Uma pequena introdução às

Redes Neurais Artificiais

e Aplicações

Curso compacto – Informativo

Luiz P. **Calôba**

caloba@ufrj.br , www.lps.ufrj.br/~caloba

COPPE / UFRJ

Maio de 2011.

www.lps.ufrj.br/~caloba/minicurso.rn.2011.pdf

Redes Neurais Artificiais

O que é isto ?

Para que serve ?

De onde veio ?

Inteligência Artificial

Cognitiva, Simbólica

IA “tradicional”

Evolucionista

AG, Vida Artificial

Conexionista

Redes Neurais Artificiais

Redes Neurais Artificiais

Redes Feedforward

Aproximadortes e classificadores,

Redes “backpropagation”

Redes Não Supervisionadas

Classificadores

Redes Realimentadas

Otimizadores

Redes Neurais Naturais

Breve Histórico

Aristóteles - 400 A.C.

"De memória et reminiscentia"

McCulloch e Pitts - 1943

primeiro modelo matemático de neurônio

Frank Rosenblatt - 1958

perceptrons

Minsky e Pappert - 1969

"Perceptrons"

Rumelhart, Williams e Hinton - 1986

redescoberta da Backpropagation.

Como implementar as Redes Neurais Artificiais ?

Emulando os sistemas biológicos !

Neurônio biológico

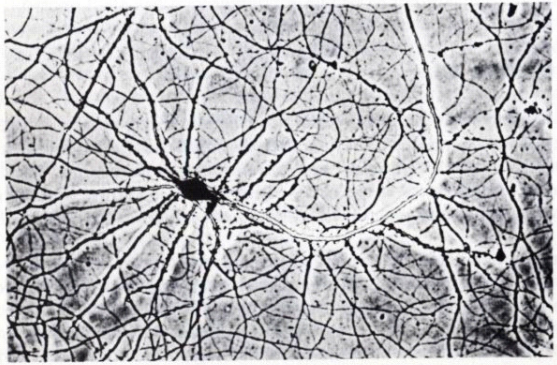
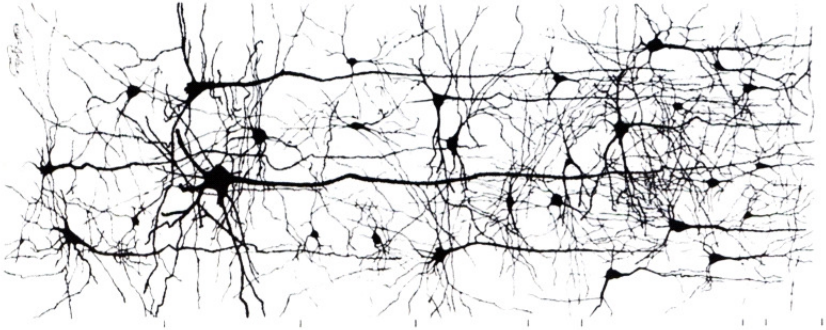
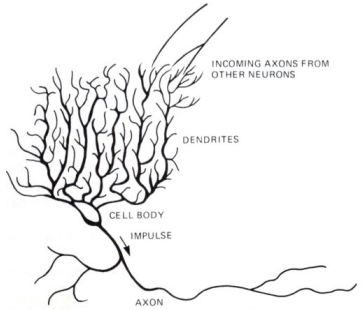
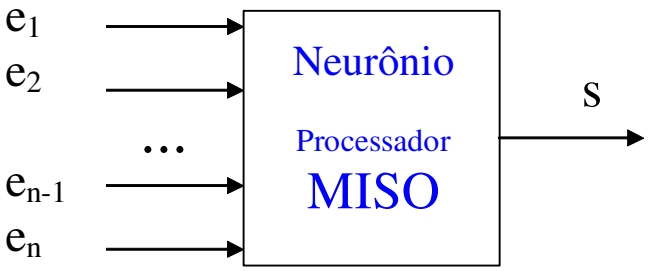


Figure 1-2a. A biological neuron magnified 400× with the dendritic tree in the foreground (courtesy of Gary Banker and Aaron Waxman, Univ. of Virginia).



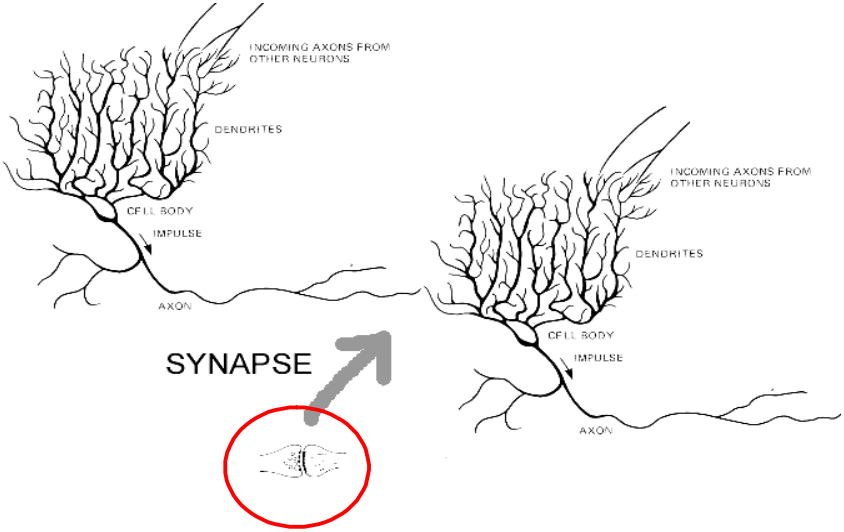
Neurônio biológico





$$\text{estado} \begin{cases} \text{ativo, excitado} & \text{saída} > s_0 \\ \text{inativo, inativo} & \text{saída} < s_0 \end{cases}$$

Comunicação entre neurônios:



Memória:

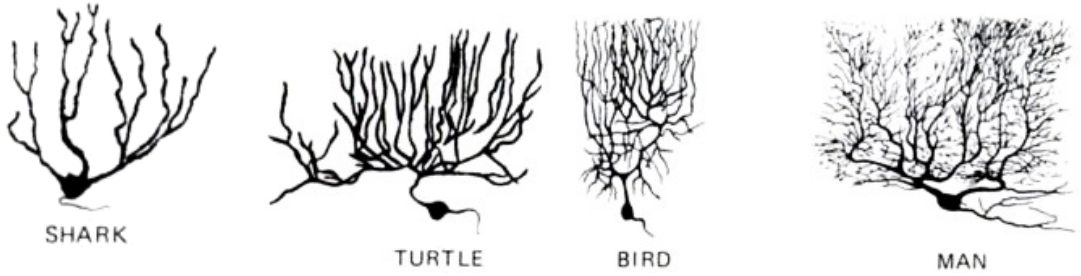
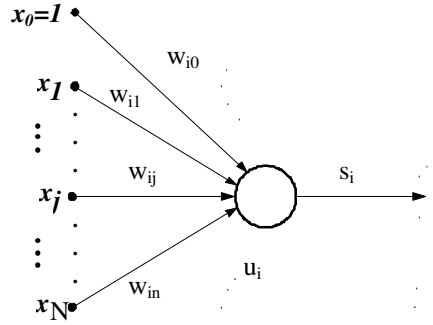
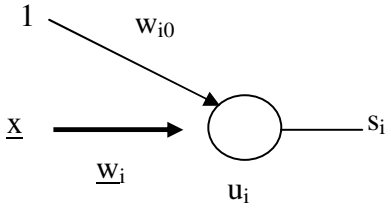


Figure 7-13. Growth of the dendritic trees and axon branches of cortical pyramidal cells in the human, from fetus to adult. (Courtesy of Sidman and Rakic 1982, and Poliakov in Sarkisov and Preobrazenskaya 1959.)

Neurônio - elemento de processamento

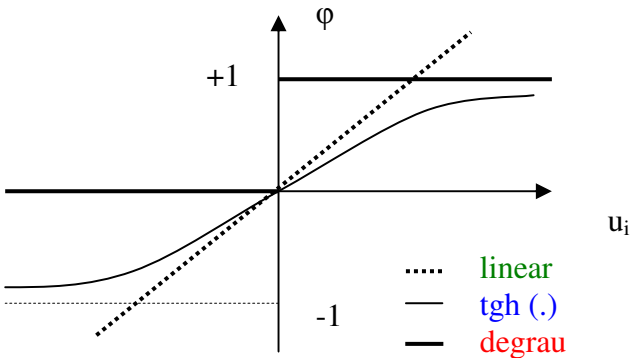




$$u_i = \sum_{j=1}^{N} w_{ij}\, x_j + w_{i0} = \underline{w}_i^t\, \underline{x} + w_{i0}$$

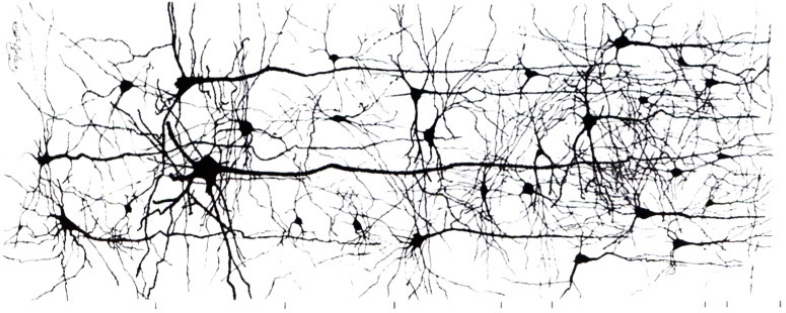
$$s_i = s(u_i)$$

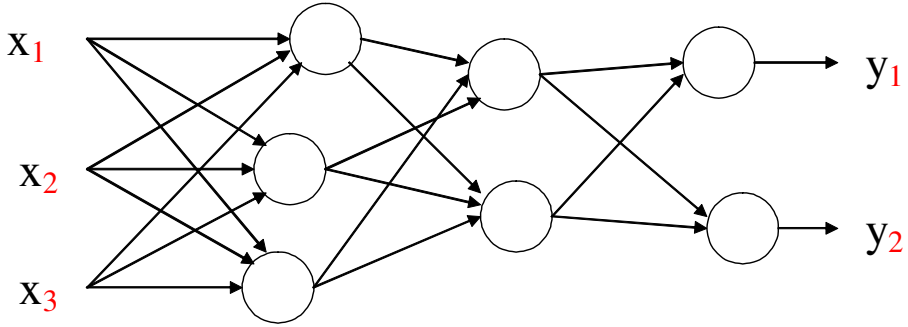
Função de Ativação s(u)



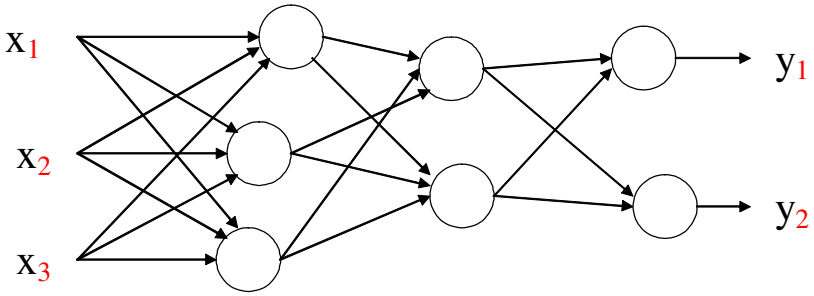
Neurônio	Função de ativação $s_i(u_i)$	Ganho linearizado $g_i(s_i) = ds_i / du_i$
linear	u_i	1
Não linear, tipo tgh	$\mathrm{tgh}(u_i) = \frac{1-e^{-2u_i}}{1+e^{-2u_i}}$	$1-s_i^2$
Não linear, Tipo binário	$\mathrm{deg}(u_i) = \begin{cases} 0 & \text{se } u_i < 0 \\ 1 & \text{se } u_i \geq 0 \end{cases}$	-

Arquitetura da Rede





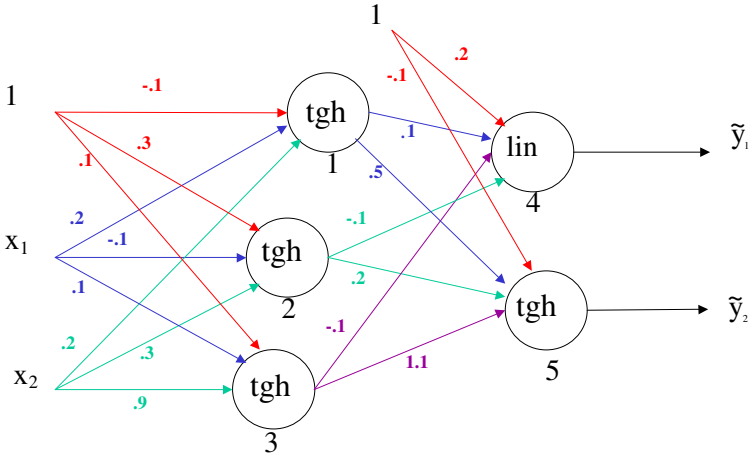
Arquitetura da Rede



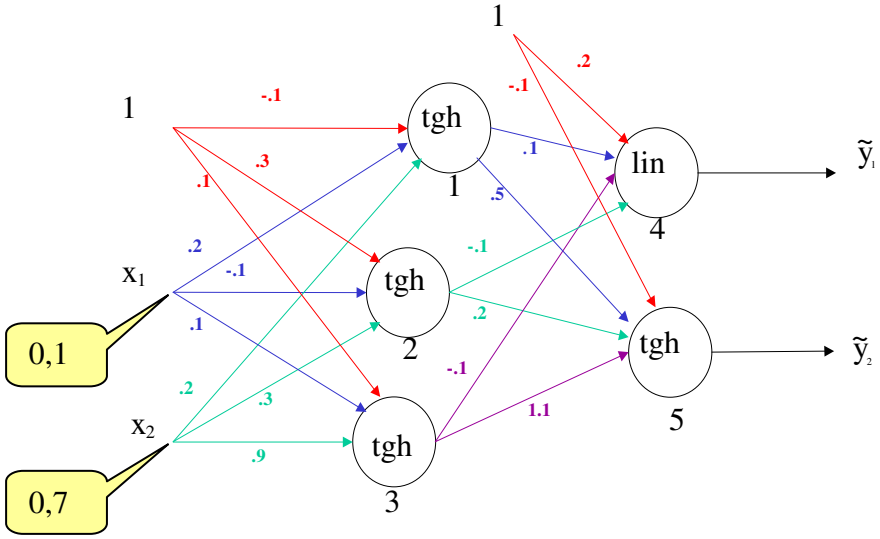
Rede feedforward (sem realimentação)

- Estática
- Estruturalmente estável

Exemplo de operação da rede:



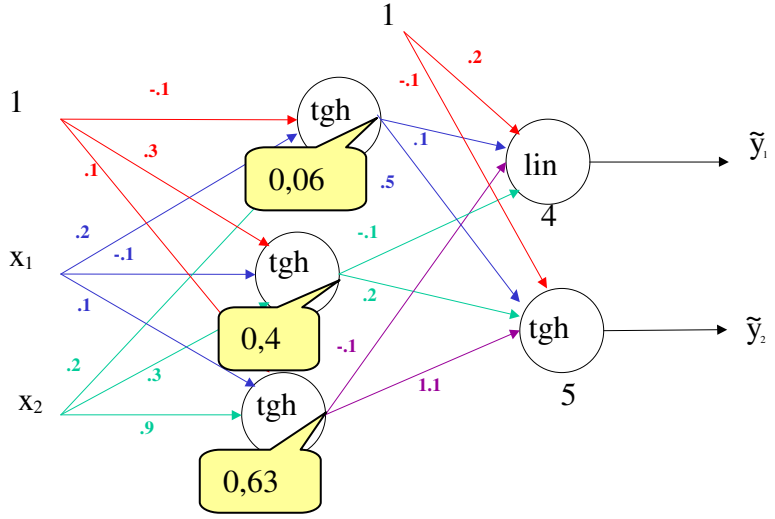
$$\underline{\mathbf{x}} = \begin{bmatrix} 0,1 \\ 0,7 \end{bmatrix} \qquad \tilde{\mathbf{y}} = ?$$



$$u_1 = -0{,}1 + (0{,}2)(0{,}1) + (0{,}2)(0{,}7) = 0{,}06$$

$$v_1 = \text{tgh}(\,0{,}06\,) = 0{,}06$$

$$v_2 = 0{,}46 \qquad v_3 = 0{,}63$$

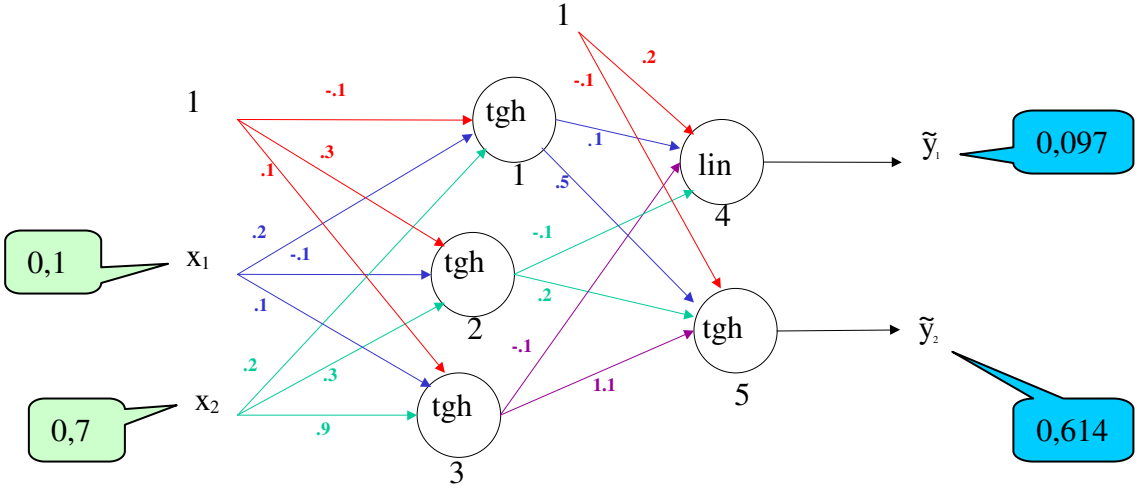


$$u_4 = 0{,}2 + (0{,}1)(0{,}06) + (-0{,}1)(0{,}46) + (-0{,}1)(0{,}63) = 0{,}097$$

$$v_4 = 0{,}097 \quad \text{(linear!)}$$

$$u_5 = -0{,}1 + (0{,}5)(0{,}06) + (0{,}2)(0{,}46) + (1{,}1)(0{,}63) = 0{,}715$$

$$v_5 = \text{tgh}(0{,}715) = 0{,}614$$



$$\underline{\mathbf{x}} = \begin{bmatrix} 0{,}1 \\ 0{,}7 \end{bmatrix} \qquad \tilde{\mathbf{y}} = \begin{bmatrix} 0{,}097 \\ 0{,}614 \end{bmatrix}$$

Uma Rede Neural Feedforward é apenas um mapeador não linear !

Parte I –

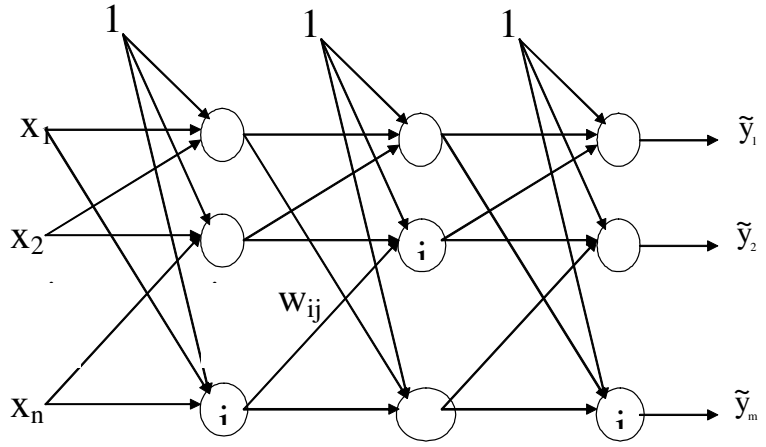
Redes Neurais Feedforward

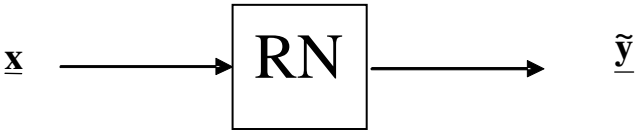
Aproximador Universal

Aprendizado Backpropagation

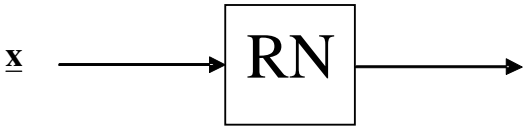
Redes Neurais *FeedForward*

Redes "*Backpropagation*"





$$\underline{\tilde{y}} = \varphi\left(\underline{x}\right)$$



$$\tilde{\underline{y}} = \varphi\left(\underline{x}\right)$$

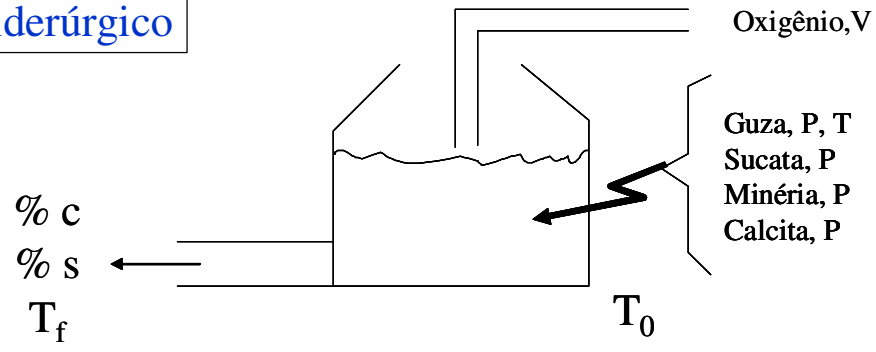
Para que serve ?

Aplicações:

Simuladores;
Controladores;
Classificadores;
Reconhecimento de padrões;
Filtragem não linear;
etc...

Simulação de Sistemas

Conversor Siderúrgico

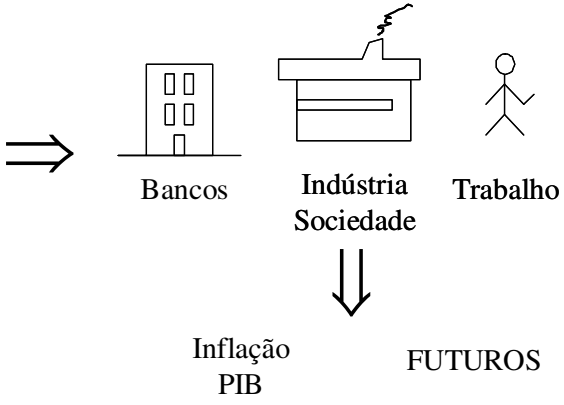


Simulação de Sistemas

Sistema Econômico

- Produção de bens de consumo
- intermediários
- capital
- Quantidade de moeda
- Inflação

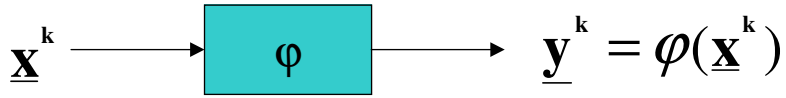
PASSADOS



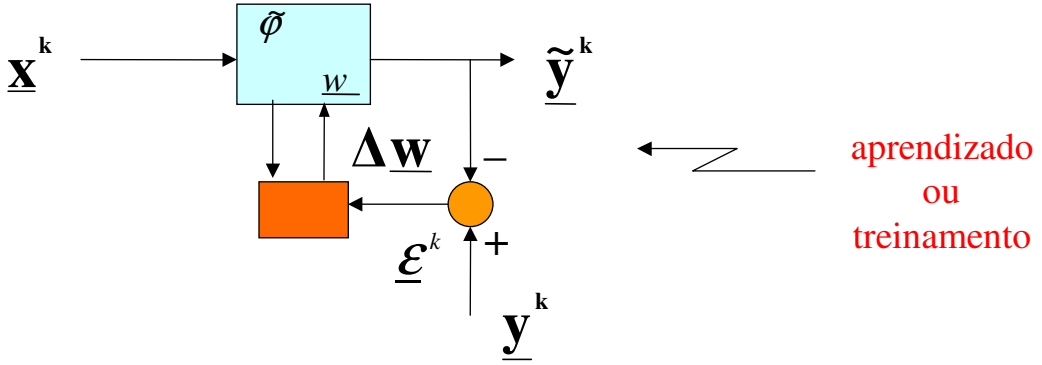
Simulação de Sistemas

Sistema Real, Planta

$$(\mathbf{x}^k, \mathbf{y}^k) \qquad k = 1,2,\ldots,P$$



Simulador



Aprendizado: Minimizar o erro na saída

Função objetivo à minimizar:

erro quadrático na saída para o k-ésimo par

$$\varepsilon^{2^k} = \left| \underline{y}^k - \underline{\tilde{y}}^k \right|^2 = \sum_{m=1}^{M} \left(y_m^k - \tilde{y}_m^k \right)^2$$

erro médio quadrático

$$F_0(\underline{w}) = \underline{E}_{k=1}^{P} \varepsilon^{2^k} = \underline{E}_{k=1}^{P} \left| \underline{y}^k - \underline{\tilde{y}}^k \right|^2 = \underline{E}_{k=1}^{P} \sum_{m=1}^{M} \left(y_m^k - \tilde{y}_m^k \right)^2$$

Problema:

Como encontrar o vetor $\underline{w_0}$ que minimize $F_0(\underline{w})$?

$$F_0(\underline{w_0}) \leq F_0(\underline{w}) \quad \forall \underline{w} \neq \underline{w_0}$$

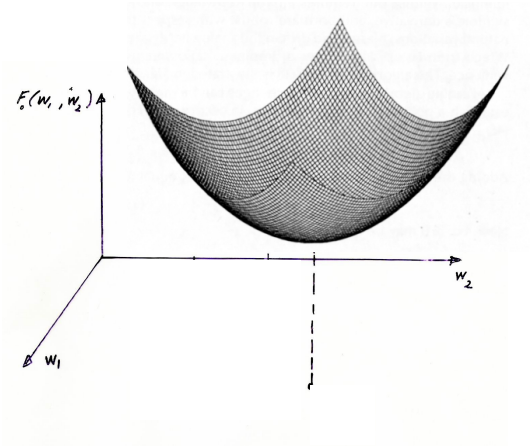
Solução:

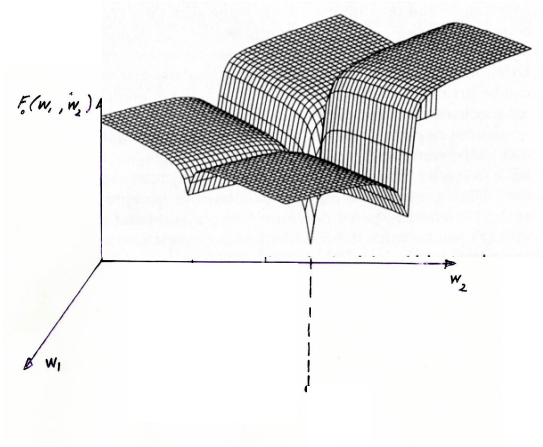
analítica - muito complicada

métodos numéricos recursivos

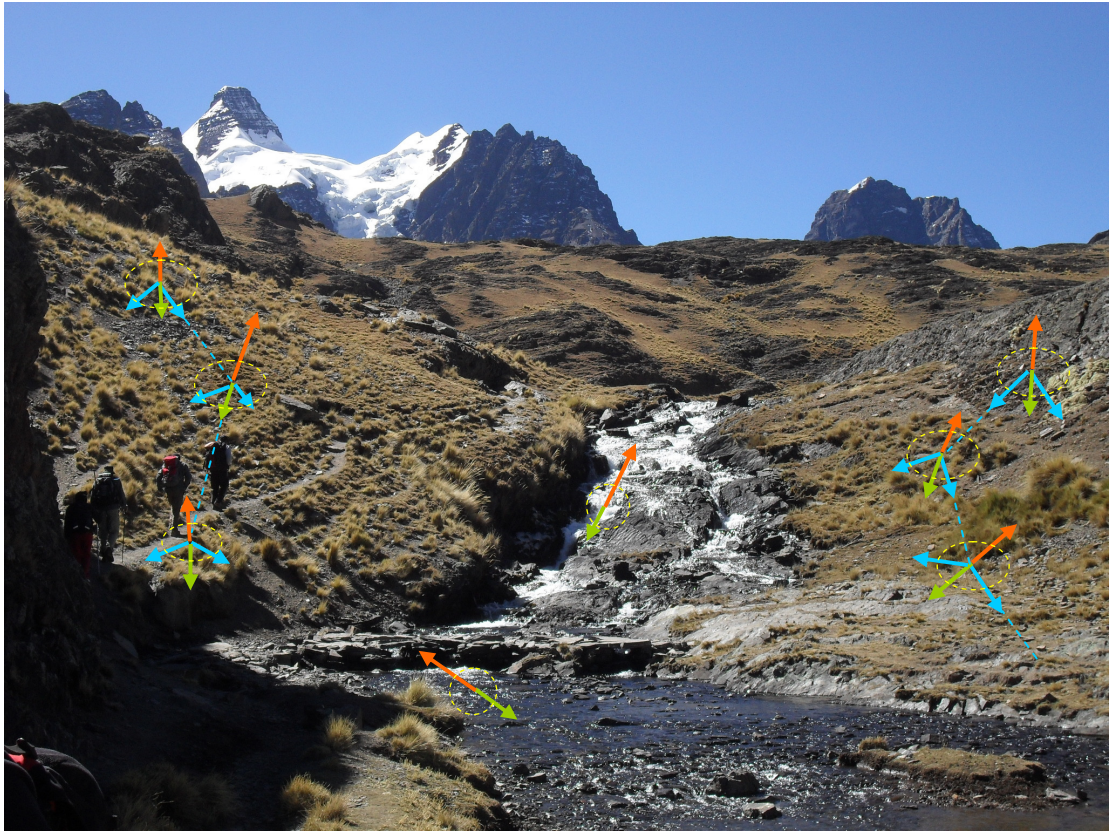
Minimização por métodos numéricos recursivos

$$\underline{w} \rightarrow \underline{w} + \Delta\underline{w} \quad | \quad F(\underline{w} + \Delta\underline{w}) < F(\underline{w})$$





Descida continuada, permanente - decisão local



Que direção diminui F ?

Gradiente:

$$F_0(w_1, w_2, \cdots) = F(\underline{\mathbf{w}})$$

$$\underline{\mathbf{w}} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \end{bmatrix} \qquad \nabla_{\underline{\mathbf{w}}} F_0 = \underline{\nabla} = \begin{bmatrix} \frac{\partial F_0}{\partial w_1} \\ \frac{\partial F_0}{\partial w_2} \\ \vdots \end{bmatrix}$$

Propriedades do Gradiente:

1. $\underline{\mathbf{w}} \rightarrow \underline{\mathbf{w}} + \Delta\underline{\mathbf{w}} \Rightarrow F_0 \rightarrow F_0 + \Delta F_0$

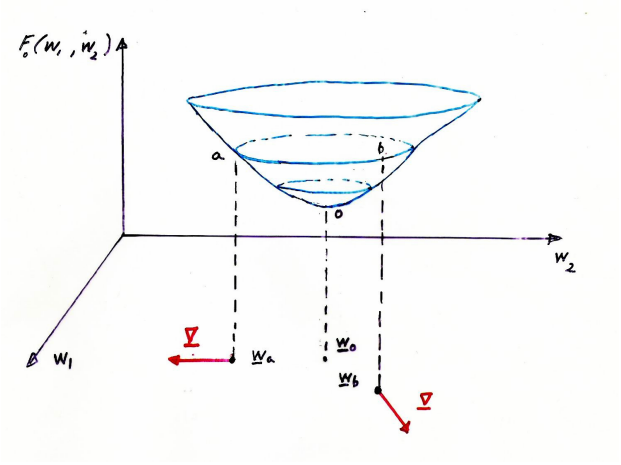
se $\Delta\underline{\mathbf{w}} = \alpha\underline{\nabla} \Rightarrow \Delta F_0$ é máximo

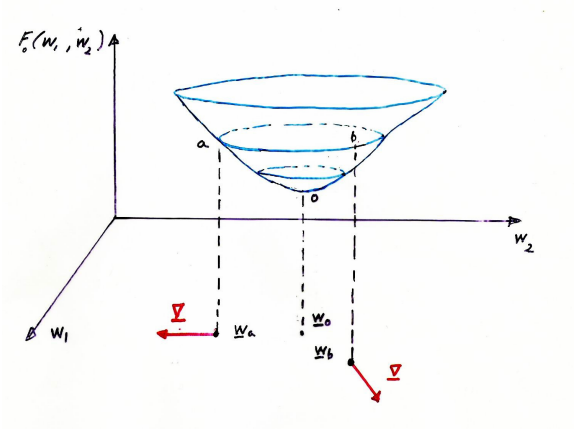
2. $\underline{\nabla}\Big|_{\underline{\mathbf{w}_0}} = \underline{\mathbf{0}}$

Gradiente descendente

ou descida por gradiente

$$\underline{w} \longrightarrow \Delta\underline{w} = -\alpha \nabla(\underline{w}) \longrightarrow \underline{w} = \underline{w} + \Delta\underline{w}$$





Complexidade de cálculo:

$$\Delta w_{ij} = -\alpha \frac{\partial F}{\partial w_{ij}}$$

Fim do processo:

$$\nabla \underline{w}_{otimo} = \underline{0} \quad \Rightarrow \quad \Delta \underline{w}_{otimo} = \underline{0}$$

Como calcular

$$\Delta w_{ij} = -\alpha \frac{\partial F_0(w_{ij})}{\partial w_{ij}}$$

???

$$1 - F_0(w_{ij}) = \underset{k}{E}\left[\left(\varepsilon^k\right)^2\right] = \frac{1}{K}\sum_{k=1}^{K}(\varepsilon^k)^2$$

$$\frac{\partial F_0(w_{ij})}{\partial w_{ij}} = \frac{\partial \underset{k}{E}\left[\left(\varepsilon^k\right)^2\right]}{\partial w_{ij}} = \underset{k}{E}\left[\frac{\partial\left(\varepsilon^k\right)^2}{\partial w_{ij}}\right]$$

2 – Para cada par entrada – saída k

como $(\varepsilon)^2 = \sum_{l=1}^{m} \left(\varepsilon_l\right)^2 = \sum_{l=1}^{m} \left(y_l - \tilde{y}_l\right)^2$ e $\frac{\partial y_l}{\partial w_{ij}} = 0$

então $$\frac{\partial (\varepsilon)^2}{\partial w_{ij}} = -2\, v_j\, \delta_i$$

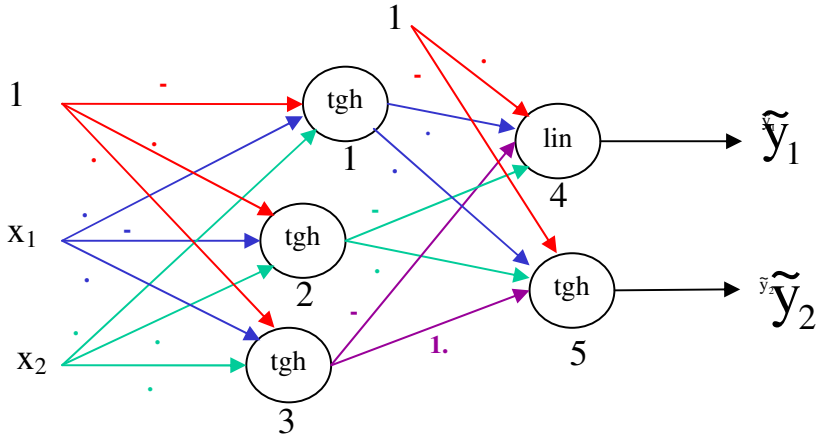
onde $$\delta_i = \sum_{l=1}^{m} \varepsilon_l \frac{\partial \tilde{y}_l}{\partial u_i} = \sum_{l=1}^{m} \varepsilon_l\, g_{li}$$

3 - e então:

$$\Delta w_{ij} = -\alpha \frac{\partial F_0\left(w_{ij}\right)}{\partial w_{ij}} = 2\alpha \underset{k}{E}\left(v_j\, \delta_i\right)$$

Algoritmo:

1 - Rede Original:



Propagar o sinal na rede original e conservar o valor do sinal v_j na entrada de cada sinapse w_{ij}.

2 – Criar uma "Rede Associada" a partir da Rede Original

2.1 - "Linearizar" a rede original

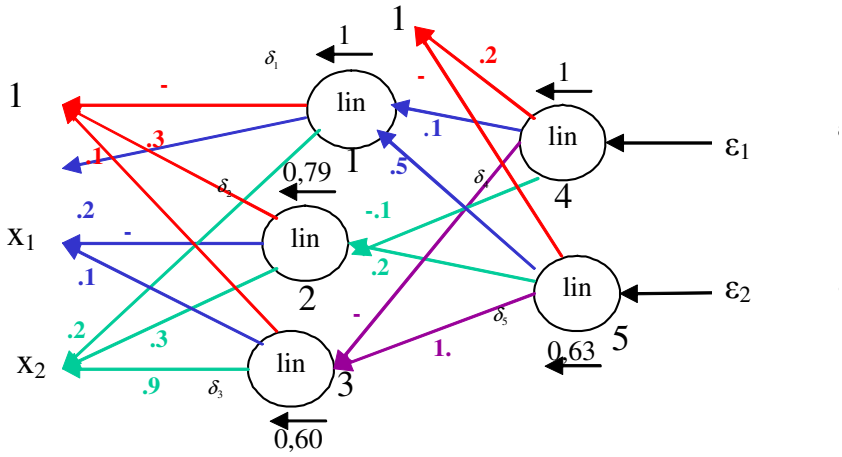
Neurônio não linear>>> Neurônio linear

$v_0 = \tgh(u_0)$ >>> $v = (1-v_0^2)\, u$

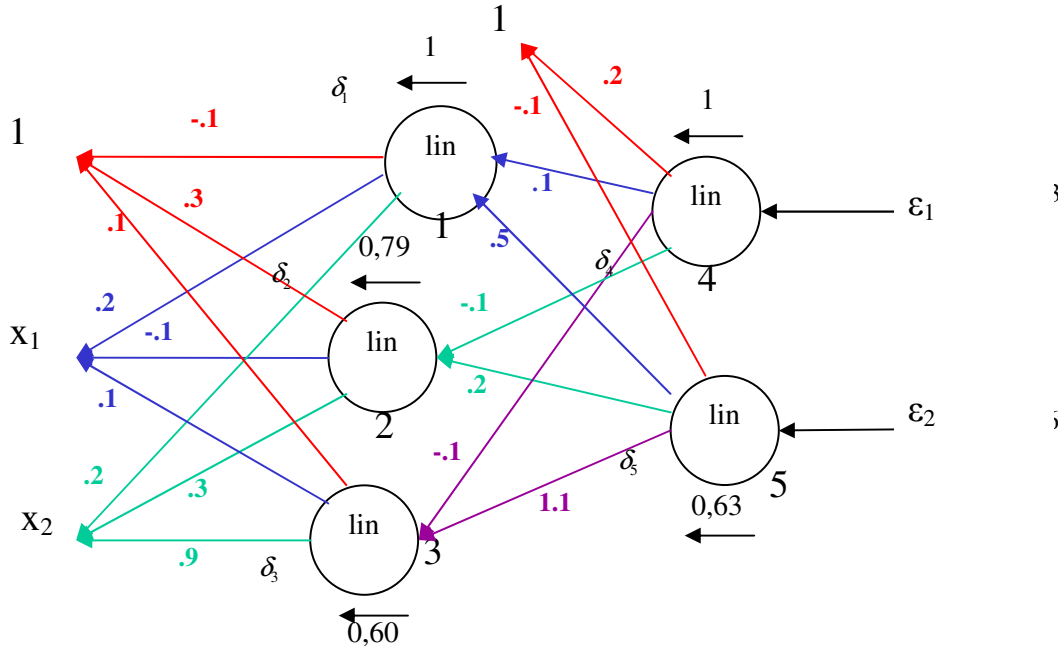
Neurônio linear >>> Neurônio linear

$v_0 = u_0$ >>> $v = u$

2.2- Inverter todos os sentidos de transmissão



3 – (Retro)propagar o erro ε_l em cada saída l através da rede "associada" e conservar o valor do sinal δ_i de erro retropropagado que chega na saída (original) de cada sinapse w_{ij}.



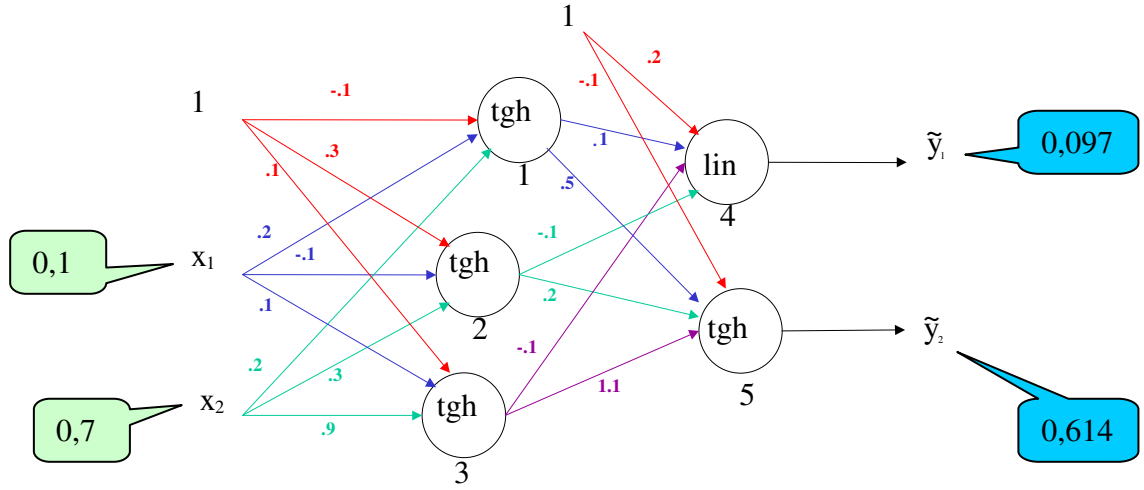
4 – O acréscimo para cada sinapse wij para o par k entrada-saída em operação é dado por:

$$\Delta_k w_{ij} = 2\ \alpha\ v_j\ \delta_i$$

5 – O acréscimo a ser aplicado na sinapse w_{ij} é o valor esperado dos acréscimos calculados para cada sinapse,

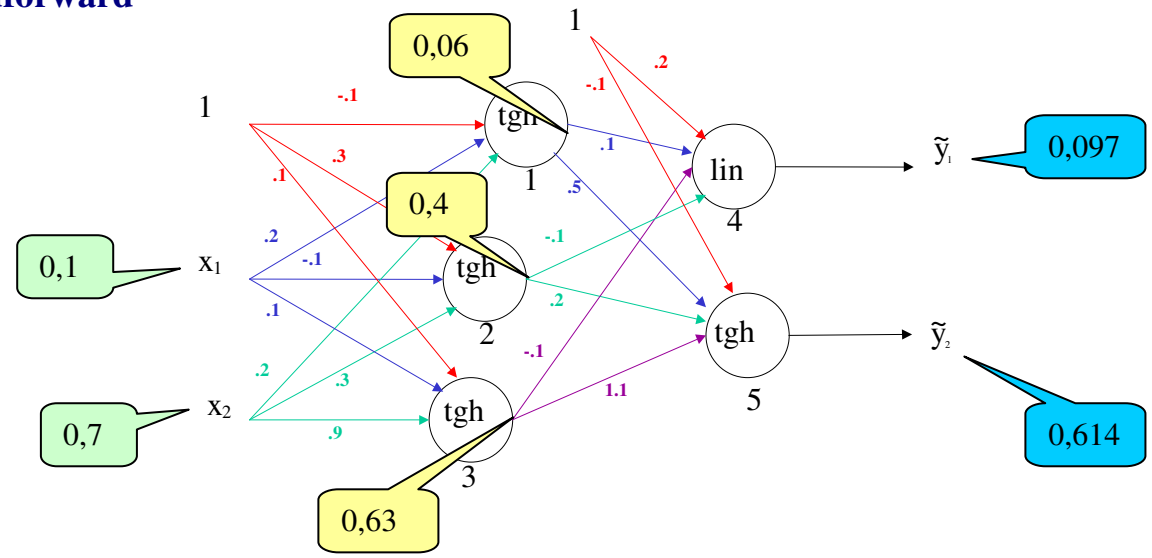
$$\Delta w_{ij} = E\left(\Delta_k w_{ij}\right) = 2\ \alpha\ E\left(v_j \delta_i\right)$$

Exemplo anterior



$$\underline{\mathbf{x}} = \begin{bmatrix} 0,1 \\ 0,7 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 0,2 \\ 1 \end{bmatrix} \qquad \tilde{\mathbf{y}} = \begin{bmatrix} 0,097 \\ 0,614 \end{bmatrix} \qquad \varepsilon = \begin{bmatrix} 0,103 \\ 0,386 \end{bmatrix}$$

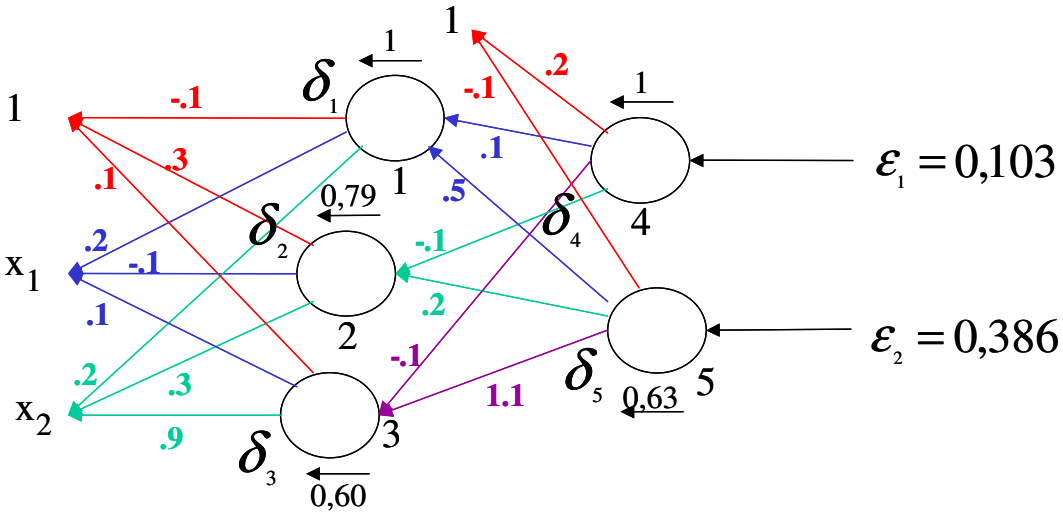
Signal Feedforward

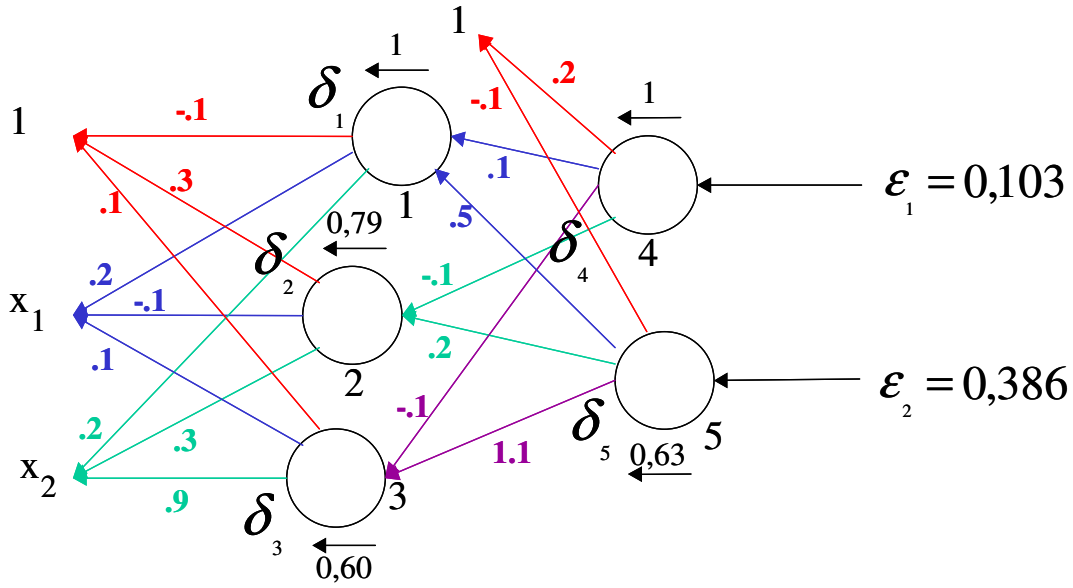


v_j $\tilde{y}_i$ ε_i

Rede associada para a

retropropagação do erro:





$$\delta_5 = (0,386)(0,63) = 0,24 \qquad \delta_4 = (0,103)(1) = 0,103$$

$$\delta_3 = [(0,103)(-0,1) + (0,24)(1,1)](0,60) = 0,152$$

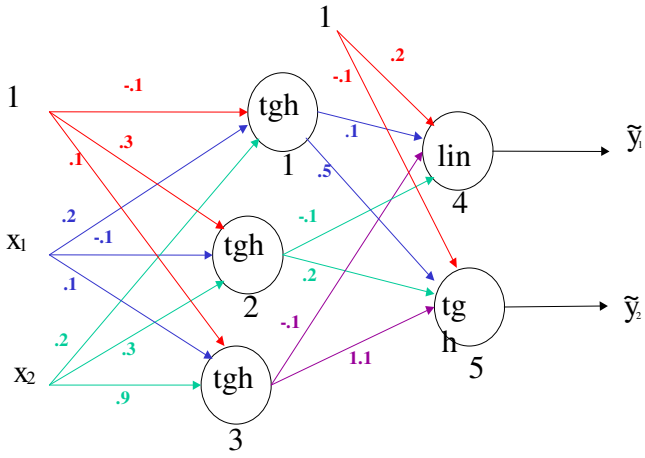
$$\delta_2 = [(0,103)(-0,1) + (0,24)(0,2)](0,79) = 0,030$$

$$\delta_1 = [(0,103)(0,1) + (0,24)(0,5)](1) = 0,130$$

Treinamento Regra Delta

Atualização dos valores das sinapses:

$$\Delta w_{ij} = 2\alpha v_j \delta_i$$



$$\Delta b_4 = (2)(0,1)(1)(0,103) = 0,021$$

$$b_4 \rightarrow 0,2 + 0,021 = 0,221$$

$$\Delta b_1 = (2)(0,1)(1)(0,130) = 0,026$$

$$b_1 \rightarrow -0,1 + 0,026 = -0,074$$

$$\Delta w_{41} = (2)(0,1)(0,06)(0,103) = 0,001$$

$$w_{41} \rightarrow 0,1 + 0,001 = 0,101$$

$$\Delta w_{3b} = (2)(0,1)(0,7)(0,152) = 0,021$$

$$w_{3b} \rightarrow 0,9 + 0,021 = 0,921$$

Alguns detalhes e comentários sobre o processo

1 - Escolha das variáveis de entrada

2 – Escalamento das variáveis

3- Dimensionando a rede; valores iniciais

4 – Acompanhamento do treinamento pela evolução do erro

5 – Generalização: overtraining, validação cruzada

6 – Crítica pos treinamento

7 - Outros tipos de erro: percentual, ponderado, etc.

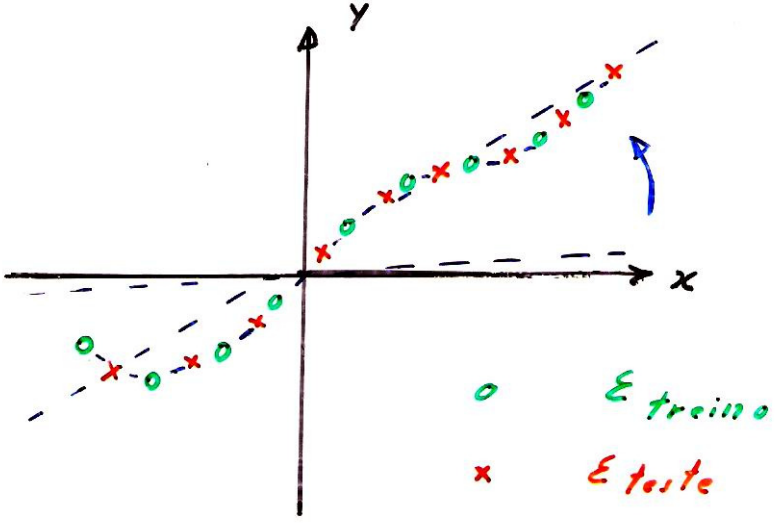
8 – Outros Processos de Treinamento: passo variável, 2ª ordem

A rede neural como um aproximador (saídas contínuas)

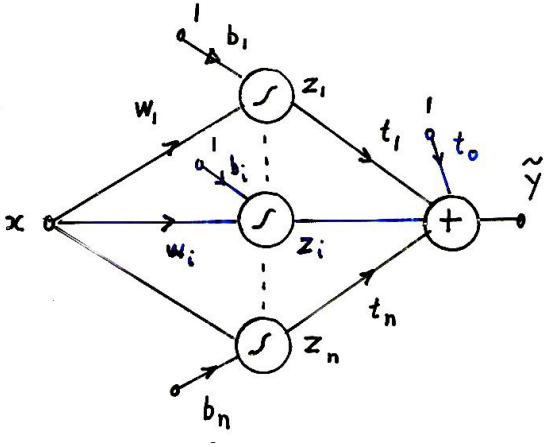
Demo Aproximador

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

$y = \varphi(x)$ $\quad\quad$ $\tilde{y} = \varphi(x)$



Rede Neural

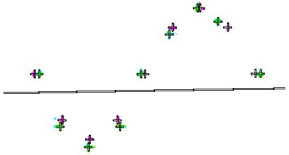


número de neurônios na camada intermediária ajustável
passo treinamento α ajustável
decaimento do passo α ajustável
valores iniciais ajustáveis

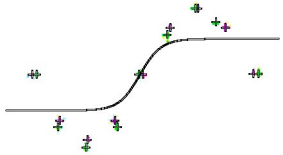
Treinamento e

Efeito do numero de neurônios na camada intermediaria

Inicio do treinamento

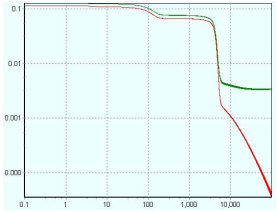


fim, 1 neuronio

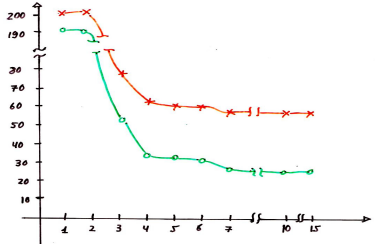


fim, 3 neuronios

Evolução do erro



Erro final vs # neuronios



Alguns exemplos de aplicações:

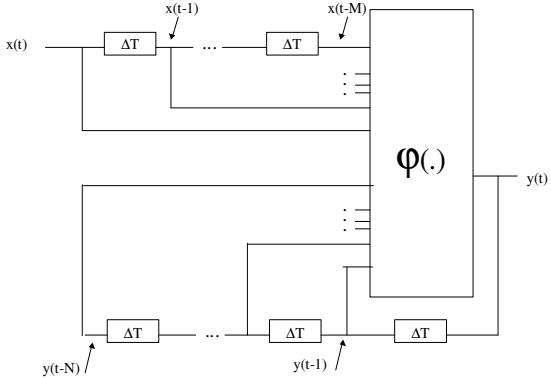
Aproximadores (saídas contínuas):

- Modelagem de um Conversor Siderúrgico da CSN
- Modelagem de 3 plantas de destilarias da Petrobras
- Previsão de atenuação em enlace de telefonia celular e em enlace de comunicação terra-satelite
- Previsão de atrazo no pagamento por clientes (pessoas e instituições)
- Previsão do PIB brasileiro
- Previsão de volatilidade de opções Telebrás
- Previsão de consumo elétrico domiciliar
- Previsão de demanda de pico de energia

Redes Neurais na Modelagem de Sistemas Dinâmicos e Séries Temporais.

Modelo ARMA

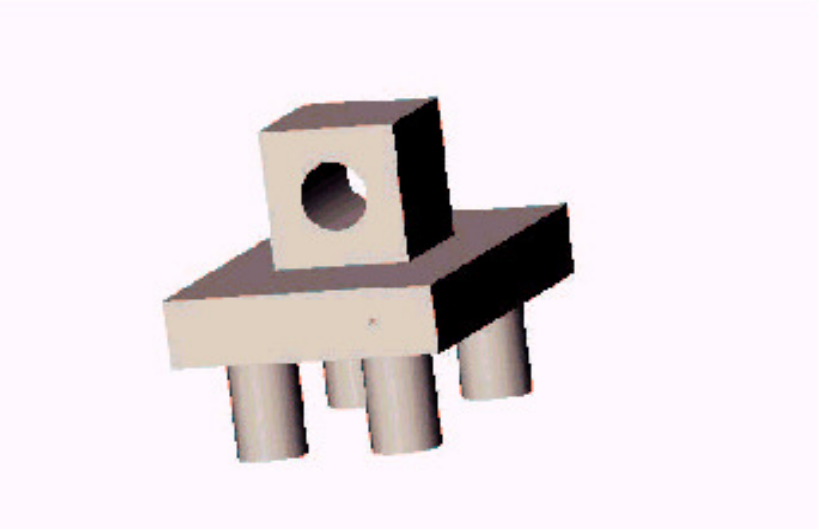
$$\tilde{y}(t) = \sum_{i=0}^{M} b_i \, x(t-i) + \sum_{i=1}^{N} a_i \tilde{y}(t-i)$$



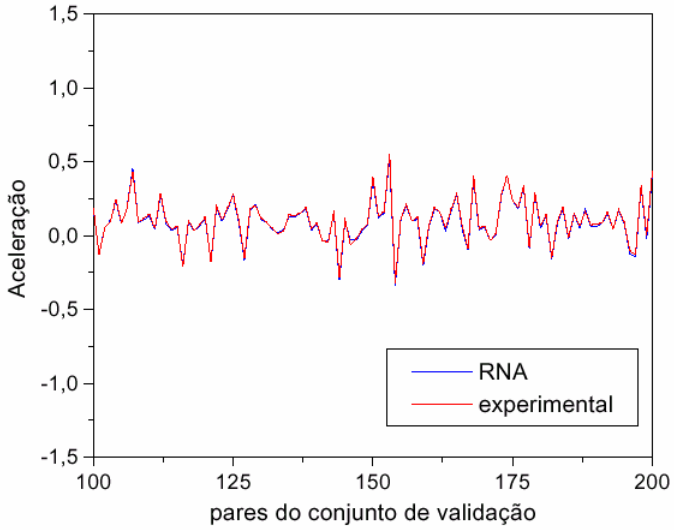
Modelo NARMA

$$y(t) = \varphi[x(t), x(t-1), \cdots, x(t-M), y(t-1), \cdots, t(t-N)]$$

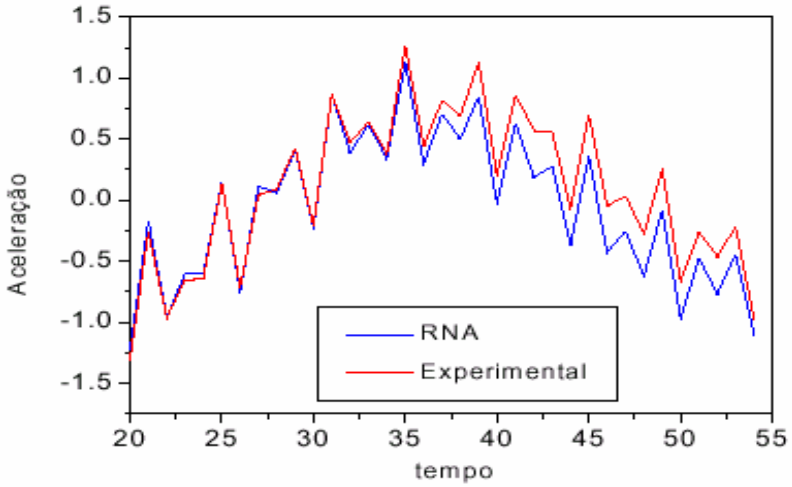
Exemplo:



Operação em serie - paralelo



Operação em paralelo



Uso em Series Temporais

S(t) = Tendência +

Sazonalidade +

Ciclos senoidais +

Outros artefatos deterministicos +

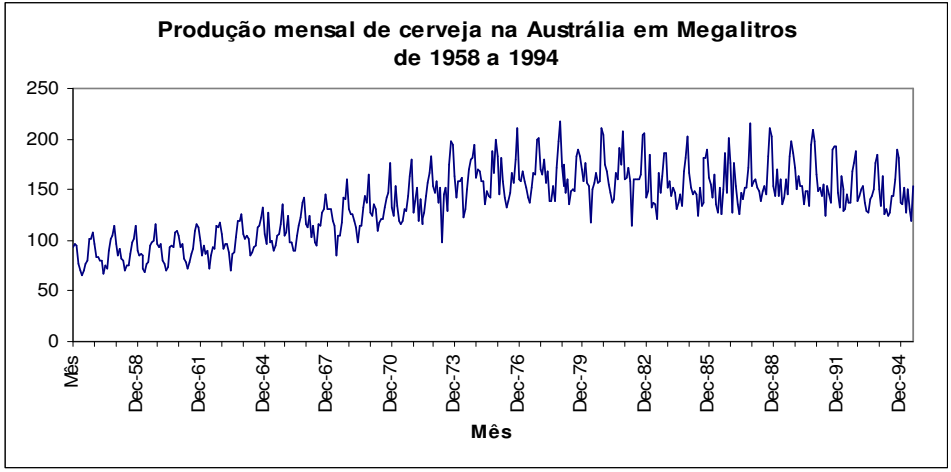
Componentes não lineares +

Ruido não correlato

Trabalhar sobre a serie residual !

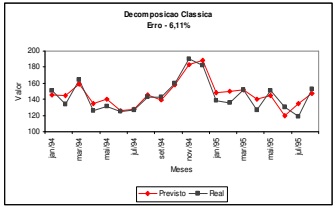
Series Temporais

Exemplo:

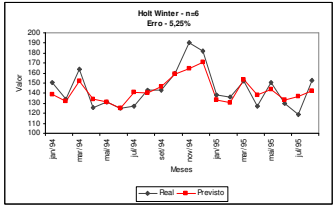


Resultados (período de teste):

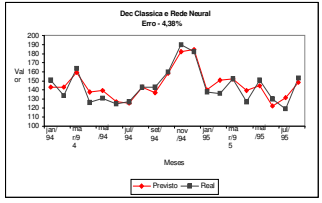
Convencional



Holt-Winters



RNs



	Decomp.	Holt Winter	RN
1º Ano	4.71%	4.81%	3.68%
20 Meses	6.11%	5.25%	4.35%

Uso em modelagem:

Estudo da importância das entradas:

Relevância de entradas

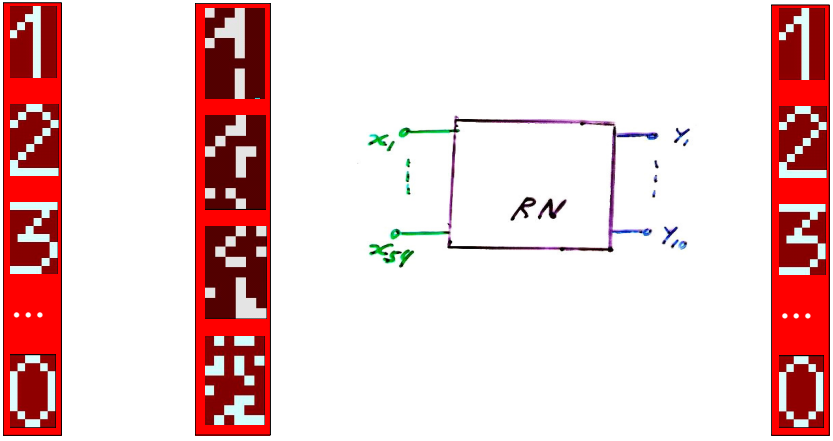
Estabelecimento de modelos:

Pruning: Simplificação de estruturas para obtenção das equações do modelo fenomenológico

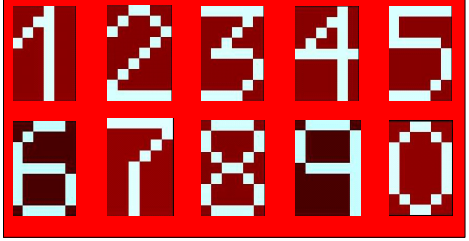
Modelos híbridos neural-fenomenológicos

A rede neural como classificador (saídas lógicas)

Demo: OCR

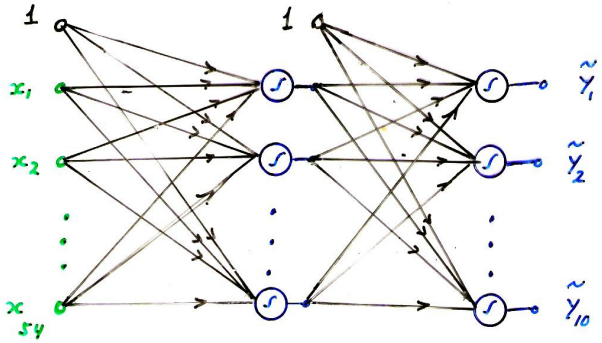


Padroes



Entradas: Padrões + 20% de ruido

Rede Neural



dim $\underline{x}$ = 54 dim $\underline{y}$ =10 componentes binarias

2 camadas de neuronios

neurônios camada intermediaria: 1 – 30

Passo treinamento .1 , max 500 epocas

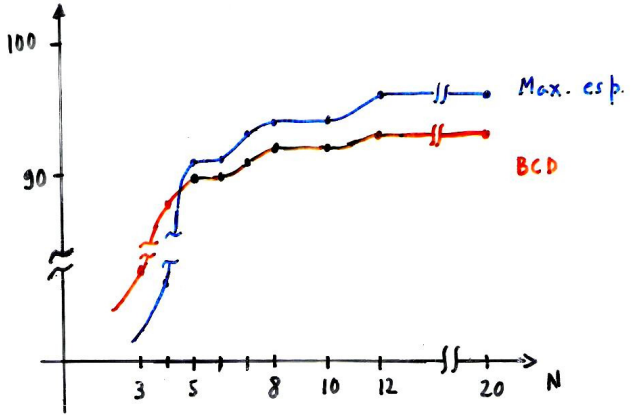
Ruído: 0 – 20 %

Capacidade de Mapeamento

$R_{treino} = R_{teste} = 15\ \%$

1 camada >>> 77 % acertos

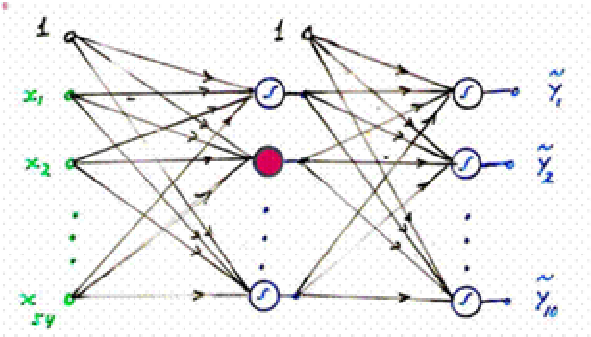
2 camadas f (N) N = # neurônios cam. intermed.

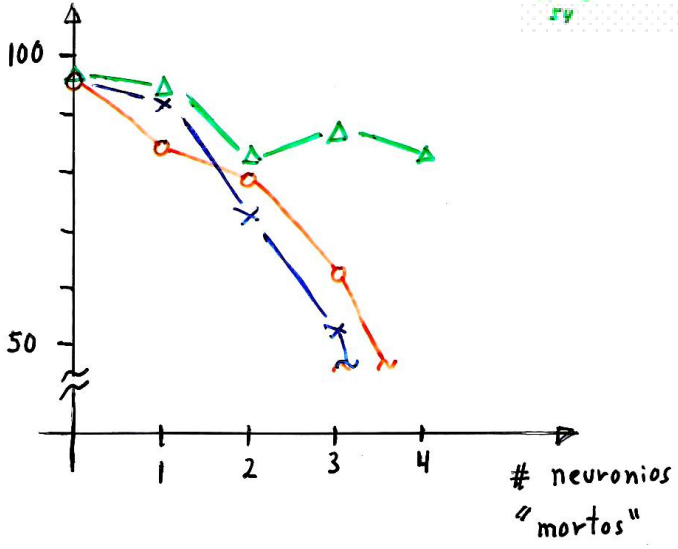


Robustez

Morte de neurônios

Neurônio morto: y = 0





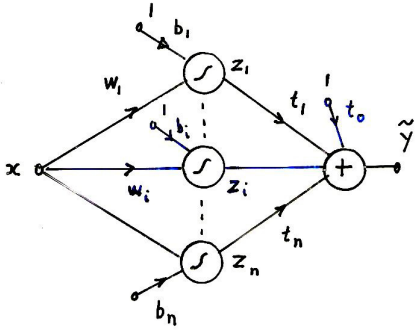
Alguns exemplos de aplicações:

Classificadores (saídas lógicas):

- Identificação do locutor e do conteúdo da alocução
- Identificação da confiabilidade de informantes
- Identificação de navios baseado em sinais de sonar passivo e de detetor magnético
- Identificação de malignidade em anomalias na mama
- Identificação de patologias em ECG
- Identificação de partículas em detectores usados no estudo de física de altas energias
- Identificação não invasiva de consumo elétrico domiciliar instantâneo
- Classificação de defeitos em soldas via analise radiográfica e por ultra-som
- Classificação de falhas em linhas de transmissão de energia

SVM – Máquinas de Vetor Suporte

$$\tilde{y} = \sum_j \varphi_j(\vec{x})$$



Polinomial $\varphi_j = P_j(\vec{x})$

Perceptrons $\varphi_j = t_j tgh\left(\vec{w}_j^t \vec{x} + w_{j0}\right)$

Base radial gaussiana $\varphi_j = t_j \exp(-\left\|\vec{x} - \vec{w}_j\right\|^2 / 2\sigma_j^2)$

etc. $\varphi_j = \ldots$

Parte II

Camada de Kohonen,
Rede ART,
Rede Counterpropagation,
QV

Aprendizado cego, não supervisionado.

Plasticidade.

Camada de Kohonen

Rede ART, Rede CounterPropagation

- **Realizam:**

  Classificadores por similaridade

- **Características importantes:**

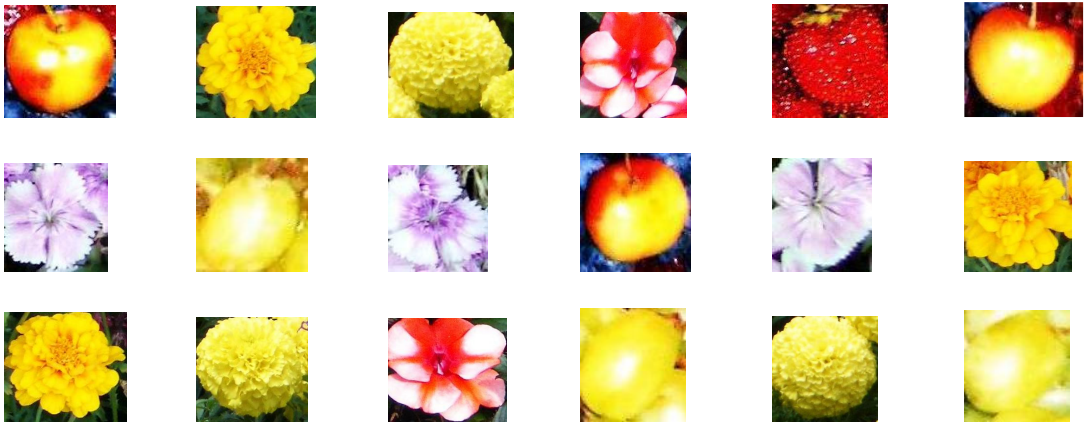
  Auto organizáveis, i.e., aprendizado não supervisionado
  Aplicações on-line
  Plasticidade

- **Aplicações:**

  Reconhecimento de padrões;
  Estatística de sinais;
  Memórias associativas;
  Filtragem não-linear;
  Compressão de dados;
  Mapas auto-organizáveis;
  etc...

Classificação por Similaridade

Classes agrupam elementos `similares` entre si



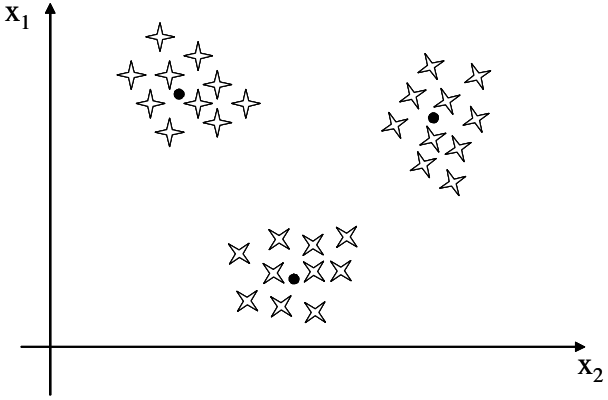
Classificação por Similaridade

objetos físicos $\bigcirc_1$ $\bigcirc_2$ ⇒ objetos matemáticos $\underline{\mathbf{x}}_1$ $\underline{\mathbf{x}}_2$

similaridade física ⇒ similaridade matemática

$\bigcirc_1 \approx \bigcirc_2$ $\qquad$ $\underline{x}_1 \cong \underline{x}_2$ ou $\left|\underline{x}_1 - \underline{x}_2\right| <<$

Classificadores por similaridade

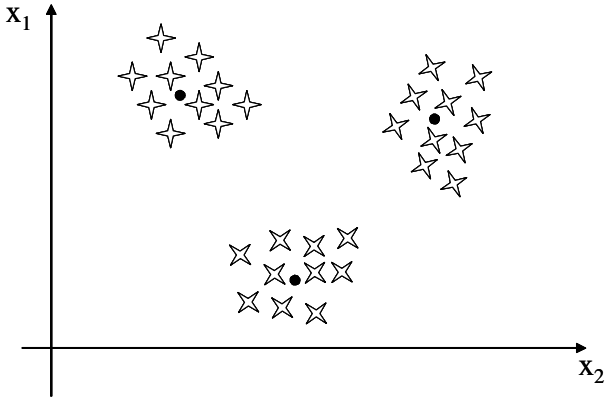


Critério básico

Dois elementos pertencem à mesma classe se

$$\left| x_1 - x_2 \right| \leq \varepsilon$$

pouco prático

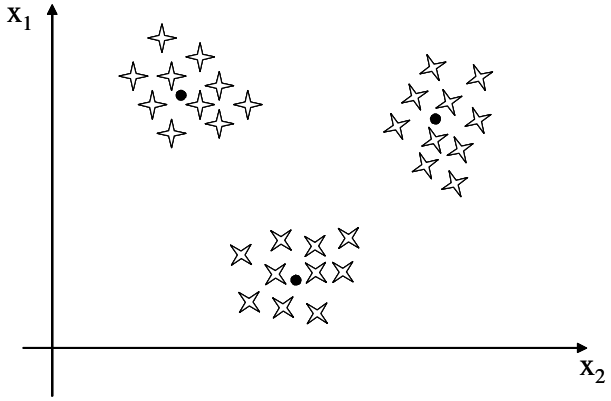


Padrão de classe:

Padrão $\underline{w}_i$ da classe C_i

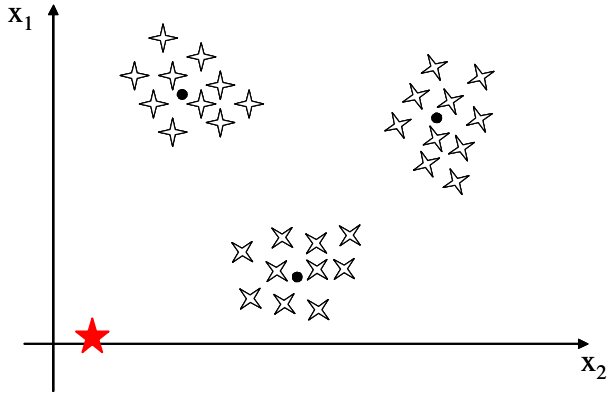
$$\underline{w}_i = \underset{\forall \underline{x}_j \in C_i}{E} \underline{x}_j = \frac{1}{N_i} \sum_{\forall \underline{x}_j \in C_i,\, j=1}^{N_i} \underline{x}_j$$

Critério 1 : Padrão mais similar à entrada



$$\underline{x}_i \in C_i \quad \text{sse} \quad \left|\underline{x}-\underline{w}_i\right|^2 < \left|\underline{x}-\underline{w}_j\right|^2 \quad \forall\, j \neq i$$

Eq (1)



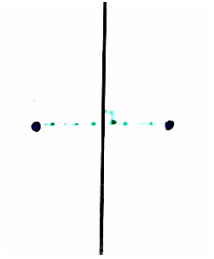
Criterio 2 – Padrao que satisfaz uma similaridade minima

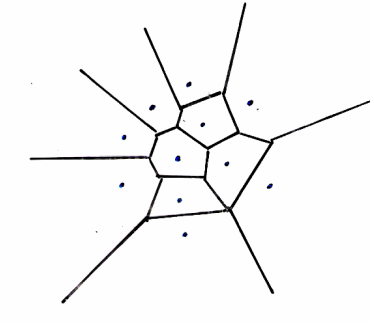
$$\underline{x}_i \in C_i \quad \text{sse} \quad \left|\underline{x} - \underline{w}_i\right|^2 < r_0^2 \qquad \text{Eq (2)}$$

r_0 – raio de similaridade

Separadores

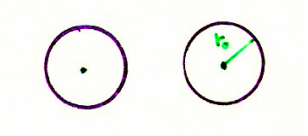
Critério 1 - padrão mais similar (mais próximo) a entrada Eq(1)

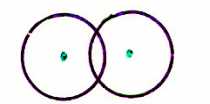




Tecelagem de Voronoi

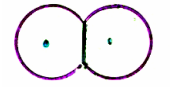
Critério 2 - mínima similaridade exigida Eq (2)

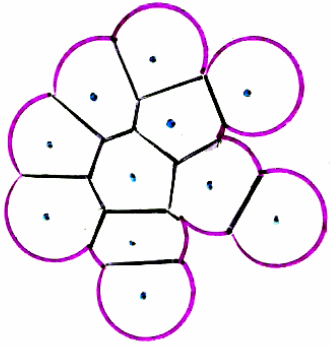




Critérios 1 e 2, Eq(1) + Eq(2)

padrão mais similar & mínima similaridade atingida

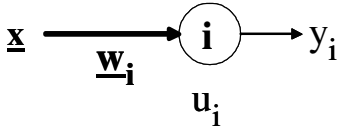




Neurônio como medidor de similaridade

uma outra definição

$$u_i = -\left\| \underline{x} - \underline{w}_i \right\|^2 = -d_i^2 \leq 0$$



u_i - medida de similaridade entre $\underline{x}$ e $\underline{w}_i$

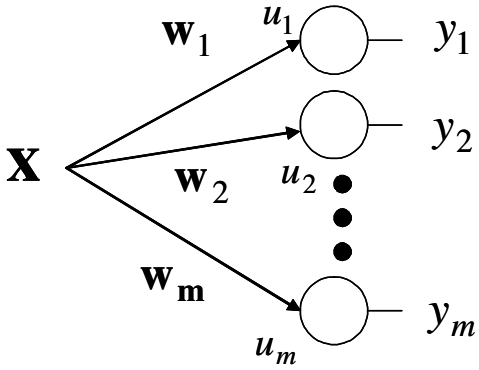
$u_i = 0$ distância nula = máxima similaridade

y_i – depende dos outros neurônios

Template Matching

$$u_i = - d_i^2$$

maior u_i =
menor distância =
maior similaridade

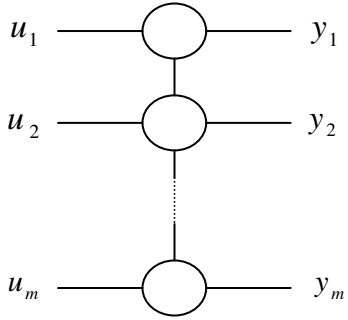


u_i é uma medida de similaridade entre x e w_i

Winner-takes-all

$$y_i = 1 \quad sse \quad u_i > u_j \quad \forall\, j \neq i$$

$$y_i = 0 \qquad caso\ contrário$$



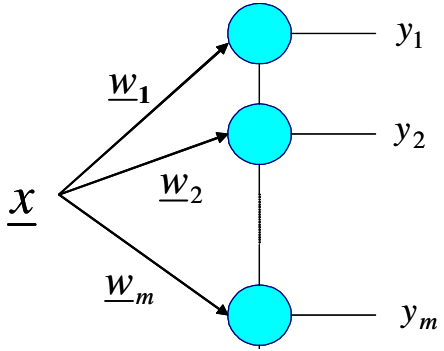
Camada de Kohonen

[versão unidimensional, simplificada (D=0)]

Classe C_i

Padrao $\underline{w}_i$

Indicador y_i

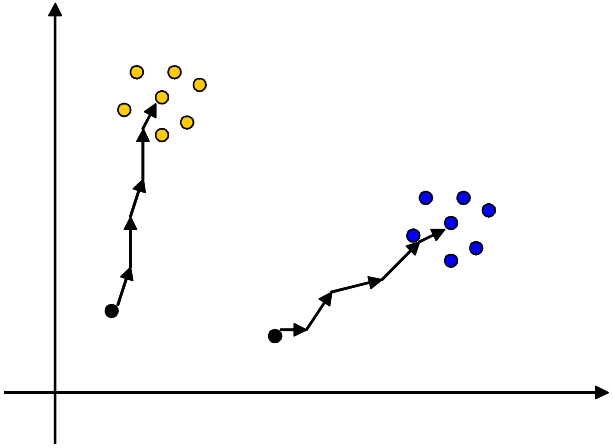


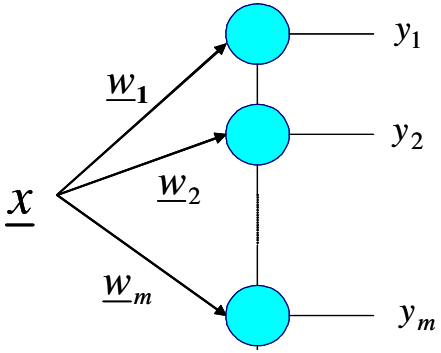
Se $y_i = 1$ então $\underline{x} \in C_i$

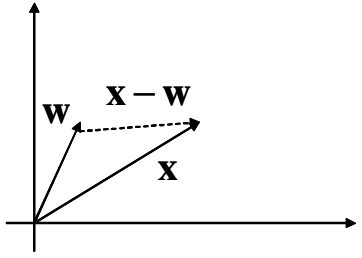
pelo critério 1 (padrão mais similar a entrada).

Camada de Kohonen

Treinamento Supervisionado





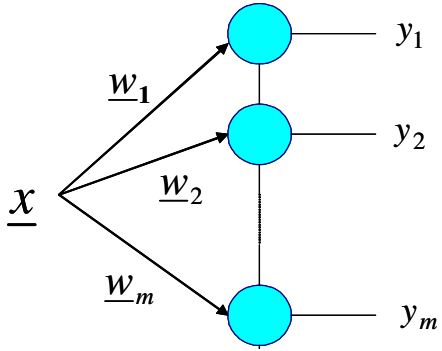


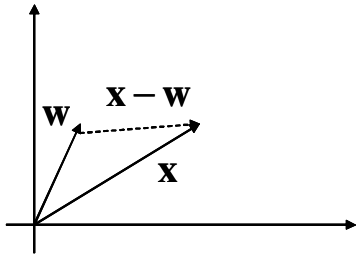
Treinamento da Camada de Kohonen

$$\underline{x}(n) \in C_i$$

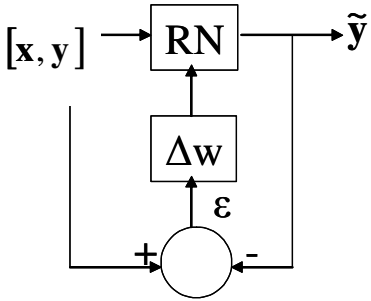
$$\underline{w}_i(n+1) = \underline{w}_i(n) + \alpha[\,\underline{x}(n) - \underline{w}_i(n)\,]$$
$$= (1-\alpha)\,\underline{w}_i(n) + \alpha\,\underline{x}(n)$$

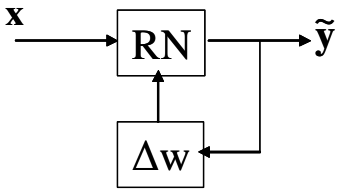
$$\underline{w}_j(n+1) = \underline{w}_j(n) \quad \forall \; j \neq i$$

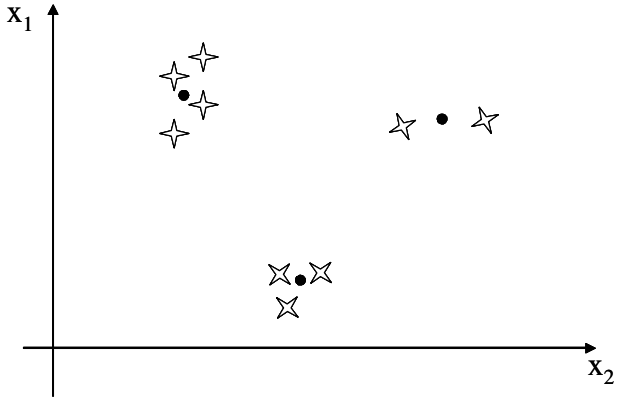




Treinamento supervisionado e não-supervisionado

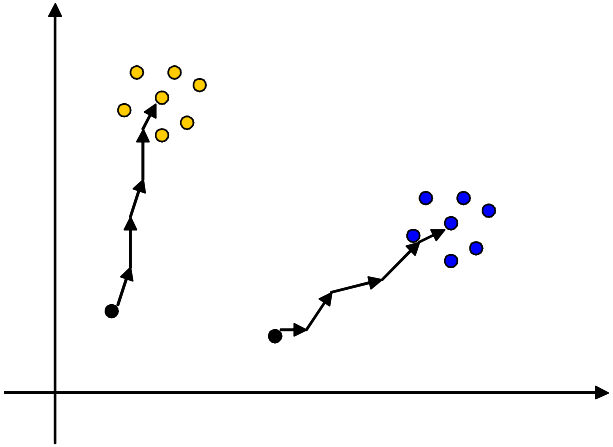


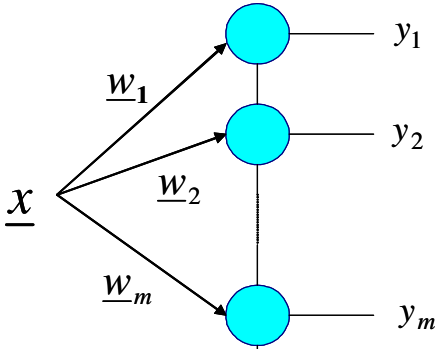




Camada de Kohonen

Treinamento não supervisionado





Camada de Kohonen

Treinamento não supervisionado

$$\underline{x}(n) \quad >>>> \quad y_i = 1$$

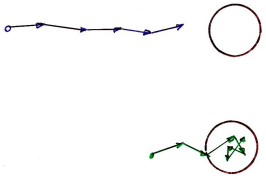
$$\begin{aligned} \underline{w}_i(n+1) &= \underline{w}_i(n) + \alpha[\,\underline{x}(n) - \underline{w}_i(n)\,] \\ &= (1-\alpha)\,\underline{w}_i(n) + \alpha\,\underline{x}(n) \end{aligned}$$

$$\underline{w}_j(n+1) = \underline{w}_j(n) \quad \forall \; j \neq i$$

Treinamento competitivo

Fim do treinamento ?

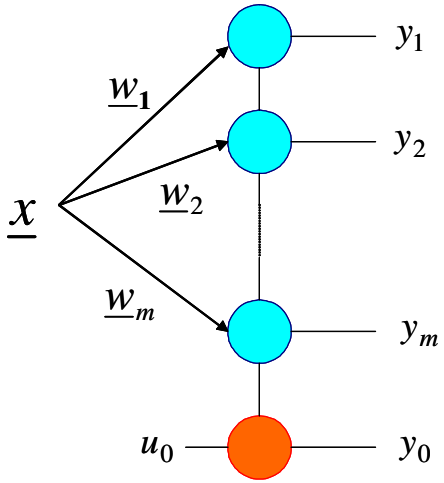
$$E\left[\underline{\Delta w}\right] = \underline{0}$$



E o segundo critério (similaridade mínima) ?

Camada de Kohonen (aumentada)

$$u_0 = - r_0^2$$



Se $y_i = 1$ entao

$\underline{x} \in C_i$

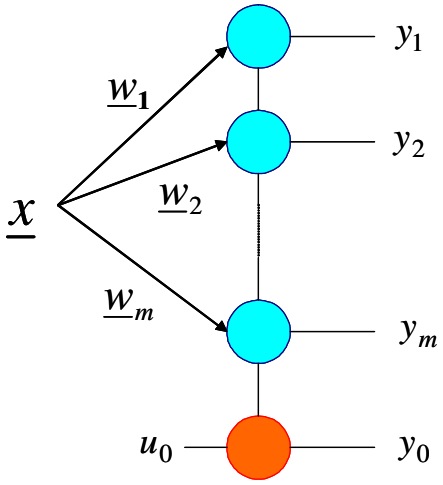
pelos critérios 1 (padrao mais similar a entrada) e

2 (satisfaz a similaridade minima exigida).

Se $y_i = 1$ então

$$\underline{x} \in C_i$$

pelos critérios
1 (centro de classe mais similar à entrada) e
2 (satisfaz à similaridade mínima)



Se $y_0 = 1$ então

$$\underline{x} \notin C_i \quad \forall\, i$$

$\underline{x}$ não satisfaz ao critério 2
para nenhuma classe

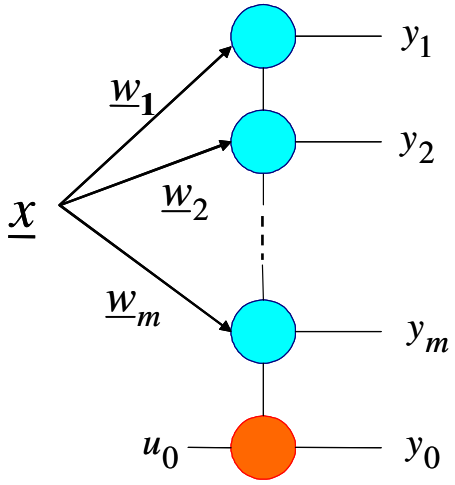
Rede ART (modificada)

Condições iniciais:

$u_0 = - r_0^2$ e

todos os neurônios $\underline{w}_i$ (0)

estão desativados



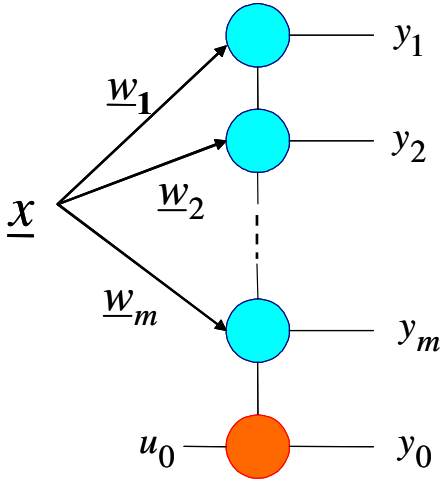
Obs: neurônio desativado não tem condição de ganhar a competição:

$\underline{w}$ não existe na rede ou $|\vec{w}| >> |\vec{x}| \ \forall \ \vec{x}$

Treinamento:

Apresentar entrada $\underline{x}$ (n)

Neurônio vencedor y_0 ou y_i



Se $y_0(n) = 1$

Ativar um neurônio i desativado

$$\underline{w}_i(n) \text{ desativado} \Rightarrow \underline{w}_i(n+1) = \underline{x}(n)$$
$$\underline{w}_j(n+1) = \underline{w}_j(n) \quad \forall \; j \neq i$$

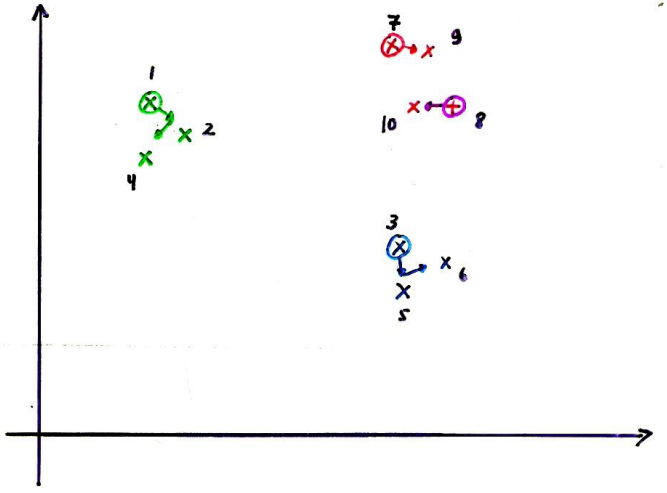
Se $y_i(n) = 1$

Treinar o neurônio vencedor, i

$$\underline{w}_i(n+1) = \underline{w}_i(n) + \alpha[\underline{x}(n) - \underline{w}_i(n)]$$
$$\underline{w}_j(n+1) = \underline{w}_j(n) \quad \forall \; j \neq i$$

Plasticidade, Treinamento competitivo

Treinamento ART



Treinamento ART

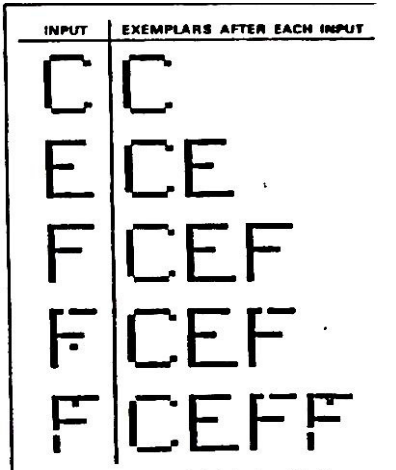
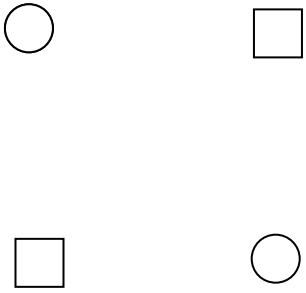


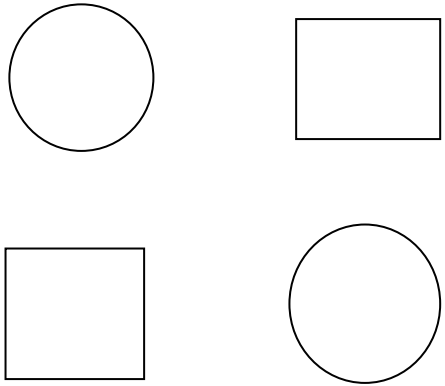
Figure 11. An example of the behavior of the Carpenter Grossberg net for letter patterns. Binary input patterns on the left were applied sequentially starting with the upper "C" pattern. Exemplars formed by top-down connection weights after each input was presented are shown at the right.

Demo ART / Kohonen

1a



1b



ART

Esquecimento

Se $y_i = 0$ por N entradas consecutivas

Guarde a informação para o futuro como uma lembrança $\underline{\mathbf{m}}_k$, $\underline{\mathbf{m}}_k = \underline{\mathbf{w}}_i$

Desative o neurônio i, $\underline{\mathbf{w}}_i = \underline{\mathbf{0}}$

Esquecimento gradual

ART

Lembrança

Se $y_0 = 0$ Ativar neurônio, mas

Verifique se algum padrão estocado serve

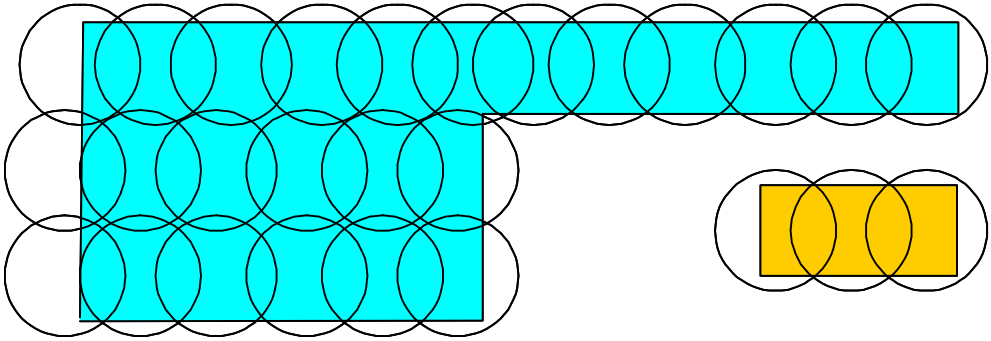
$\underline{\mathbf{x}}^t \underline{\mathbf{m}}_k > u_0$?

sim = existe lembrança $\underline{\mathbf{m}}_k$, ative $\underline{\mathbf{w}}_j = \underline{\mathbf{m}}_k$

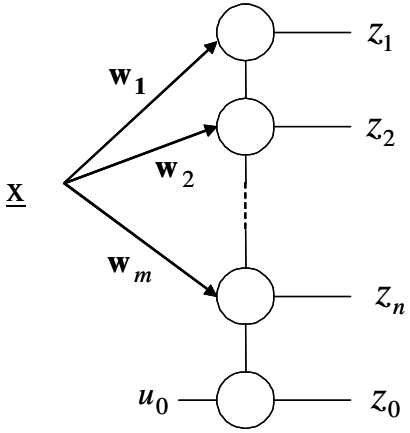
não = não existe lembrança, ative $\underline{\mathbf{w}}_j = \underline{\mathbf{x}}$

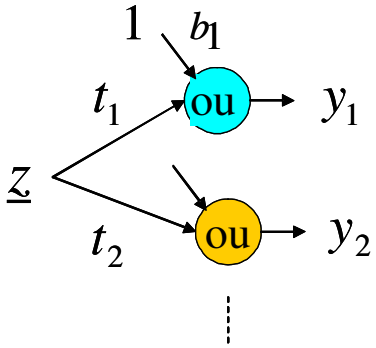
Classes Não-Esféricas

Treinamento Não-Supervisionado



- **1a. Etapa – ART – determina os domínios esféricos**
- **2a. Etapa – OU – determina os domínios vizinhos**





- **2a. Etapa – OU**

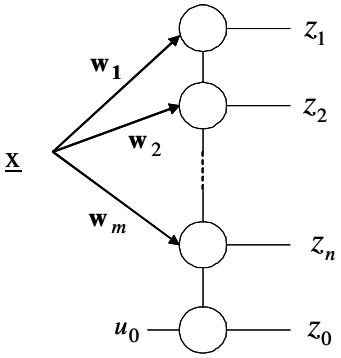
vizinhos $\left|\underline{w}_i - \underline{w}_j\right| \leq 2\, r_0$

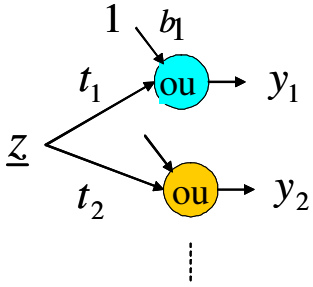
vizinhos de vizinhos

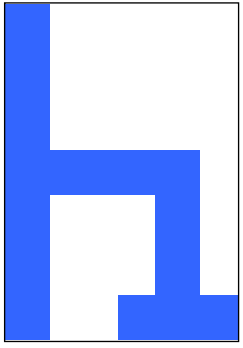
vizinhos de vizinhos de vizinhos

...

Demo:







A Speaker Verification Method Using LPC Singularity Location

Hardy L. C. P. Pinto **Rafael G. C. P. Pinto** **Luiz P. Calôba**

Representação LPC dos fonemas /a/ e /i/

Cinco locutores masculinos

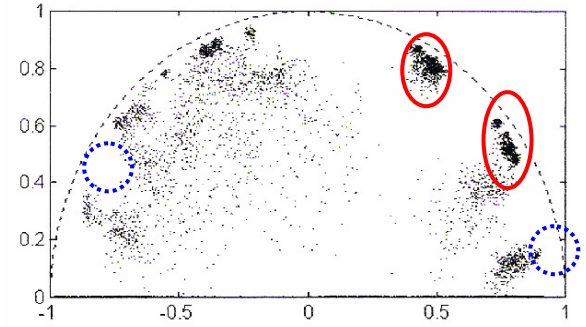


Figure 1- Roots of the predictor polynomial for segments of the sustained Portuguese phoneme /a/ for five different people of the same sex.

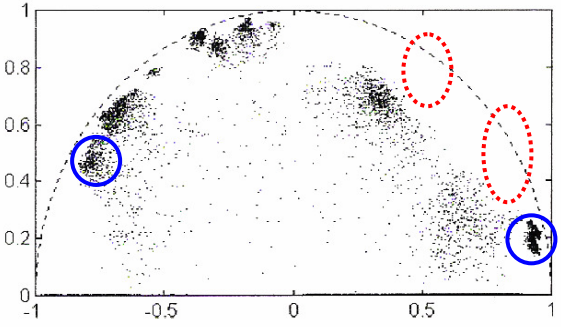


Figure 2- Same as fig. 1 for the phoneme /i/.

Diferenciação dos locutores

Fonema /a/ por dois locutores diferentes

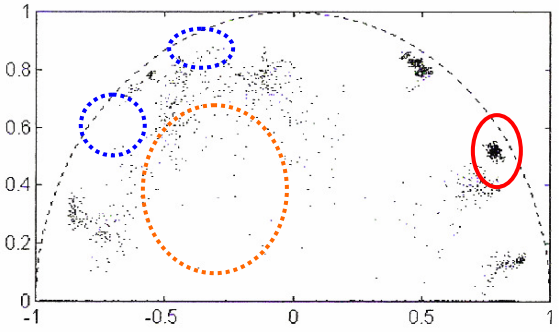


Figure 3 - Phoneme /a/ spoken by one speaker.

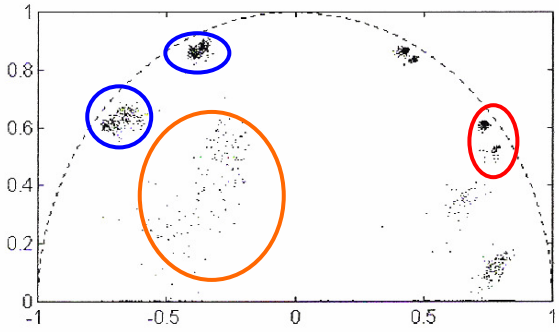


Figure 4 - Phoneme /a/ spoken by another speaker.

Rede ART modificada:

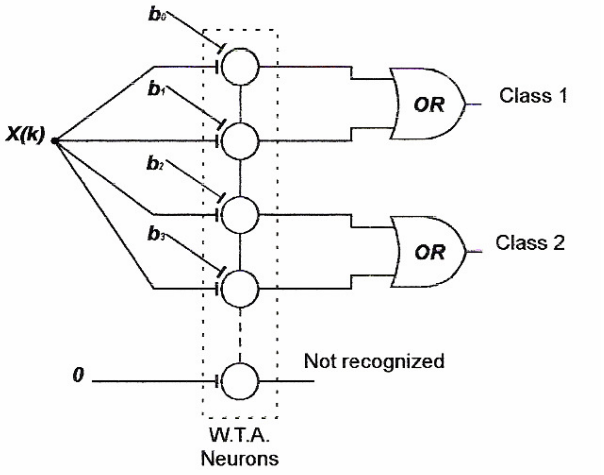


Figure 5 - Modified ART structure

Taxas de acerto de Fonemas:

TABLE I A - PHONEME TEST SET

Phoneme	Total segments	Correct	Not recognized	Wrong classif.
/a/	548	98.91 %	0.00 %	1.09 %
/e/	600	99.17 %	0.33 %	0.50 %
/i/	551	99.64 %	0.18 %	0.18 %
/o/	601	97.84 %	0.67 %	1.49 %
/u/	549	97.45 %	1.09 %	1.46 %

TABLE I B - CONFUSION TABLE FOR THE PHONEMES

	Recognized				
Spoken	/a/	/e/	/i/	/o/	/u/
/a/	98.90 %	0.37 %	0.00 %	0.73 %	0.00 %
/e/	0.17 %	99.17 %	0.00 %	0.00 %	0.33 %
/i/	0.00 %	0.00 %	99.63 %	0.00 %	0.18 %
/o/	1.50 %	0.00 %	0.00 %	97.84 %	0.00 %
/u/	0.00 %	0.18 %	1.09 %	0.18 %	97.45 %

Votação para Locutores:

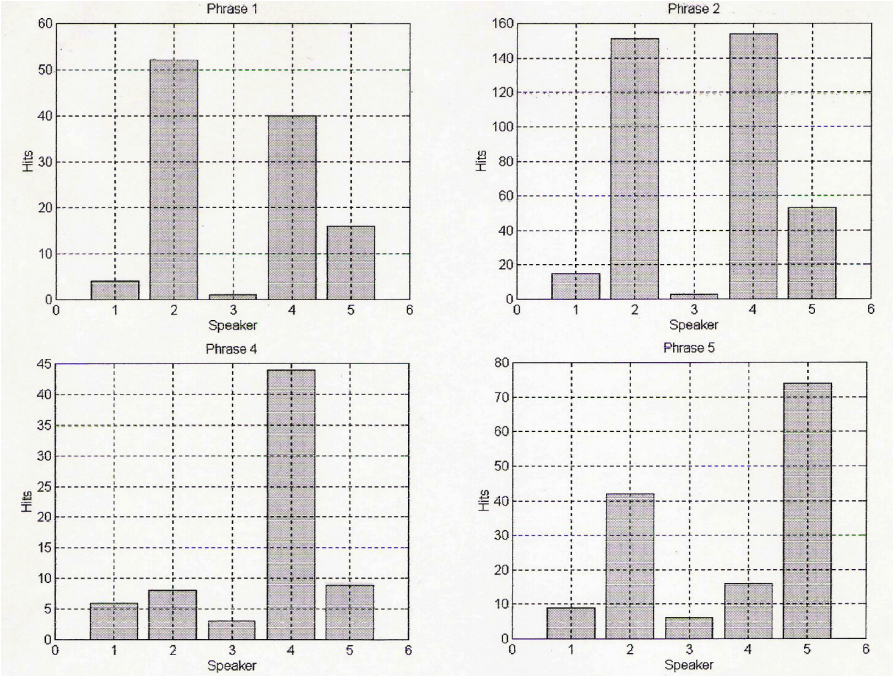


Figure 6 - Four phrases presented to the modular network for analysis. Phrase 2 was spoken by speaker 2, but there was a tie result between speaker 2 and 4.

Taxa de acerto de locutores

TABLE II A - SPEAKER TEST SET (MODULAR TOPOLOGY)

Speaker	Total segments	Correct	Not recognized	Wrong classif.
2	750	75.20 %	7.87 %	16.93 %
3	749	91.32 %	3.20 %	5.48 %
4	598	70.23 %	11.71 %	18.06 %
5	752	82.71 %	6.52 %	10.77 %

TABLE II B - CONFUSION TABLE FOR SPEAKERS (MODULAR)

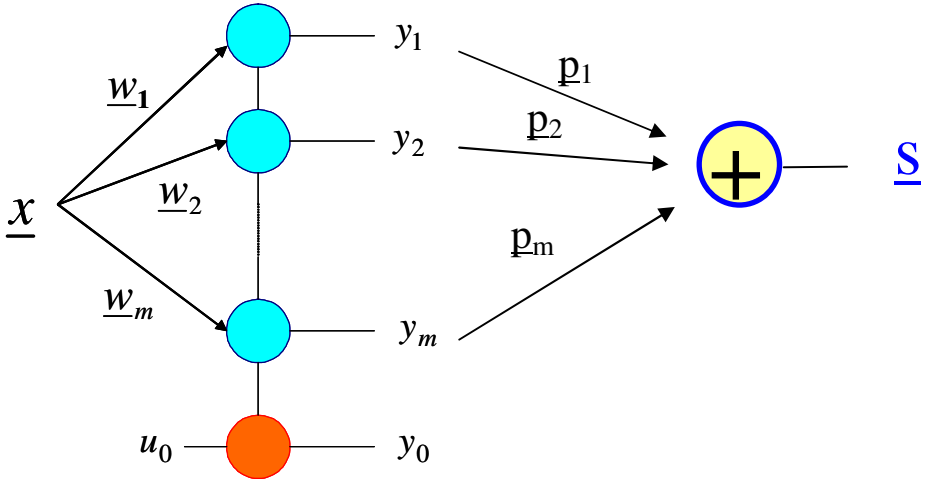
	Recognized Speaker				
Speaker	1	2	3	4	5
2	0.93 %	75.20 %	7.33 %	7.20 %	1.47 %
3	0.00 %	2.80 %	91.32 %	0.80 %	1.87 %
4	1.17 %	13.88 %	1.00 %	70.23 %	2.01 %
5	0.00 %	2.79 %	6.25 %	1.73 %	82.71 %

Os autores do trabalho:

Comparação BP vs ART

Velocidade de treinamento		**ART / BP**
Mínimos locais		**ART**
Velocidade de operação		**ART / BP**
Plasticidade		**ART**
Precisão		
	Classes esféricas	**ART**
	Caso geral	**BP**

Counterpropagation



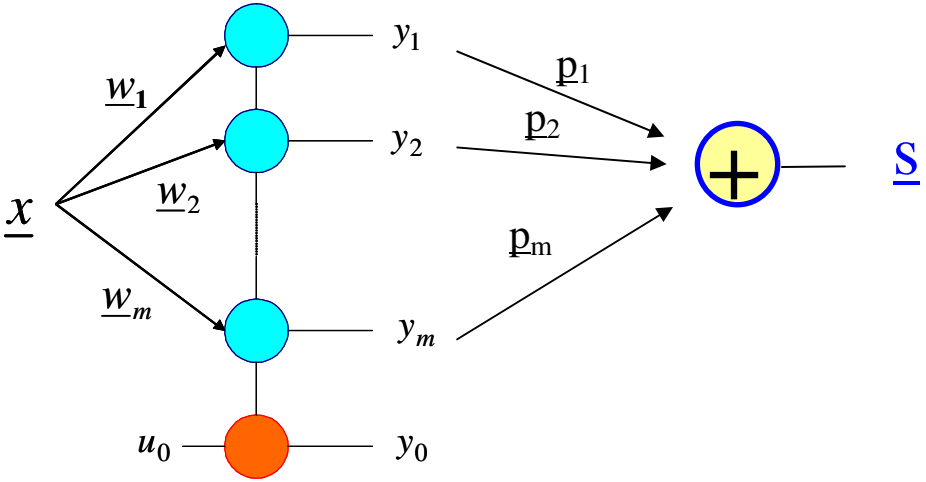
$$\underline{s} = \sum_{j=1}^{m} y_j \underline{p}_j$$

$$\underline{s} = \underline{p}_i \qquad (y_i = 1)$$

Filtragem não linear

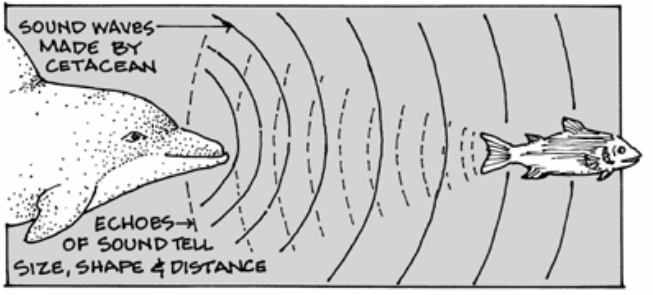
$\underline{p}_i = \underline{w}_i$

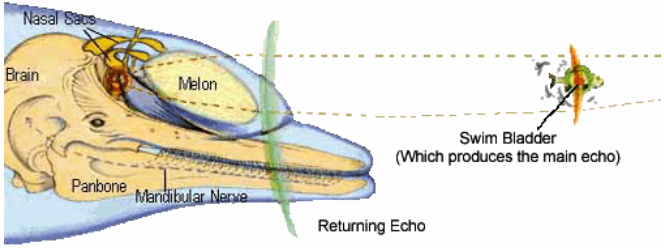
$\underline{x} = \underline{w}_i + \underline{r}$ $\qquad$ $\underline{s} = \underline{w}_i$



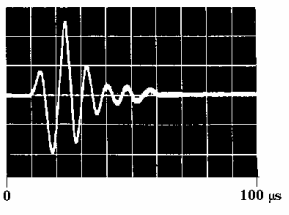
Detecção de padrões desconhecidos

Ex 1: Dolphin Ecolocation





Time to frequency patterns:



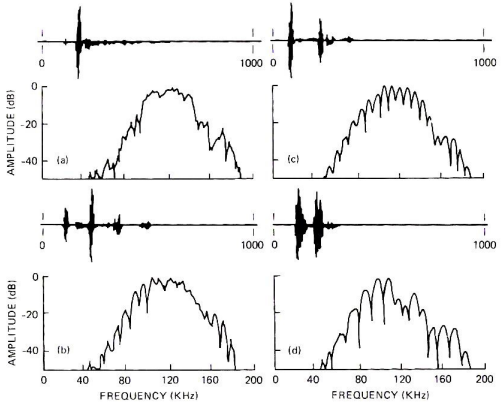
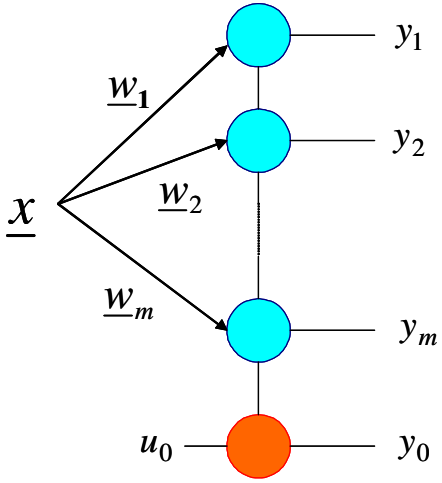
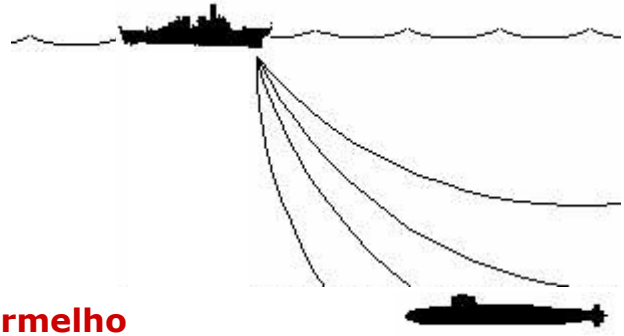


Figure 10-11. Amplitude display and Fourier transform for each item in the dolphin echolocation experiment. (From H. L. Roitblatt, et al. Dolphin Echolocation: Identification of returning echoes using a counterpropagation network. *Proc. IJCNN.* © 1989 IEEE.)



Ex 2: Sonar Passivo

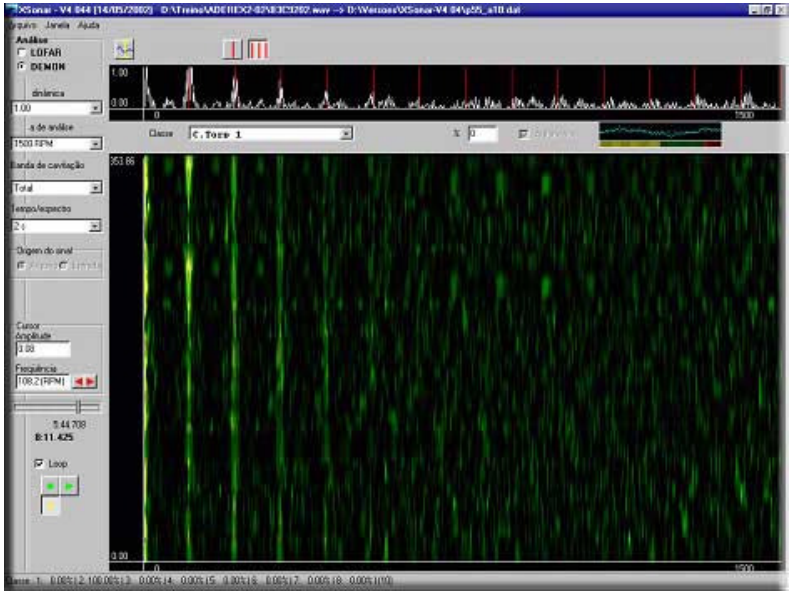


A Caçada Ao Outubro Vermelho

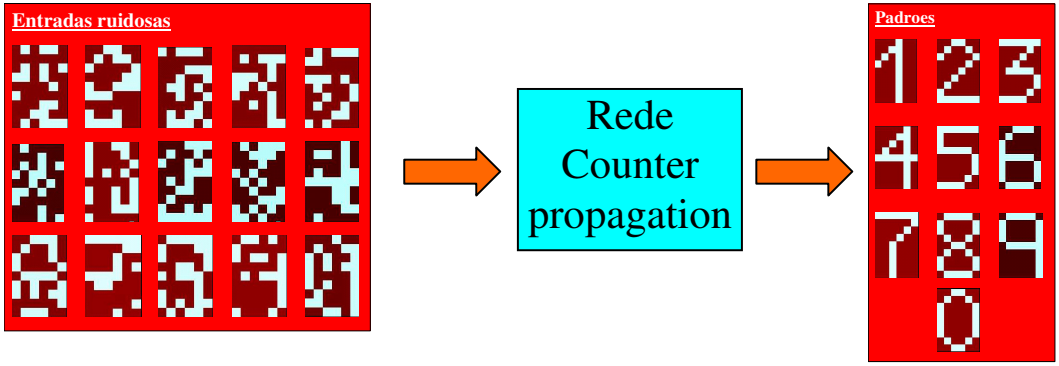
(The Hunt For Red October)

Classificador de Contatos Sonar (IPqM / UFRJ):

Classificação de alvos via análises Lowfar e Demon do ruído irradiado utilizando-se **reconhecimento de padrões através de redes neurais**.

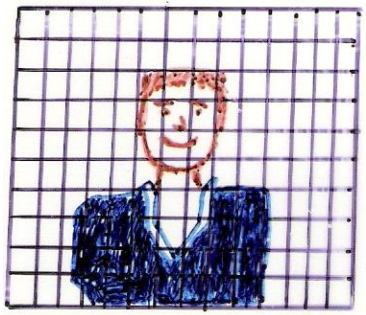


Ex 3: OCR: Figuras com ruído



Compressão de informações

Compressão de imagens



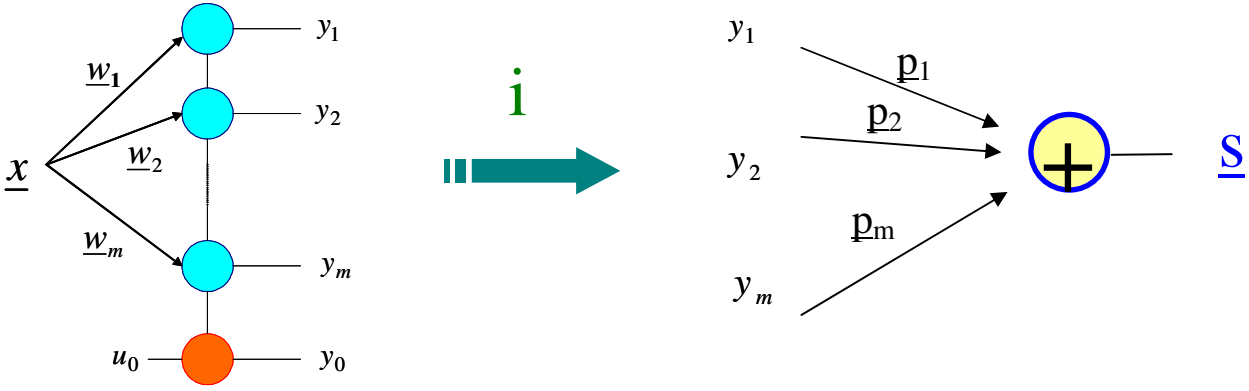


Imagem P&B

BMP
1 quadro 4x4 pix = 16 pix
16 pix/quadro 8 bits/pix = 128 bits/quadro

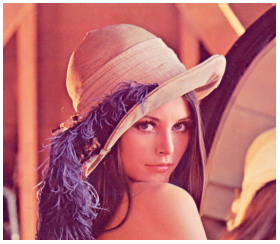
Compressao RN
1 quadro 4x4 pix
8 padroes/quadro 3 bits/padrao = 24 bits/quadro

Taxa de compressao: 128/24 = 5.3

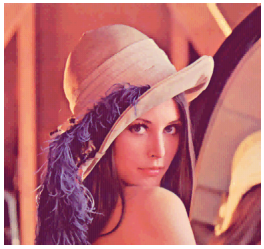
Exemplo de Compressao

Imagens original e comprimidas

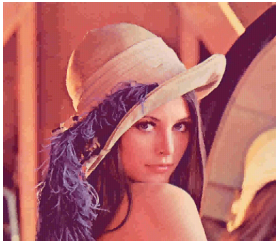
(1x)



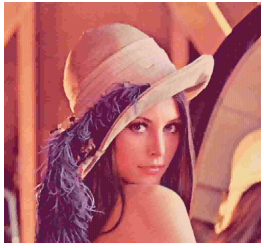
(11x)



(15x)



(28x)



Efeito do codebook (compressão aprox 20x)

codebook Lena

codebook house

Parte III -

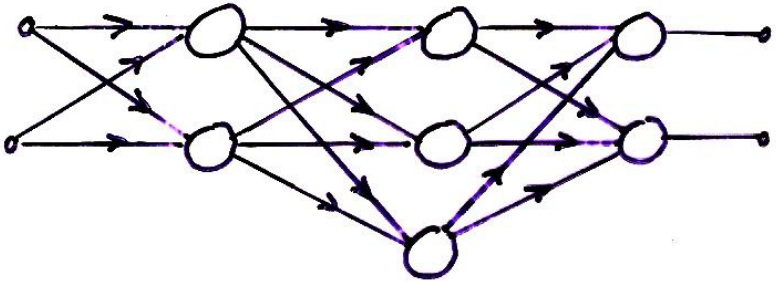
Redes de Hopfield

Redes realimentadas

Problemas de otimização combinatória

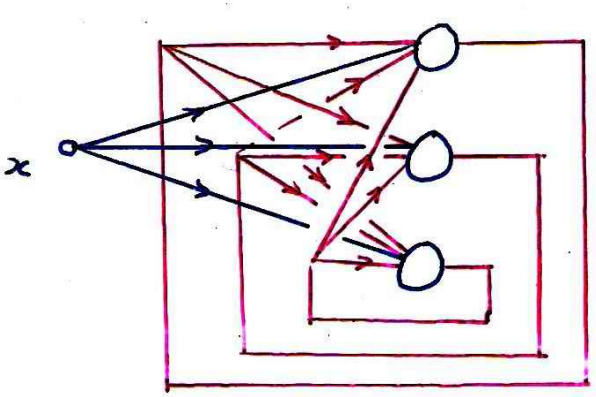
Redes diretas e redes realimentadas

Redes Diretas ou Feedforward



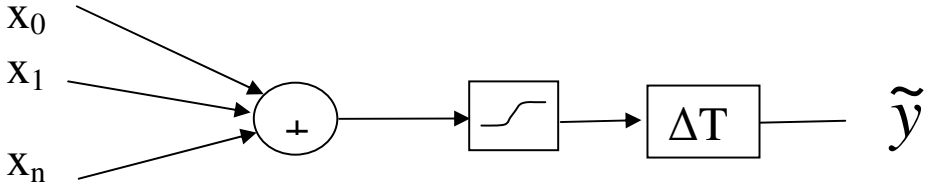
Estável

Redes Realimentadas

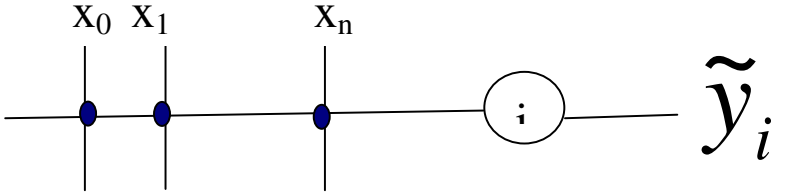


Estável ??

Modelo do neurônio para redes realimentadas

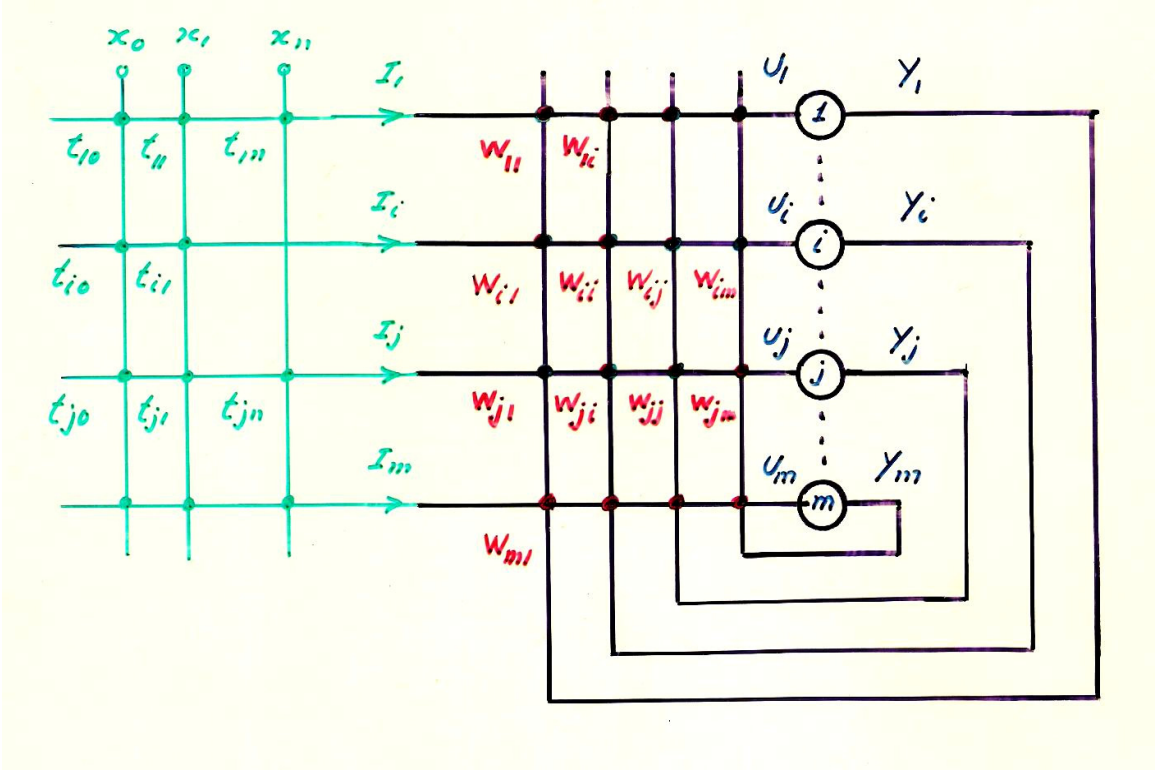


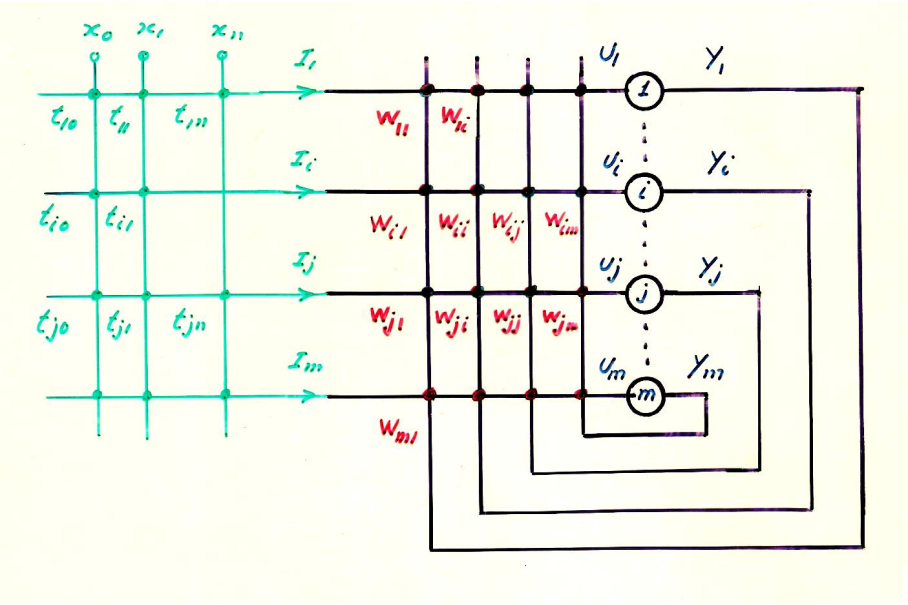
$$\tilde{y}(t) = \varphi\left[\sum_{j=0}^{n} w_{ij}\, x_j(t-1)\right]$$



Atenção: Nova notação !

Rede de Hopfield





$$\underline{I} = \underline{T}\ \underline{x} \qquad \underline{x} = ctte(t)$$

$$\underline{u}(t) = \underline{W}\ \underline{y}(t) + \underline{I}$$

$$\underline{y}(t) = \varphi\left[\underline{u}(t-1)\right]$$

Estabilidade de Sistemas Dinâmicos não Lineares

Função de Lyapunov - E(t)

$$E(t) = E\left[(\underline{y}(t) \right]$$
$$E(t) \geq Min > -\infty \qquad \forall t$$
$$E(t+1) \leq E(t) \qquad \forall t$$

Se E(t) existe então a rede e estável e converge para

$$y = y_{min} = \text{minimante de } E(\underline{y})$$

Para a rede de Hopfield

$$E(\underline{y}) = -\left[\frac{1}{2}\,\underline{y}^{t}\,\underline{\underline{W}}\,\underline{y} + \underline{y}^{t}\,\underline{I}\right] + c$$

é função de Lyapunov.

e daí ???

Problemas de otimização combinatória:

Dada uma função objetivo $F(\underline{y}), \quad \underline{y} \in \{0,1\}^m$

Deseja-se encontrar $\underline{y}_0$ = Minimante F($\underline{y}$)

Rede de Hopfield

Dada uma função $E(\underline{y}), \quad \underline{y} \in \{0,1\}^m$

A rede evolui ate que $\underline{y} = \underline{y}_{min}$ = Minimante E($\underline{y}$)

Se, por construção, $E(\underline{y}) = F(\underline{y})$

A rede de Hopfield resolveu o problema

de otimização combinatória !!!

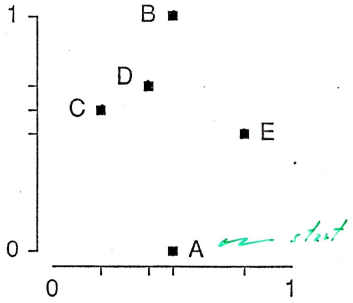
Problemas:

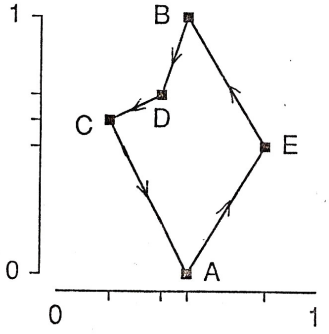
Formulação F complexa

Formulação E = F complexa.

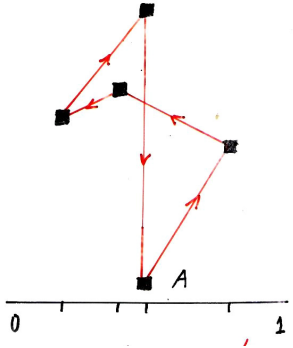
Mínimos locais >> recozimento simulado.

Caixeiro Viajante:



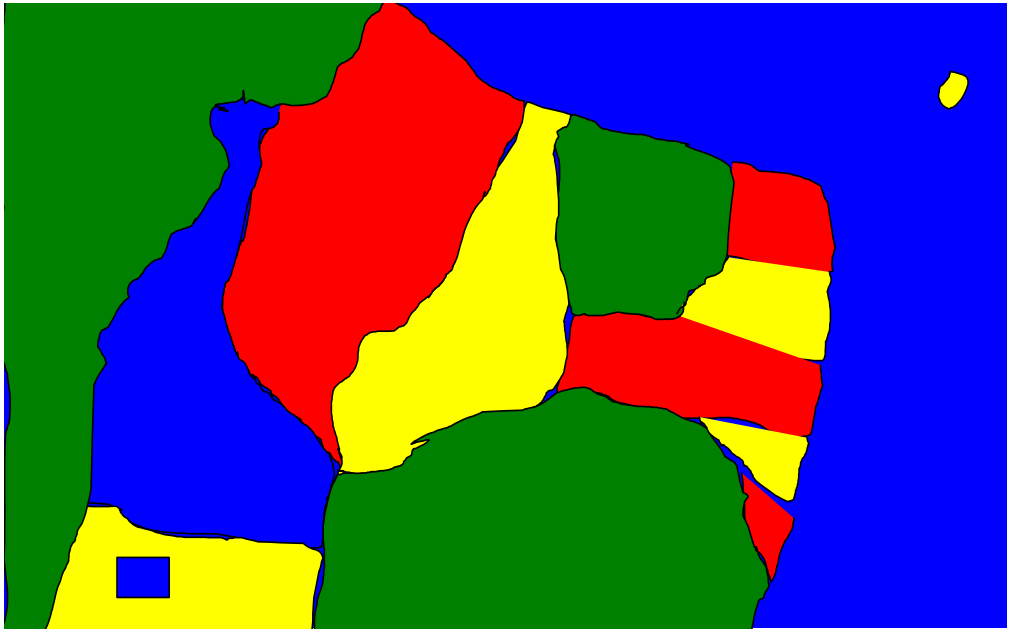


Solução ótima

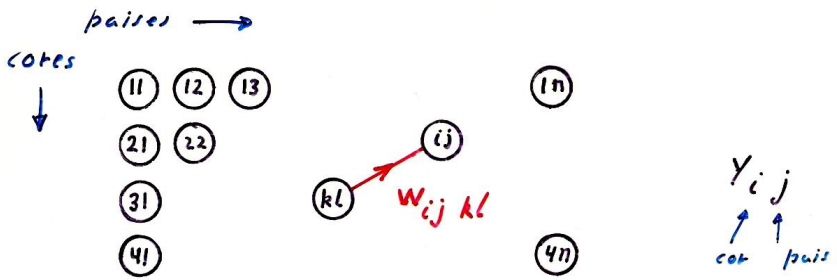


Vizinho mais próximo

O problema das Quatro cores:



A rede neural



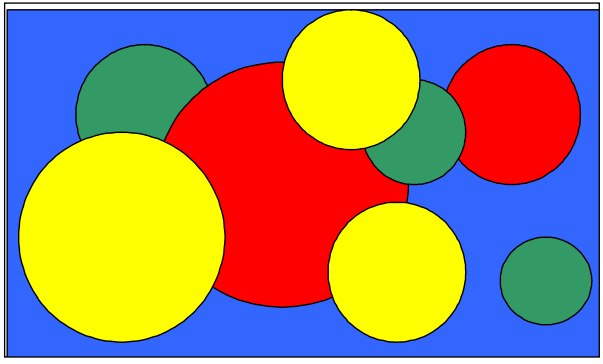
Função objetivo do problema:

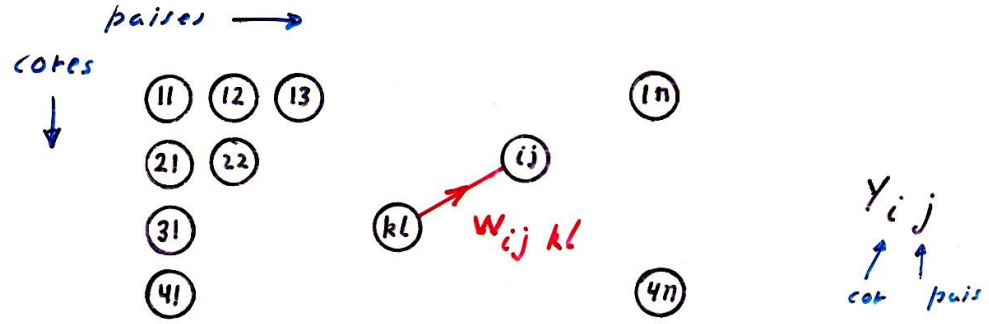
$$F(\underline{y},\underline{w}) = a\sum_{i,j} y_{ij}(1-y_{ij}) + b\sum_{j}(1-\sum_{i} y_{ij})^2 + c\sum_{i}\sum_{j}\sum_{k\neq j} v_{jk}\; y_{ji}\, y_{ki}$$

Função de Lyapunov da rede:

$$E(\underline{y},\underline{w}) = -\left[\frac{1}{2}\sum_{i}\sum_{j}\sum_{kl\neq ij} w_{ijkl}\; y_{ij}\; y_{kl} + \sum_{i}\sum_{j} w_{ijij}\; y_{ij}^2 + \sum_{i}\sum_{j} I_{ij}\, y_{ij}\right] + c$$

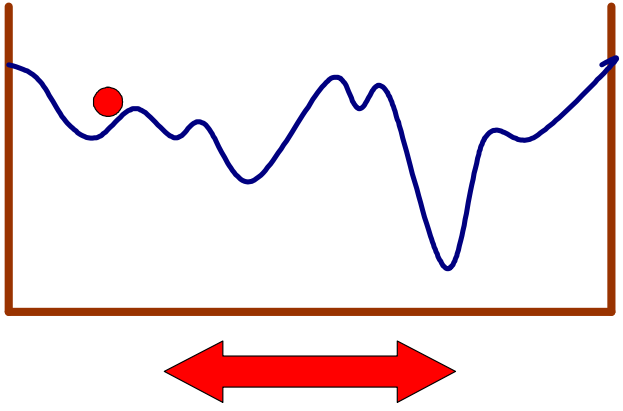
Demo 4 cores:



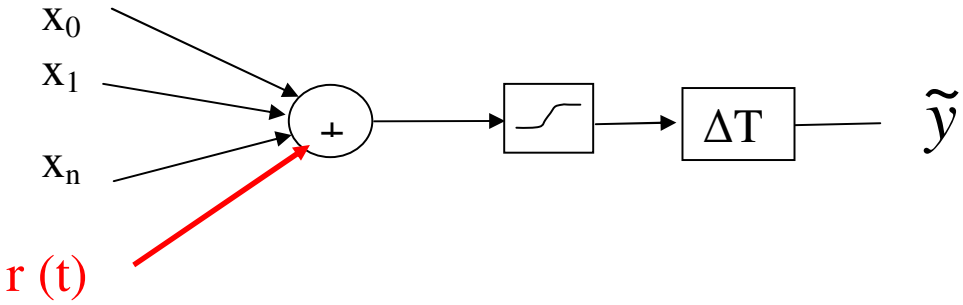


E os mínimos locais ??

Simulated Annealing, Recozimento Simulado



Simulated Annealing na Rede de Hopfield:



$$\tilde{y}(t) = \varphi\left[\sum_{j=0}^{n}\left[\ w_{ij}\, x_j(t-1) + r(t)\ \right]\right]$$

r (t) ?

CRYPTANALYSIS OF SPEECH SIGNALS CIPHERED BY TSP USING ANNEALED HOPFIELD NEURAL NETWORK AND GENETIC ALGORITHMS

J. A. Apolinário Jr.[1,3] apolin@coe.ufrj.br
P. R. S. Mendonça.[1] mendonca@coe.ufrj.br
R. O. Chaves.[1] chaves@coe.ufrj.br
L. P. Calôba.[1,2] caloba@coe.ufrj.br

Time Segment Permutation

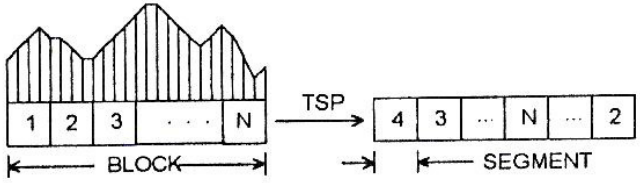


Fig. 1 - Speech signal divided in blocks and segments.

Segment Matching Criterium

$$cd(R,L) = \left[c_R(0) - c_L(0)\right]^2 + 2\sum_{i=1}^{P}\left[c_R(i) - c_L(i)\right]^2 \quad (1)$$

Neurons arrangement and Objective Function

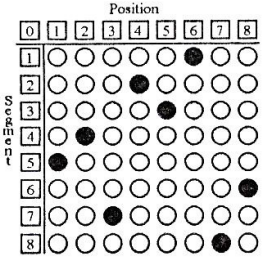


Fig. 4. Each circle corresponds to a neuron and the filled circles show the activated neurons. The permutation of the above configuration is 5-4-7-2-3-1-8-6.

$$F_A = \sum_{ij} A_{ij} v_{ij} \left(1 - v_{ij}\right),$$

$$F_B = \sum_i B_i \left(1 - \sum_j v_{ij}\right)^2,$$

$$F_C = \sum_j C_j \left(1 - \sum_i v_{ij}\right)^2,$$

$$F_D = \sum_{\substack{ijk \\ i \neq k}} D_{ijk} v_{ij} \left(d_{ki} v_{k,j-1} + d_{ik} v_{k,j+1}\right) \text{ e}$$

$$F_E = \frac{1}{2} \sum_i E_i d_{0i} v_{i1} .$$

Lyapunov function and synapses values

$$E = -\frac{1}{2} \sum_{\substack{ijkl \\ k \neq i \\ l \neq j}} w_{ijkl} v_{ij} v_{kl} - \frac{1}{2} \sum_{\substack{ijk \\ k \neq i}} w_{ijkj} v_{ij} v_{kj} - \frac{1}{2} \sum_{\substack{ijl \\ l \neq j}} w_{ijil} v_{ij} v_{il} - \sum_{\substack{ij \\ k \neq i \\ l \neq j}} w_{ijij} v_{ij}^2 - \sum_{ij} t_{ij00} v_{ij} + ct$$

$$t_{ij00} = \begin{cases} -A + 2(B + C), & \text{if } j \neq 1; \\ -A + 2(B + C) - (1/2)\mathrm{Dd}_{0i}, & \text{if } j = 1; \end{cases}$$

$$w_{ijij} = A - B - C;$$

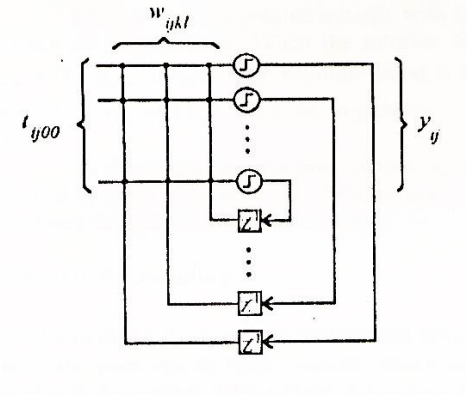
$$w_{ijil} = -2B;$$

$$w_{ijkj} = -2C;$$

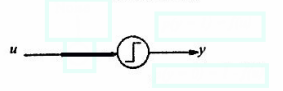
$$w_{ijkl} = \begin{cases} Dd_{ki}, & \text{if } j = l + 1; \\ Dd_{ik}, & \text{if } j = l - 1; \\ 0, & \text{otherwise.} \end{cases}$$

Hopfield NN

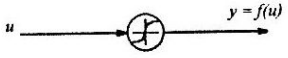
and its neurons



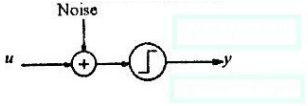
Discret NN



Continuous NN,
Mean fiend annealing



Discrete NN,
Simulated annealing



Resultados:

Percentual de acertos

Apresentação dos Sons

Exemplos	School	Vega
Tipo de processo	% acertos	
Som Criptografado	0	0
Vizinho + Proximo	52	20
Mean Field Ann.	56	57
Simulated Ann. e Busca exaustiva	73	91
Som Original	100	100

Parte IV -

Conclusão

Últimos Comentários

Redes Neurais

não são a panacéia universal !!

Quando usar ?

Existe um algoritmo (modelo fenomenológico) satisfatório ?

SIM - então use o algoritmo

NÃO - então pense em usar redes neurais

O que usar ?

1 - Qual o tipo de problema ?

aproximação >>> use BP

classificação >>> use BP ou ART

otimização combinatória >>>> use Hopfield

2 – Que tipo de treinamento e possível ?

supervisionado é preferível, com BP ou ART

não supervisionado >>> ART

Que cuidados tomar ?

Pré processamento dos dados e fundamental

Aproximadores ou Classificadores ?

Acompanhamento muito cuidadoso do processo

Crítica pré, durante e pos processamento

Otimização combinatória ?

Hopfield

Recozimento simulado e fundamental

Processo muito critico, computação pesada

Como usar ?

Emular em PC (99% dos casos)

Para saber mais:

Referências bibliográficas:

Para iniciar:

* 1 – Ivan Silva, I.; Spatti, D. e Flauzini, R. - "Redes Neurais Artificiais para engenharia e ciências aplicadas", Artliber,2010

2 -Wasserman, P. – “Neural Computing”, Van Nostrand Reinhold, 1989, Cap 1-6.

3 - Sarle, W.S. - ftp://ftp.sas.com/pub/neural/FAQ.html - uma página com muitas informações interessantes.

Para aprofundar:

* 4 - Haykin, S., “Neural Networks and Learning Machines”, Pearson, 2009. Ver também: Haykin, S., “Redes Neurais, Teoria e Prática”, Bookman, 2001.

5 - Hertz, J.; J. Krog, A.; Palmer, R. – “Introduction to the Theory of Neural Computation”, Addison Wesley, 1991.

6 - Cichoki, A.; Unbehauen, R. - "Neural networks for Optimization and Signal Processing", Wiley, 1993.

7 - Bishop, C. M. - "Neural Networks for Pattern Recognition", Oxford, 1999.

Software:

Neuralworks, Neuroshell

* Toolboxes: Matlab, Statistica

Neunet www.cormactech.com/neunet

Freeware:

* Weka www.cs.waikato.ac.nz/ml/weka/

SNNS (Stuttgart Neural Network Simulator) ftp.informatik.uni-stuttgart.de

NeuroLab, SIRENE (UFRJ e CEPEL) www.lps.ufrj.br/~caloba

Sociedades científicas:

International Neural Networks Society, INNS,

IEEE Computational Intelligence Society

Sociedade Brasileira de Inteligência Computacional, SBIC,
(antiga Sociedade Brasileira de Redes Neurais, SBRN), www.sbrn.org.br

Revistas

Neural Networks, INNS

IEEE Trans. on Neural networks, IEEE

Learning and non-linear models, SBRN, www.sbrn.org.br

Congressos:

International Joint Conference on Neural Networks.
anual, da INNS e IEEE-NNS. Próximos: San Jose, CA, Julho de 2011; Brisbane, 2012

Congresso Brasileiro de Redes Neurais,
bi-anual, da SBRN. Próximo: Fortaleza, Novembro de 2011

Simpósio Brasileiro de Redes Neurais,
bi-anual, da SBC. Próximo: 2012

Cursos em RNs no PEE da COPPE:

CPE 721 – Redes Neurais Feedforward
2011-2 – quartas e sextas, 10 às 12 h.

CPE xxx - Tópicos Especiais em Redes Neurais (?)
2011 - 3 - quartas e sextas, 10 às 12 h.

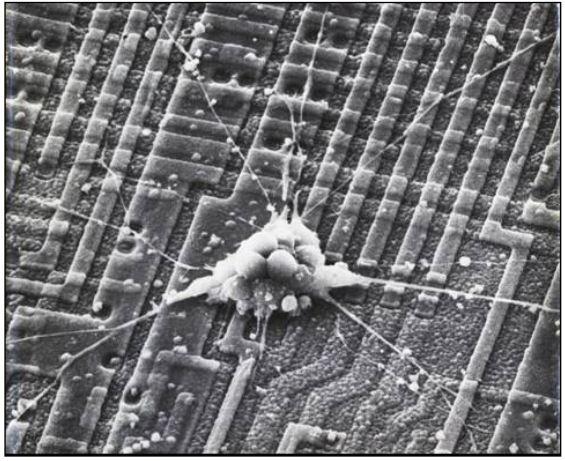
CPE 722 – Redes Neurais Não Supervisionadas e Agrupamento (?)
2010-3 – quartas e sextas, 10 às 12 h.

CPE 723 - Otimização Natural
2012-1 - quartas e sextas

Pré requisitos: apenas noções de álgebra linear

INNS

International Neural Networks Society



Fim.

Obrigado pelo interesse.