erro**
3. Determinar em qual direção (+ ou -) e modificar os pesos para minimizar o erro
4. Determinar a quantidade para se modificar cada peso
5. Aplicar as correções aos pesos
6. Repetir de 1 a 5 com todos os exemplos de treinamento, até que uma margem de erro de treinamento seja atingida

# Backpropagation





# Backpropagation

- O erro de um único exemplo em uma única unidade de saída é definido por:

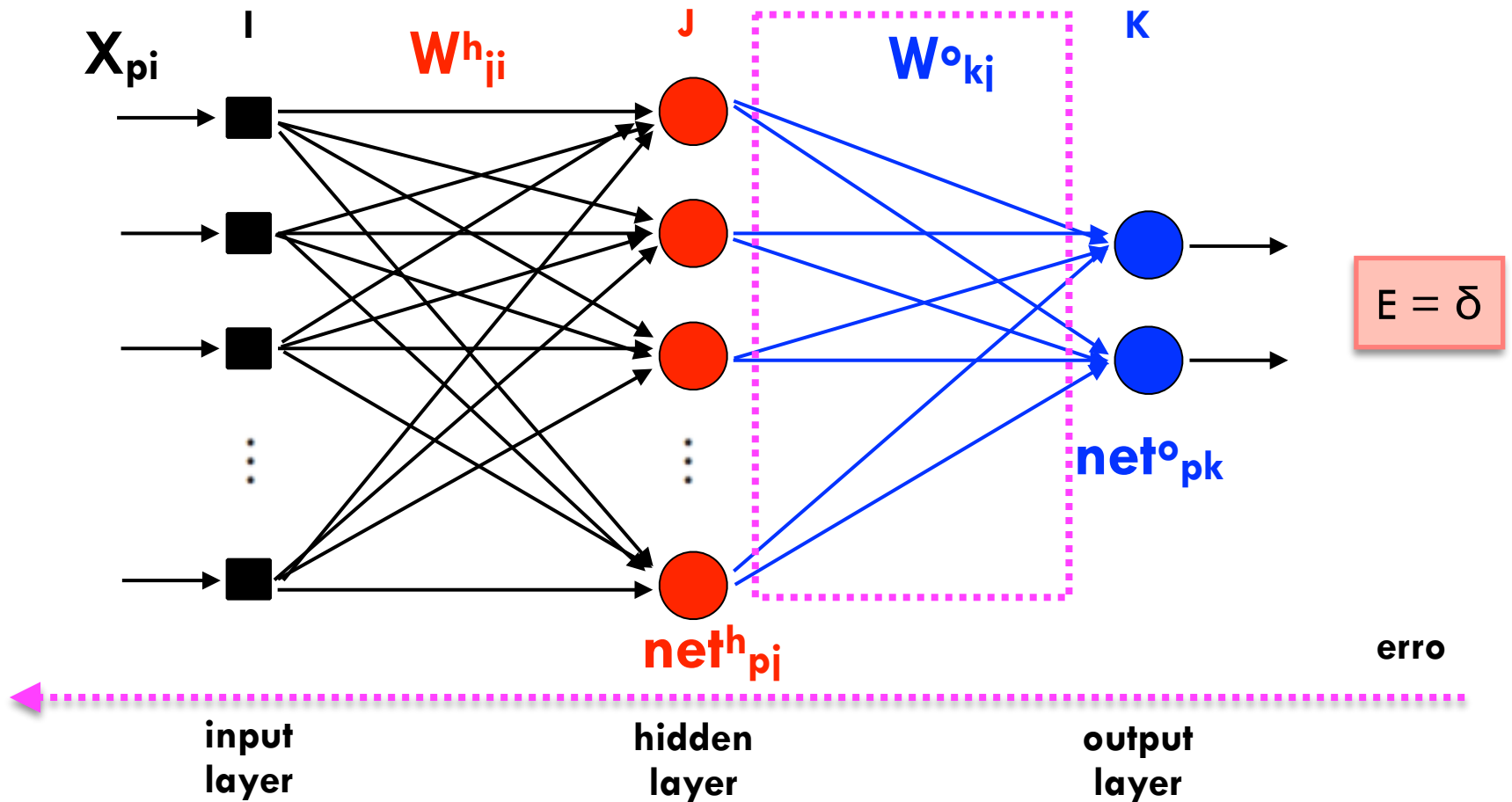
$$\delta_{pk} = (y_{pk} - o_{pk})$$

- onde:
  - $p$  refere-se ao exemplo “ $p$ ” do conjunto de treinamento
  - $y_{pk}$ : é a saída desejada
  - $o_{pk}$ : é a saída obtida

# Backpropagation

1. Aplicar um vetor de entrada para a rede ( $X$ ) e calcular os valores de saída
2. Comparar as saídas atuais com as saídas desejadas e obter uma medida de erro
- 3. Determinar em qual direção (+ ou -) e modificar os pesos para minimizar o erro**
- 4. Determinar a quantidade para se modificar cada peso**
- 5. Aplicar as correções aos pesos**
6. Repetir de 1 a 5 com todos os exemplos de treinamento, até que uma margem de erro de treinamento seja atingida

# Backpropagation



# Backpropagation

- Os pesos da camada de saída ( $W_{kj}^o$ ) são atualizados de acordo com:

- termos de erro da camada de saída:

$$\delta_{pk}^o = (y_{pk} - o_{pk}) * f_{o_k}'(net_{pk}^o)$$

- ajuste de pesos da camada de saída

$$W_{kj}^o(t+1) = W_{kj}^o(t) + \eta * \delta_{pk}^o * i_{pj}$$

# Backpropagation

- Deixando em uma única fórmula:

$$W_{kj}^o(t+1) = W_{kj}^o(t) + \eta (y_{pk} - o_{pk}) * f_k'(net_{pk}^o) * i_{pj}$$

$y_{pk}$  : é a saída esperada para o padrão p

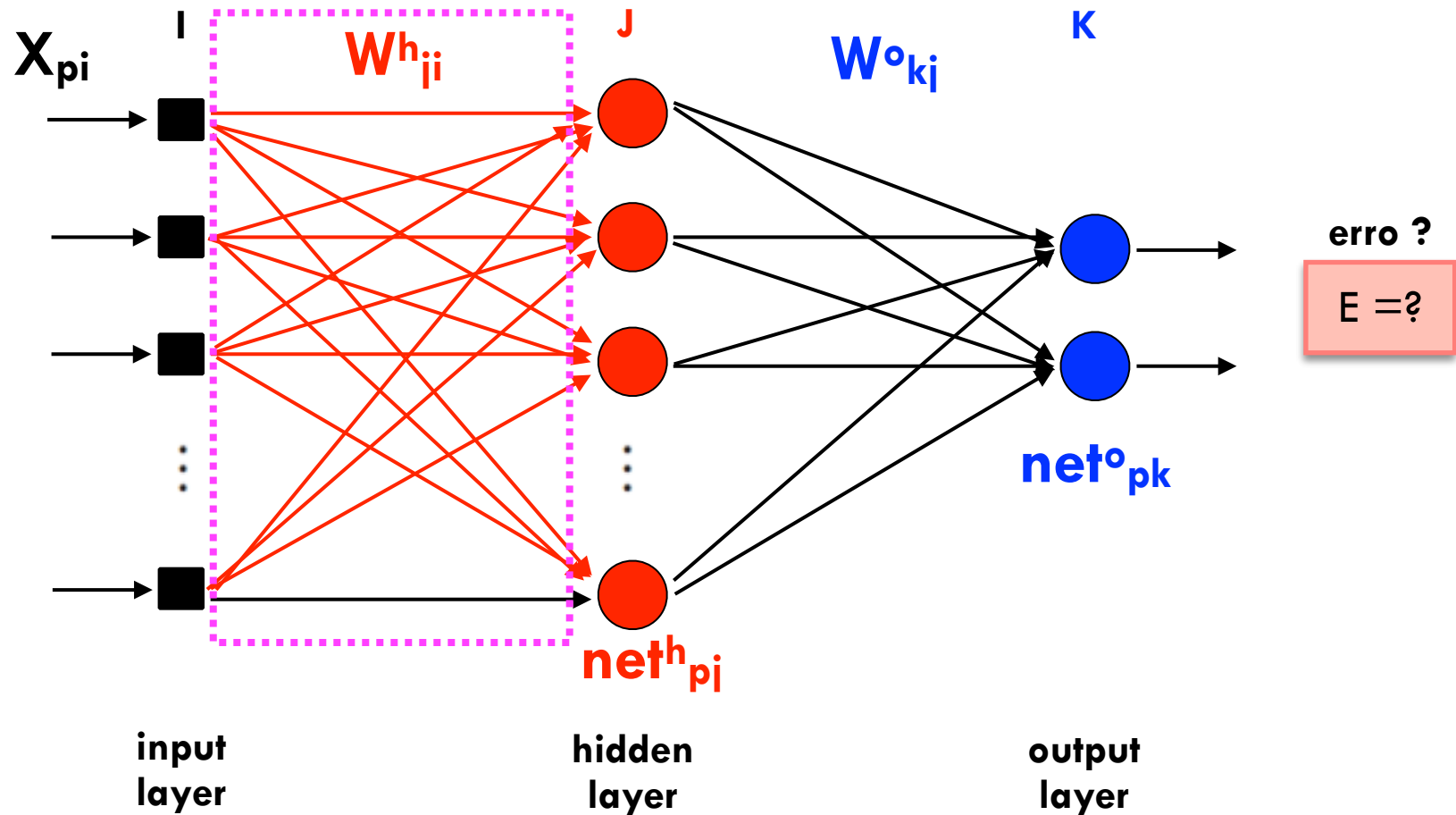
$o_{pk}$  : é a saída obtida para o padrão p

$i_{pj}$  : é a saída de ativação da unidade j da cama oculta

$f_k'$  : é a derivada da função de ativação

$\eta$  : é uma taxa de aprendizado, positiva e menor que 1

# Backpropagation



# Backpropagation

- Os pesos da camada de saída ( $W^{h_{ji}}$ ) são atualizados de acordo com:

- termos de erro da camada oculta:

$$\delta_{h_{pi}} = f^{h'_{i}}(\text{net}^{h_{pi}}) \sum_k (\delta_{o_{pk}} * W^{o_{ki}})$$

- ajuste de pesos da camada oculta

$$W^{h_{ji}}(t+1) = W^{h_{ji}}(t) + \eta * \delta_{h_{pi}} * x_{pi}$$

# Backpropagation

- Por fim, calcula-se o erro total da rede para todos os exemplos de entrada ( $M$ ):

$$E_p = 0.5 * \sum^M_k (y_{pk} - o_{pk})^2$$



# Roteiro

- 1 Introdução
- 2 *Backpropagation*
- 3 **Algoritmo Resumido**
- 4 Exemplo / Exercício
- 5 Síntese / Próximas Aulas
- 6 Referências

# Algoritmo resumido

1. Aplicar o valor de entrada  $X_p$  na rede
2. Calcular os valores de entrada na camada oculta ( $\text{net}_{pi}^h$ )

$$\text{net}_{pj}^h = \sum_{i=1}^N w_{ji}^h x_{pi} + \theta_j^h$$

3. Calcular as saídas da camada oculta ( $i_{pi}$ )

$$i_{pi} = f_i^h(\text{net}_{pi}^h)$$

# Algoritmo resumido

4. Avance para a camada de saída. Calcular os valores de entrada para cada unidade de saída ( $\text{net}^o_{pk}$ )

$$\text{net}^o_{pk} = \sum_{j=1}^L w^o_{kj} i_{pj} + \theta^o_k$$

5. Calcular as saídas da camada de saída ( $o_{pk}$ )

$$o_{pk} = f^o_k(\text{net}^o_{pk})$$

# Algoritmo resumido

6. Calcular os termos de erro para as unidades de saída:

$$\delta^o_{pk} = (y_{pk} - o_{pk}) * f'^o_k(\text{net}^o_{pk})$$

7. Calcular os termos de erro para as unidades ocultas:

$$\delta^h_{pi} = f^{h'}_i(\text{net}^h_{pi}) \sum_k (\delta^o_{pk} * W^o_{kj})$$

**Obs:** o erro das unidades ocultas é calculado **ANTES** do ajuste de pesos da camada de saída

# Algoritmo resumido

8. Atualizar os pesos da camada de saída

$$W^o_{kj} (t+1) = W^o_{kj} (t) + \eta * \delta^o_{pk} * i_{pj}$$

9. Atualizar os pesos da camada oculta

$$W^h_{ji} (t+1) = W^h_{ji} (t) + \eta * \delta^h_{pi} * x_{pi}$$

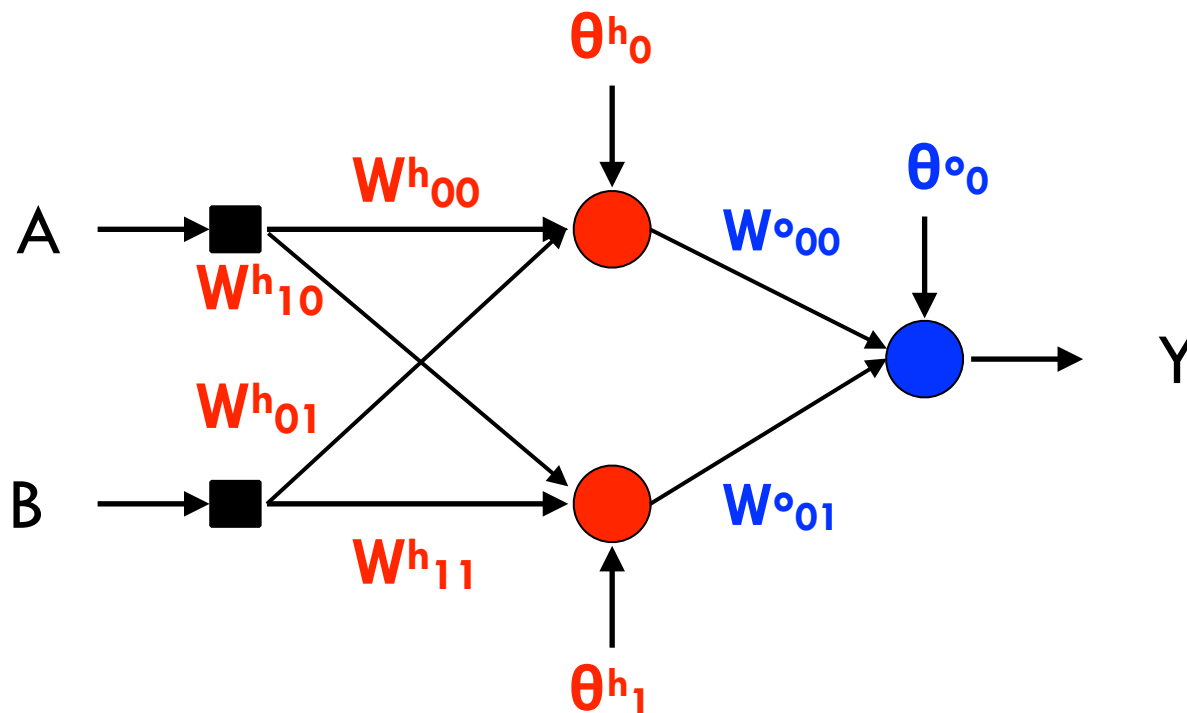
10. Calcular o erro total da época

- a. Indica o quão bem a rede está aprendendo.
- b. quando for menor que um limiar, parar

# Roteiro

- 1 Introdução
- 2 *Backpropagation*
- 3 Algoritmo Resumido
- 4 Exemplo / Exercício
- 5 Síntese / Próximas Aulas
- 6 Referências

# Exemplo



XOR dataset

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

# Exemplo

epoch	$\theta^h0$	$\theta^h1$	$\theta^o0$	$W^h00$	$W^h10$	$W^h01$	$W^h11$	$W^o00$	$W^o01$
0	0.05	0.06	0.07	0.2	0.15	0.35	0.18	0.10	0.12
1									
2									

- $\eta = 0.2$
- $f(\text{net}) = 1 / (1 + \exp^{-\text{net}})$
- $f'(\text{net}) = \text{net} (1 - \text{net})$



# Roteiro

- 1 Introdução
- 2 *Backpropagation*
- 3 Algoritmo Resumido
- 4 Exemplo / Exercício
- 5 **Síntese / Próximas Aulas**
- 6 Referências

# Próxima Aula

## □ MLPs

- combinação de hiperplanos / regiões convexas
- Backpropagation
  - forward → propaga sinal
  - backward → propaga o erro
- exemplo (XOR)

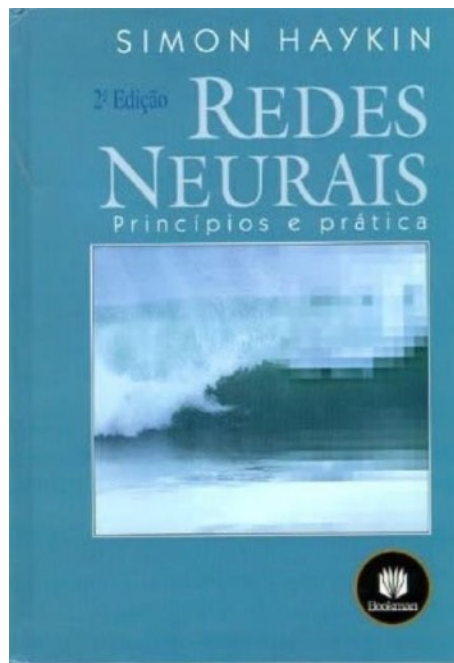
# Próxima Aula

- Implementação:
  - MLP
  - *Backpropagation*
- Redes RBF

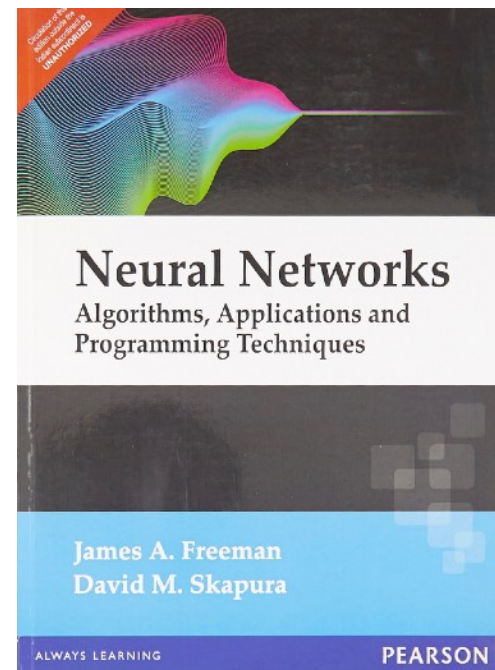
# Roteiro

- 1 Introdução
- 2 *Backpropagation*
- 3 Algoritmo Resumido
- 4 Exemplo / Exercício
- 5 Síntese / Próximas Aulas
- 6 Referências

# Literatura Sugerida



(Haykin, 1999)



(Freeman & Skapura, 1991)

# Perguntas?

Prof. Rafael G. Mantovani

[rgmantovani@uel.br](mailto:rgmantovani@uel.br)

# Exercício

epoch	$\theta^h0$	$\theta^h1$	$\theta^o0$	$W^h00$	$W^h10$	$W^h01$	$W^h11$	$W^o00$	$W^o01$
0	0.05	0.06	0.07	0.2	0.15	0.35	0.18	0.10	0.12
1									
2									

- $X = \text{XOR dataset}$
- $\eta = 0.2$
- $f(\text{net}) = \text{net}^3 + 0.5$
- $f'(\text{net}) = 3 * \text{net}^2$