

RNA – Redes Neurais Artificiais

Prof. Celso Camilo

Modelos de Neurônios Artificiais

Modelos de Neurônios

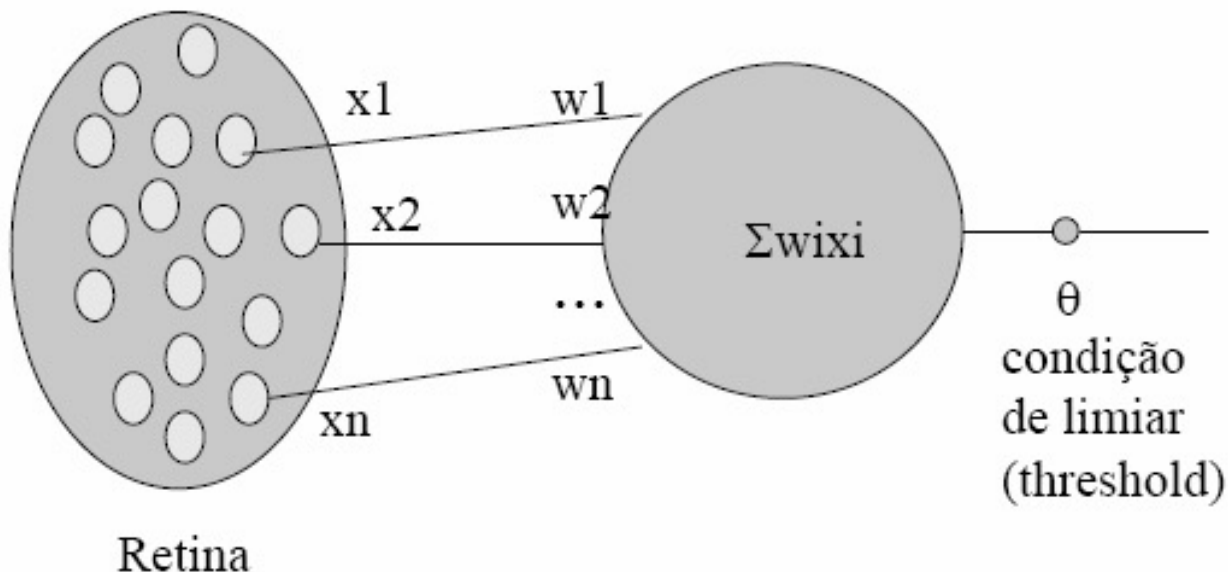
- O Modelo de McCulloch & Pitts (1943)
 - O Cérebro como um Sistema Computacional
 - 5 Suposições Básicas
 - A atividade de um neurônio é um processo tudo ou nada.
 - Um certo número fixo (>1) de entradas devem ser excitadas dentro de um período de adição latente para excitar um neurônio.
 - O único atraso significativo é o atraso sináptico.
 - A atividade de qualquer sinapse inibitória previne absolutamente a excitação do neurônio.
 - A estrutura das interconexões não muda com o tempo.

Modelos de Neurônios

- O PERCEPTRON : Frank Rosenblatt (1958)
 - “A conectividade desenvolvida nas redes biológicas contém um grande número aleatório de elementos”.
 - No início, o Perceptron não é capaz de distinguir padrões e portanto ele é genérico.
 - Pode ser treinado.
 - Com o tempo foi-se notando que a capacidade de separabilidade era dependente de “certas condições de contorno” dos padrões de entrada.

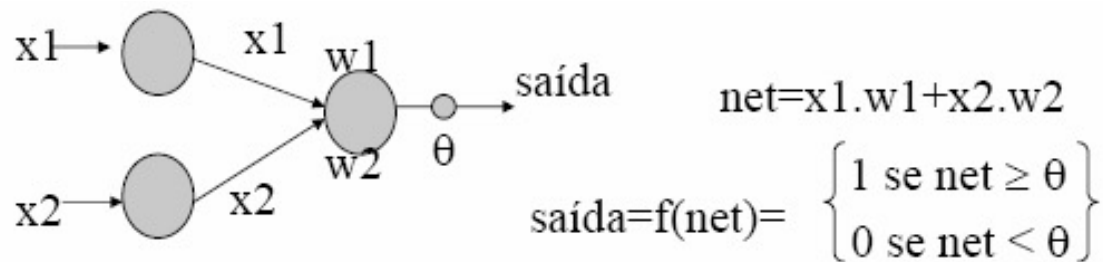
Modelos de Neurônios

- O PERCEPTRON : Frank Rosenblatt (1958)



Modelos de Neurônios

- O PERCEPTRON : Frank Rosenblatt (1958)

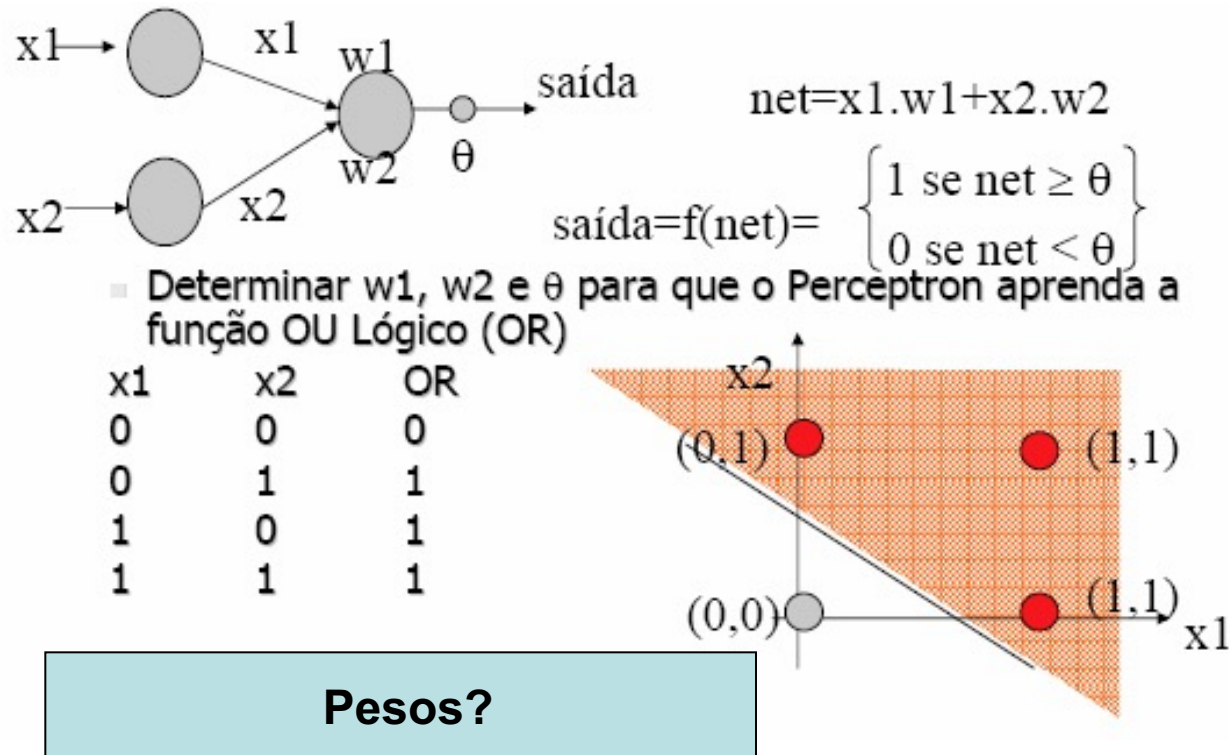


EQUAÇÃO FUNDAMENTAL DO PERCEPTRON

$w_1.x_1 + w_2.x_2 = \theta$ ← EQUAÇÃO DE UMA RETA

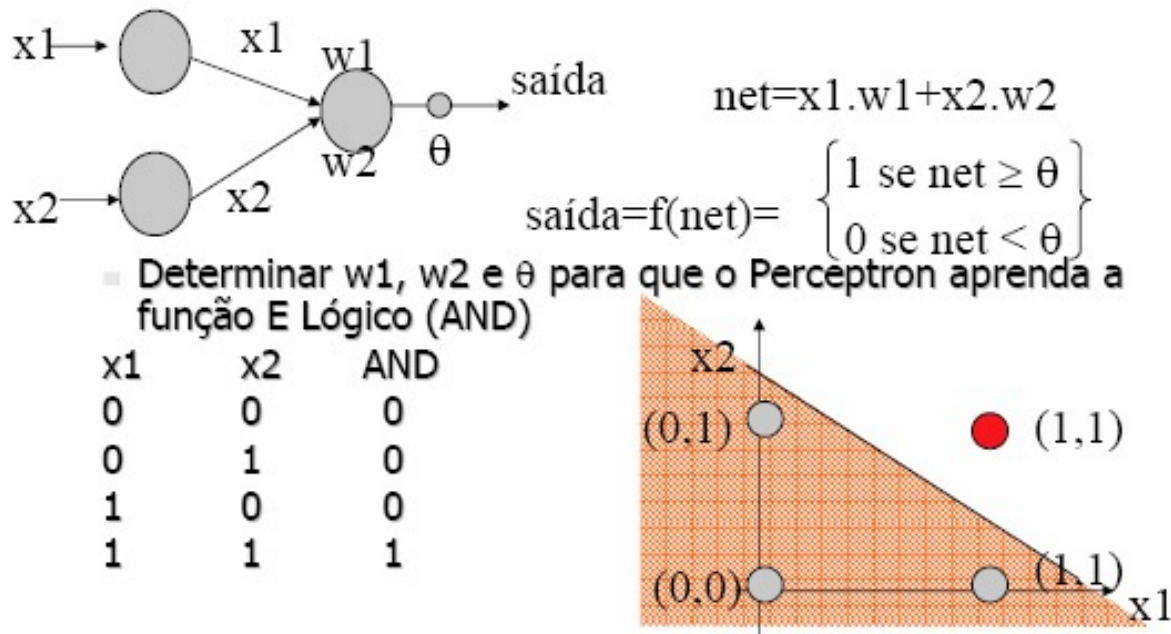
Modelos de Neurônios

- O PERCEPTRON : Exemplo de uso



Modelos de Neurônios

- O PERCEPTRON : Exemplo de uso



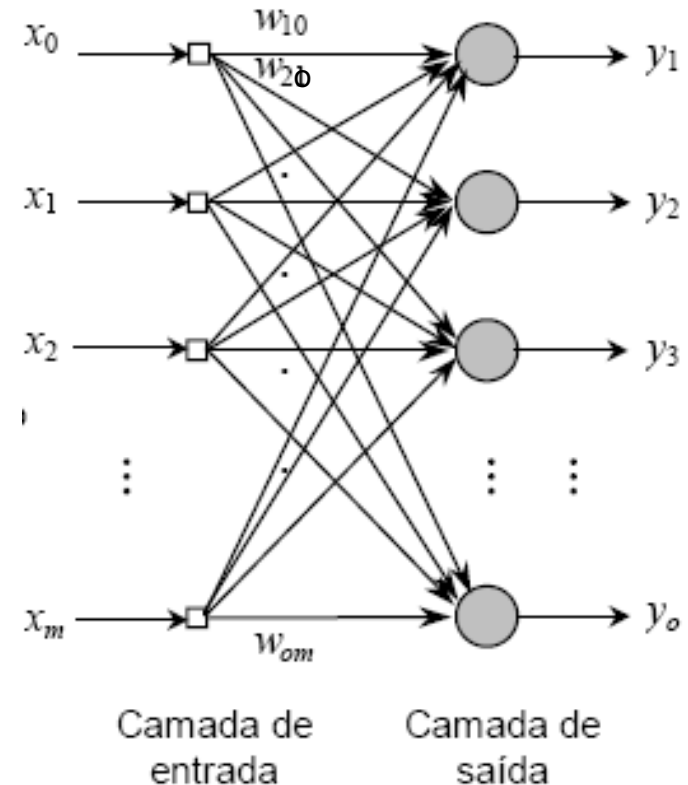
Pesos?

ARQUITETURAS DE REDES NEURAIS ARTIFICIAIS

- 3 tipos básicos de arquiteturas de RNAs:
 - Feedforward de uma única camada
 - Feedforward de múltiplas camadas e
 - Redes recorrentes.

Rede *feedforward* com uma única camada de processamento

- Os neurônios da **camada de entrada** correspondem aos **neurônios sensoriais** que possibilitam a entrada de sinais na rede (não fazem processamento).
- Os neurônios da **camada de saída** fazem **processamento**.

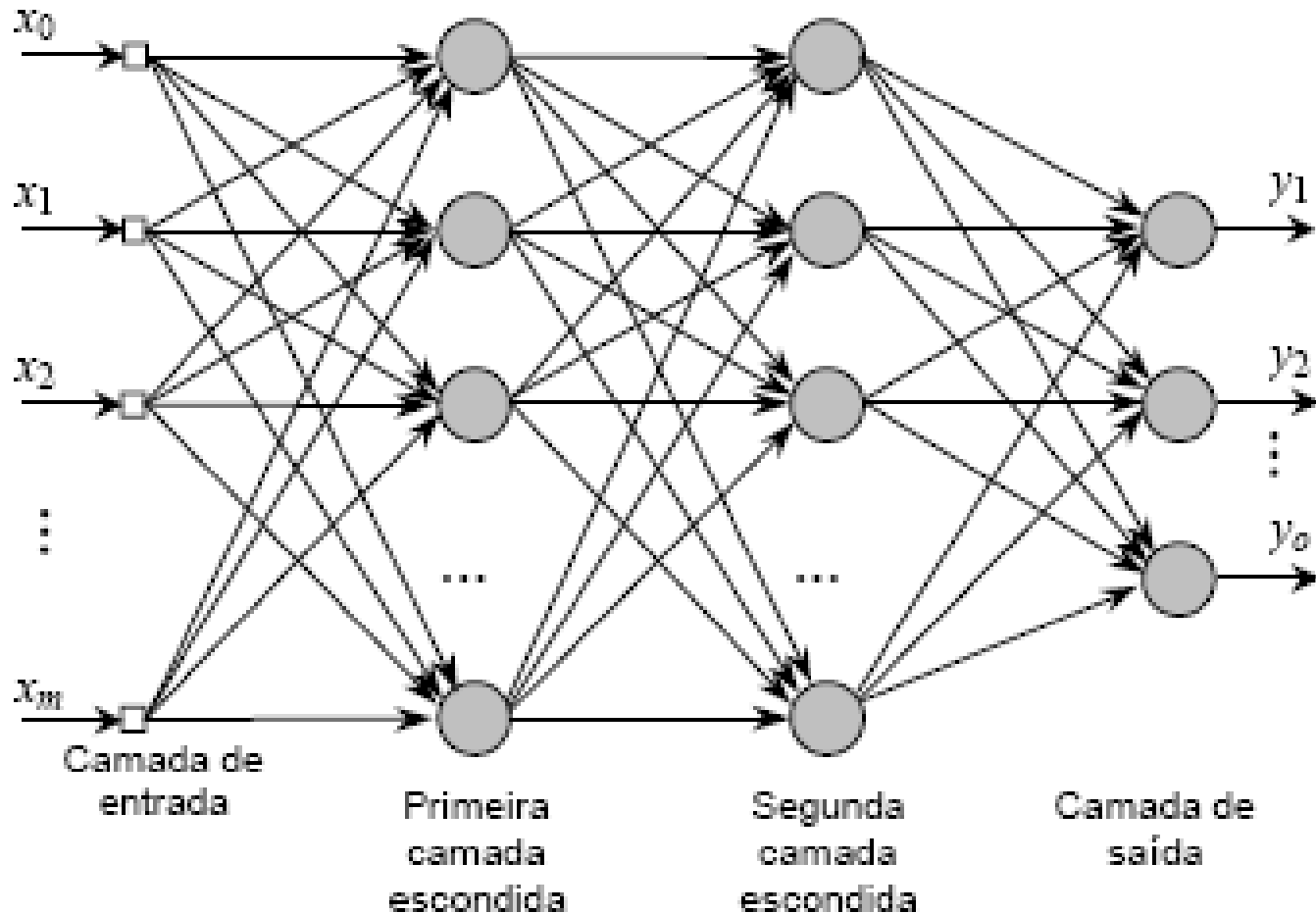


□ neurônio sensorial

● neurônio de processamento

Rede *feedforward* de Múltiplas Camadas

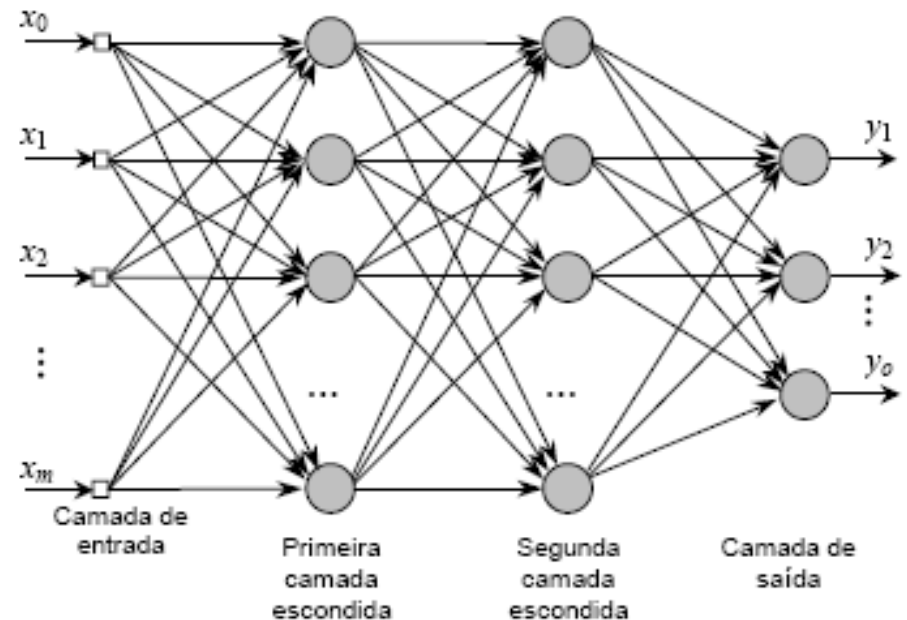
- Essas redes tem uma ou mais **camadas intermediárias ou escondidas**.



Rede *feedforward* de Múltiplas Camadas (cont.)

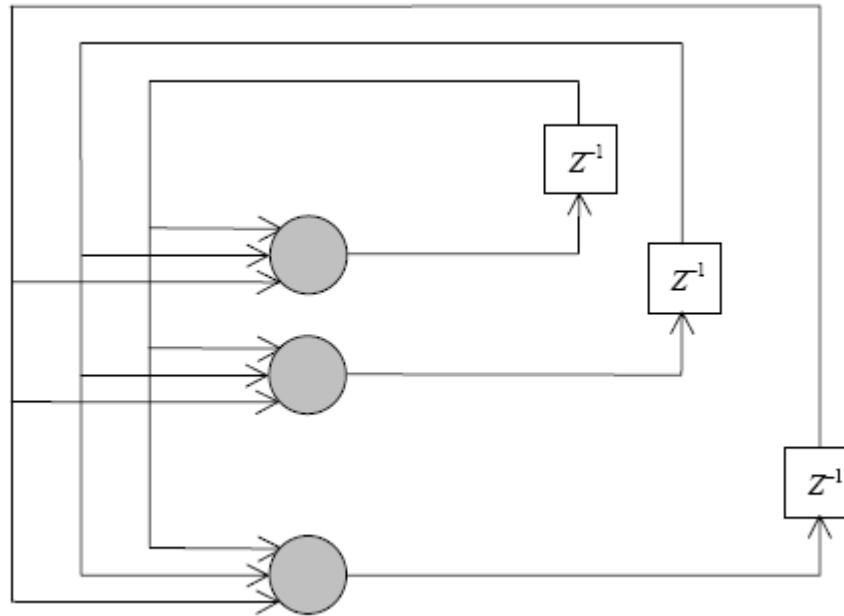
Seja W^k a matriz de pesos da camada k .

w_{ij}^k corresponde ao peso da conexão do neurônio pós-sináptico i ao neurônio pré-sináptico j na camada k .



REDES RECORRENTES (feedback)

Essas redes possuem pelo menos uma **interconexão realimentando a saída de neurônios** para outros neurônios da rede (**conexão cíclica**).



Exemplo: Rede de Hopfield

FASES DE UMA REDE NEURAL ARTIFICIAL

- Uma rede neural artificial pode se encontrar em duas fases:
- a primeira fase é a de **aprendizagem**, ou **treinamento**, em que a rede se encontra no processo de aprendizado, **ajustando os parâmetros livres**, para poder posteriormente desempenhar a função destinada; e
- a segunda fase é a de **aplicação** propriamente dita, na função para a qual ela foi destinada, como de classificação de padrões de vozes, imagens, etc.

Aprendizagem

O processo de aprendizagem implica na seguinte sequência de eventos:

- 1) a rede neural é **estimulada** por um ambiente
- 2) a rede neural sofre **modificações** nos seus parâmetros livres como resultado desta estimulação
- 3) a rede neural **responde de uma maneira nova** ao ambiente, devido as modificações ocorridas na sua estrutura interna

Modelos de Aprendizagem

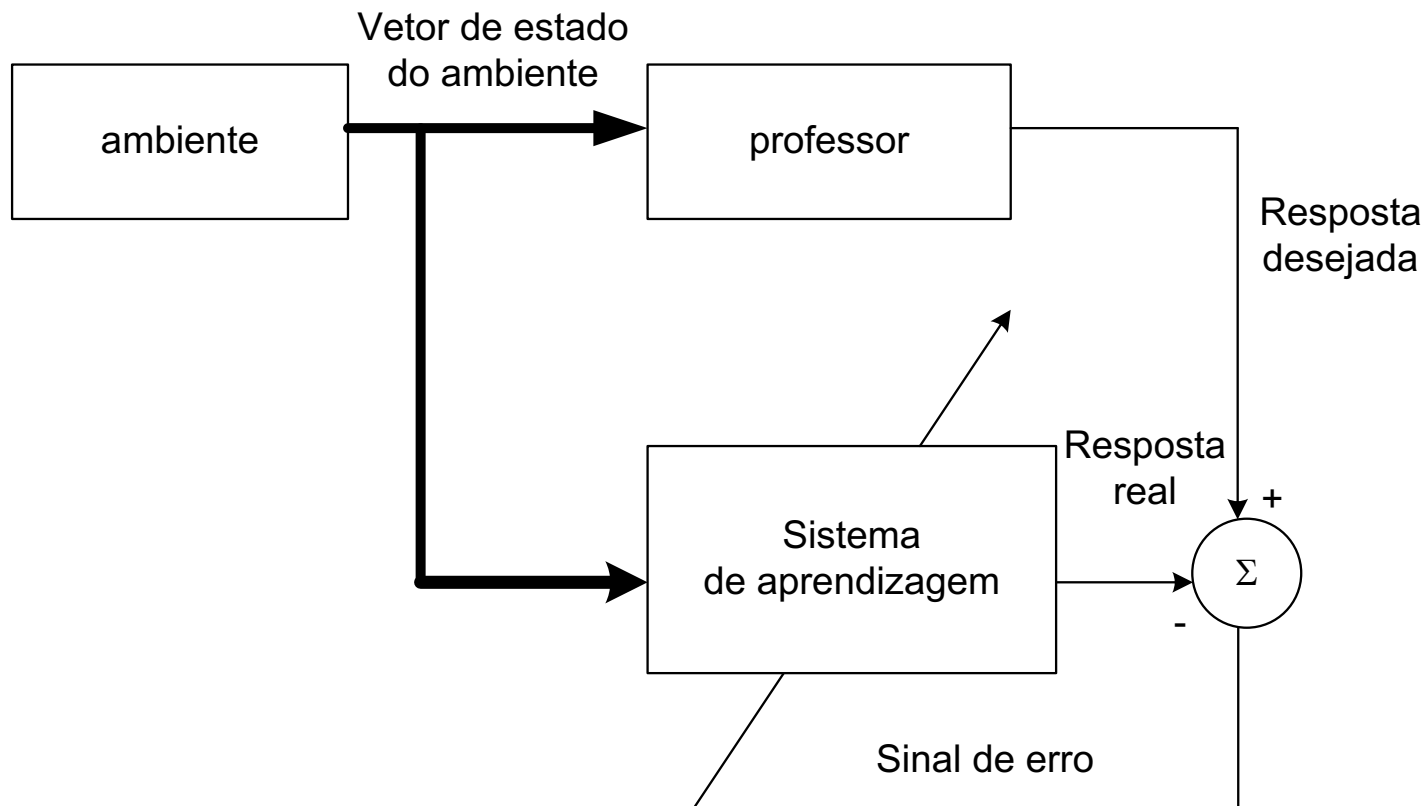
Modelos de Aprendizagem

Os principais modelos (paradigmas) de aprendizagem são:

- 1) supervisionado;
- 2) não-supervisionado; e
- 3) com reforço.

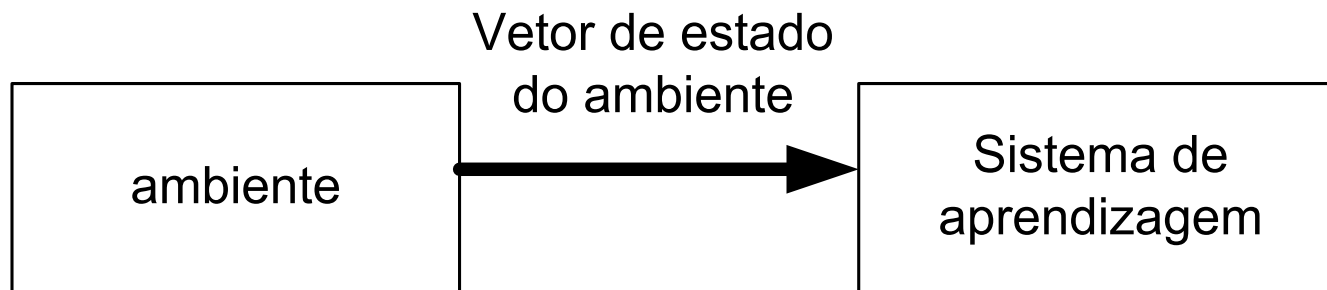
SUPERVISIONADO

- Também conhecida com **aprendizagem com professor**, consiste em que o professor tenha o conhecimento do ambiente, e **fornece o conjunto de exemplos de entrada-resposta desejada**.
- Com esse conjunto, o treinamento é feito usando a **regra de aprendizagem por correção de erro**.



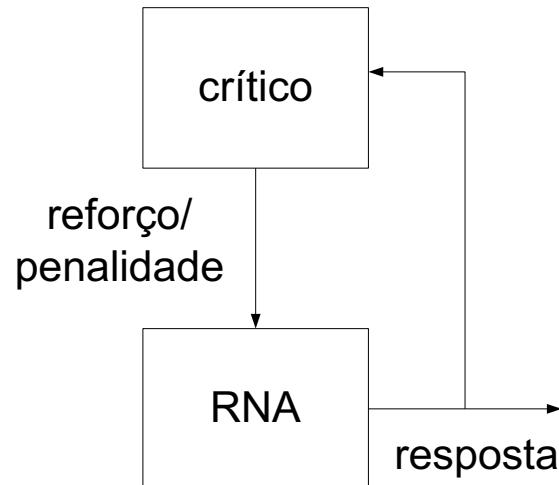
NÃO-SUPERVISIONADO

- Neste caso **nao há um professor** para supervisionar o processo de aprendizagem. Isso significa que **não há exemplos rotulados da função a ser aprendida pela rede**.
- Nesse modelo, também conhecido como **auto-organizado**, são dadas condições para realizar uma medida da representação que a rede deve aprender, e os parâmetros livres da rede são otimizados em relação a essa medida.
- Para a realização da aprendizagem não-supervisionada pode-se utilizar a **regra de aprendizagem competitiva**.



APRENDIZAGEM POR REFORÇO

- Pode ser visto como **caso particular de aprendizagem supervisionada**.
- A principal diferença entre o aprendizado supervisionado e o aprendizado por reforço é a medida de desempenho usada em cada um deles.
- No aprendizado supervisionado, a medida de desempenho é baseada no conjunto de respostas desejadas usando um critério de erro conhecido, enquanto que no aprendizado por reforço **a única informação fornecida à rede é se uma determinada saída está correta ou não**.
- A idéia básica tem origem em estudos experimentais sobre **aprendizado dos animais**. Quanto maior a **satisfação obtida** com uma certa experiência em um animal, maiores as chances dele aprender.



Algoritmos de Treinamento

Regra Hebb

- Idealizada por Hebb, a idéia básica é que se duas unidades são ativadas simultaneamente, suas interconexões tendem a se fortalecer. Se **i** recebe o sinal de saída de **j**, o peso **W_{ij}** é modificado de acordo com :

$$\Delta W_{ij} = \lambda a_i a_j$$

- onde λ (lambda) é uma constante de proporcionalidade representando a taxa de aprendizado e **a_i** e **a_j** são ativações (ou saídas) das unidades **i** e **j** respectivamente. Alguns autores representam alternativamente a matriz **W_{ij}** por **W_{ji}**.

Regra Delta

- É uma variante da Regra de Hebb, introduzida por Widrow-Hoff. A diferença quanto a de Hebb é que possui uma saída desejada d_i . Assim, o peso será proporcional à saída.

Taxa Aprendizagem

Erro

$$w_{IJ} = w_{IJ} + \alpha (d_I - y_I) x_J$$

$$b_I = b_I + \alpha (d_I - y_I)$$

Perceptron - Treinamento

Se o perceptron dispara quando não deve disparar, diminua de cada peso (W_i) um número proporcional ao sinal de entrada;

Se o perceptron deixa de disparar quando deveria, aumente um número semelhante em cada W_i .

Perceptron – Exemplo (I)

Imaginemos um problema de classificação:

	COMPOSITO R	CIENTISTA
BACH	X	
BEETHOVEN	X	
EINSTEIN		X
KEPLER		X

Perceptron – Exemplo (II)

Primeiro, codificamos as informações em base binária:

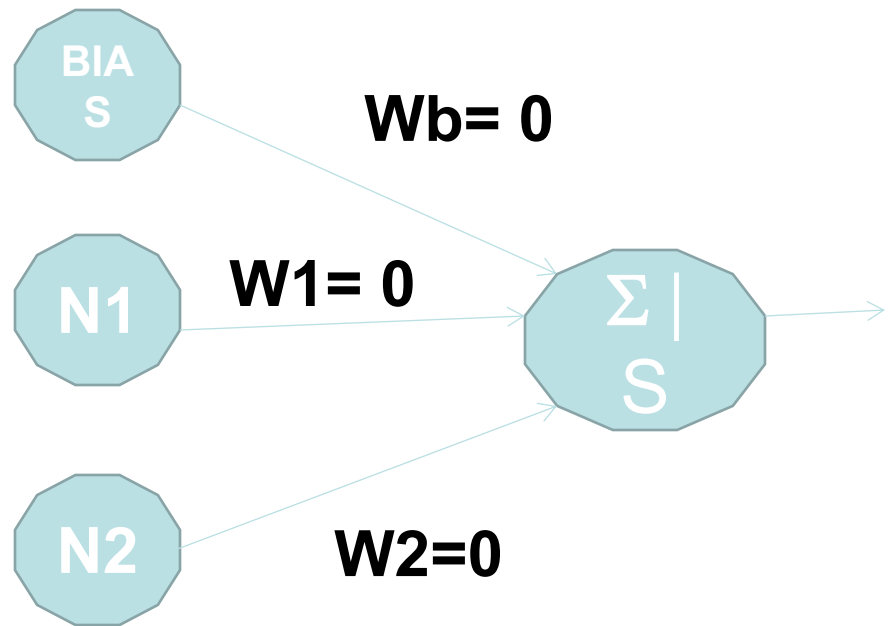
COMPOSITOR	0
CIENTISTA	1

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1

Perceptron – Exemplo (III)

Os pesos começam todos com zero (W_b , W_1 , $W_2 = 0$).
O sinal do bias será sempre positivo (1).

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1

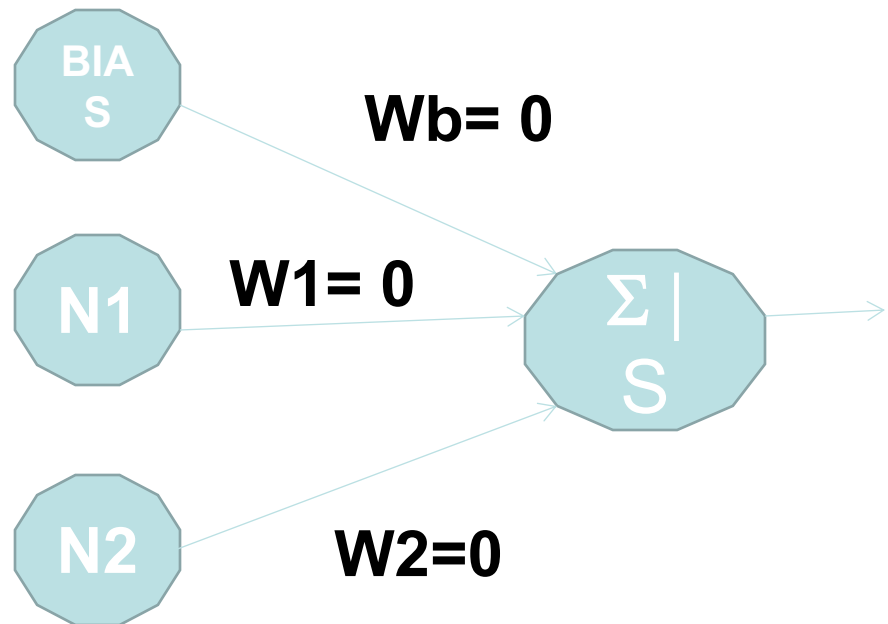


Perceptron – Exemplo (IV)

Escolhemos um valor para testar a rede.

A somatória (Σ) será: $(\text{Bias} * W_b) + (N1 * W1) + (N2 * W2)$

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
→ KEPLER	1	1	1

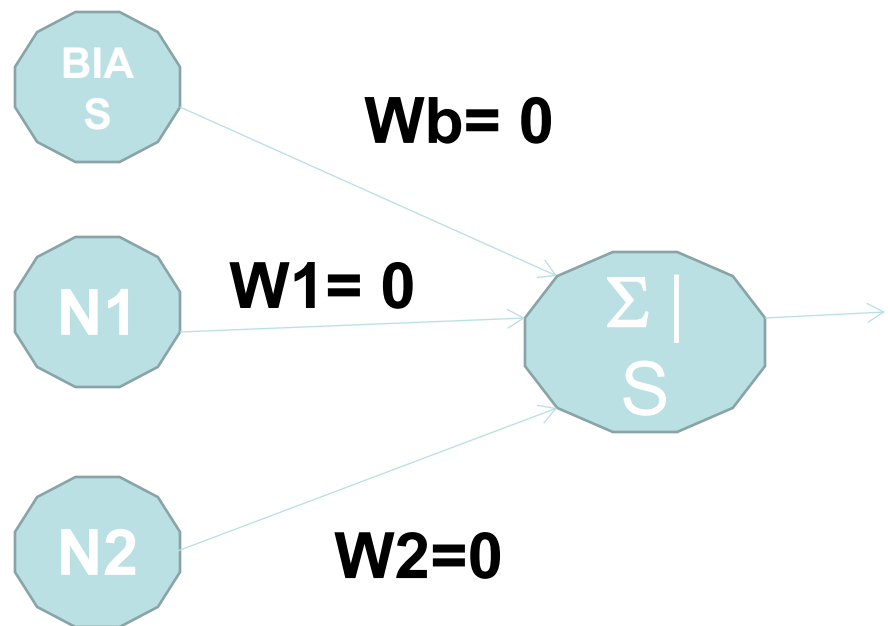


Perceptron – Exemplo (V)

Para Kepler, teremos: $(1 * 0) + (1 * 0) + (1 * 0) = 0$.

Ou seja, a saída foi 0, e a saída esperada era 1... :-/

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
→ KEPLER	1	1	1



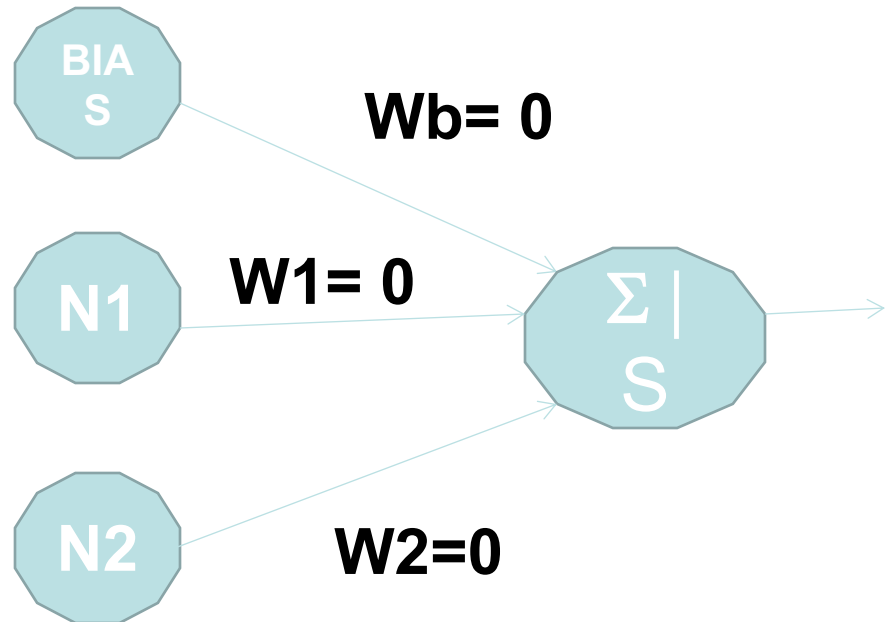
Perceptron – Exemplo (VI)

Temos que ajustar os pesos (W), usando a fórmula:

$W_{\text{novo}} = W_{\text{anterior}} +$

$(\text{ValorErro} * \text{TaxaAprendizagem} * \text{SinalEntrada})$

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



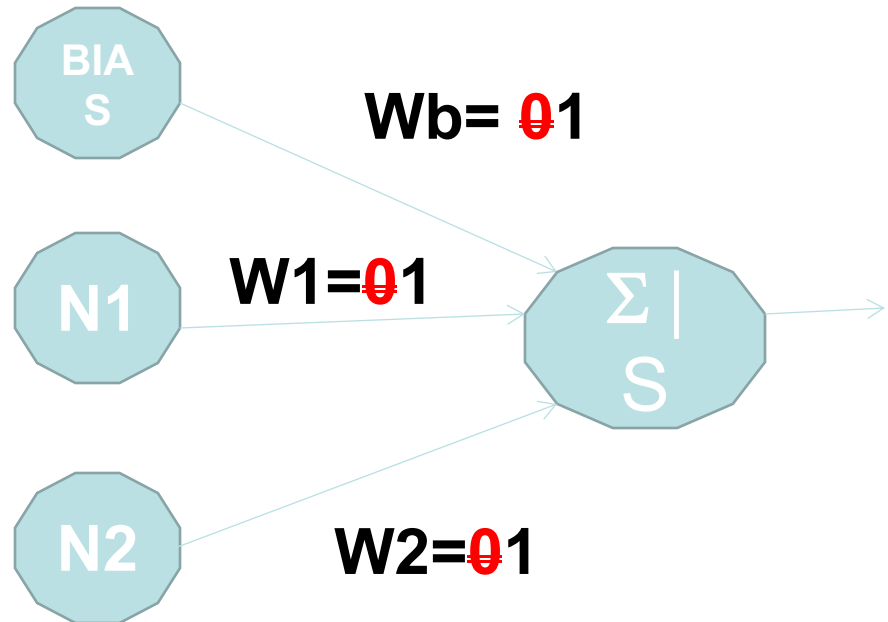
Perceptron – Exemplo (VII)

Para W_b : $W_b \text{ novo} = 0 + (1 * 1 * 1) = 1$

Para W_1 : $W_1 \text{ novo} = 0 + (1 * 1 * 1) = 1$

Para W_2 : $W_2 \text{ novo} = 0 + (1 * 1 * 1) = 1$

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



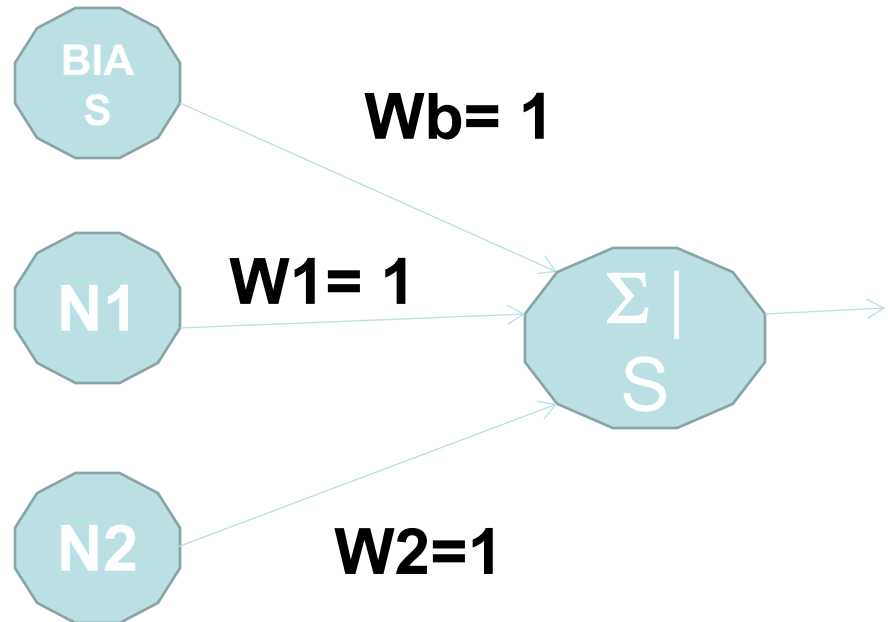
Perceptron – Exemplo (VIII)

Então, escolhemos outro valor e repetimos o processo:

$$(\text{Bias} * W_b) + (N1 * W1) + (N2 * W2) \Rightarrow$$

$$(1 * 1) + (0 * 1) + (1 * 1) = 2 \text{ (se } \Sigma > 0, \text{ sinal saída} = 1)$$

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1

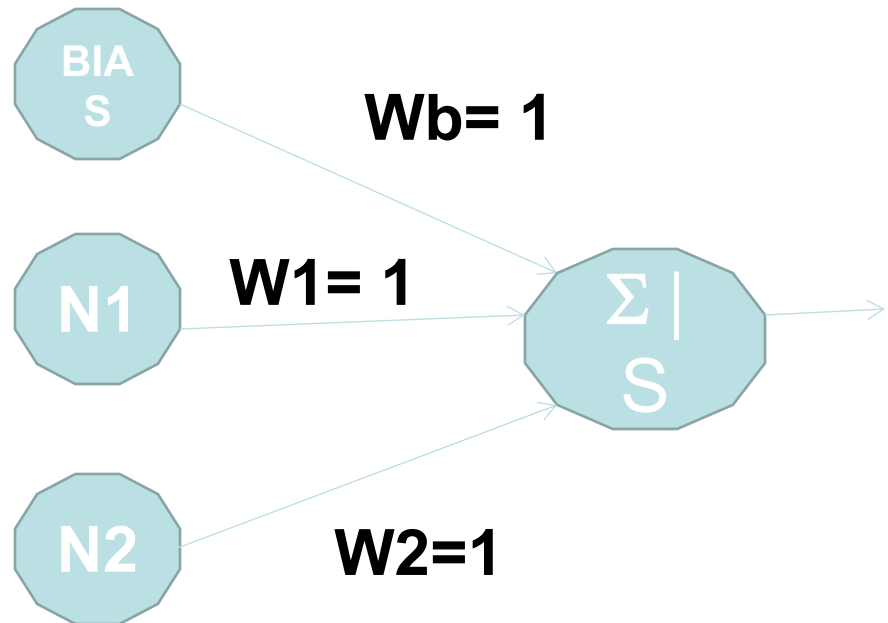


Perceptron – Exemplo (IX)

Novamente o sinal de saída esperado (0) não foi alcançado. Ou seja, é preciso corrigir os pesos.

ValorErro = SinalEsperado – SinalGerado (0 - 1 = -1)

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



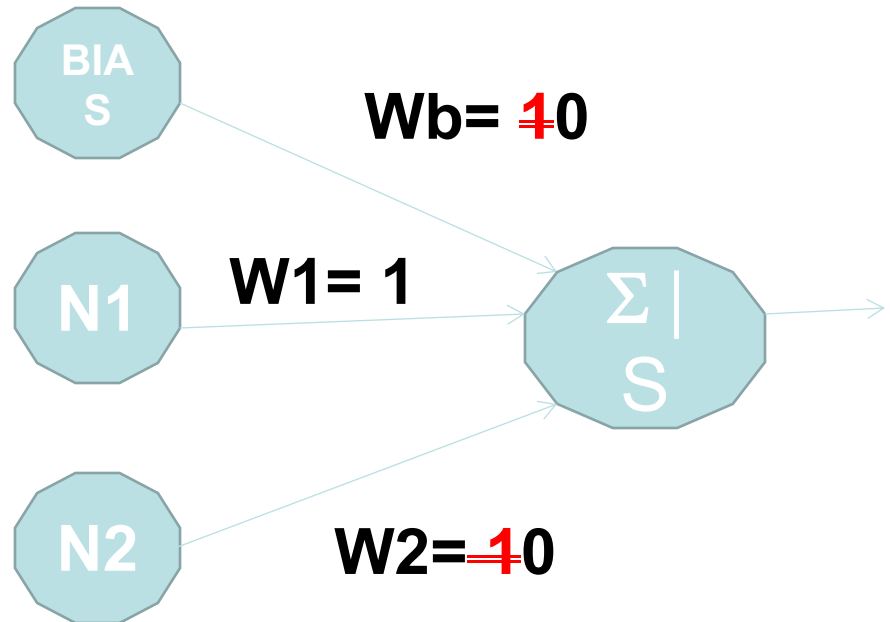
Perceptron – Exemplo (X)

Para W_b : $W_b \text{ novo} = 1 + (-1 * 1 * 1) = 0$

Para W_1 : $W_1 \text{ novo} = 1 + (-1 * 0 * 1) = 1$

Para W_2 : $W_2 \text{ novo} = 1 + (-1 * 1 * 1) = 0$

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



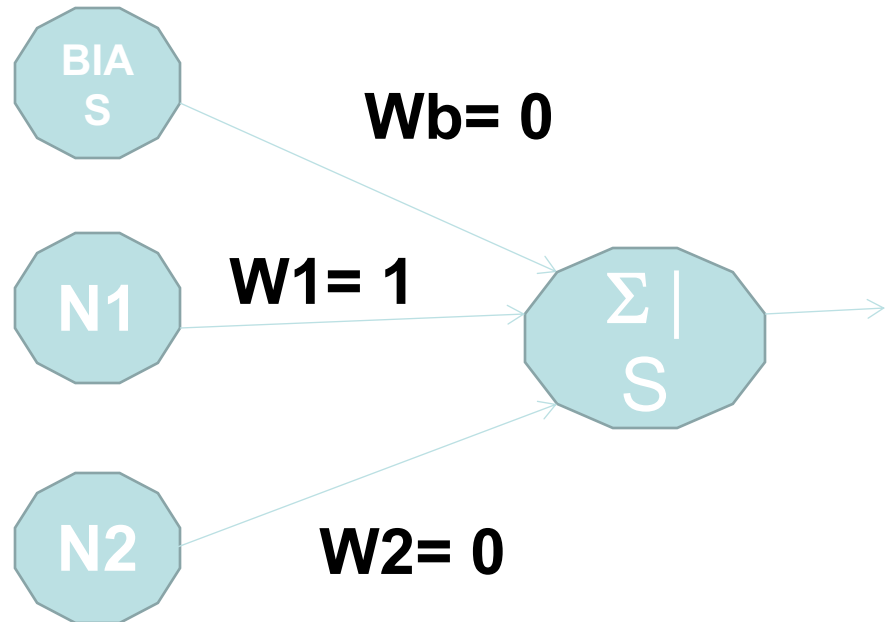
Perceptron – Exemplo (XI)

Seguimos com novo valor:

$$(\text{Bias} * W_b) + (N1 * W1) + (N2 * W2) \Rightarrow$$

$$(1 * 0) + (1 * 1) + (0 * 0) = 1 \text{ (se } \Sigma > 0, \text{ sinal saída} = 1)$$

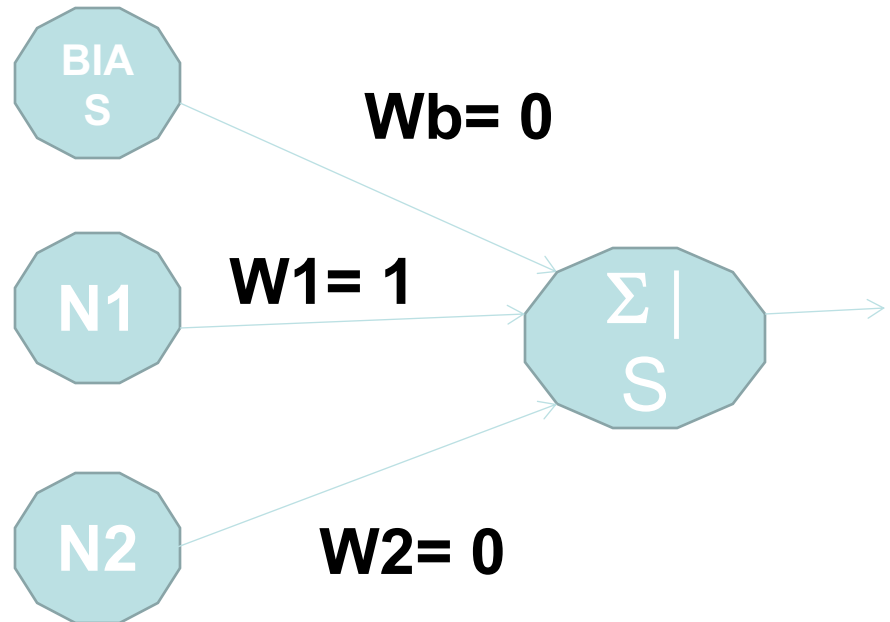
	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



Perceptron – Exemplo (XII)

Desta vez o sinal gerado (1) era o sinal esperado.
Neste caso, não precisamos ajustar os pesos.

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1




Perceptron – Exemplo (XII)

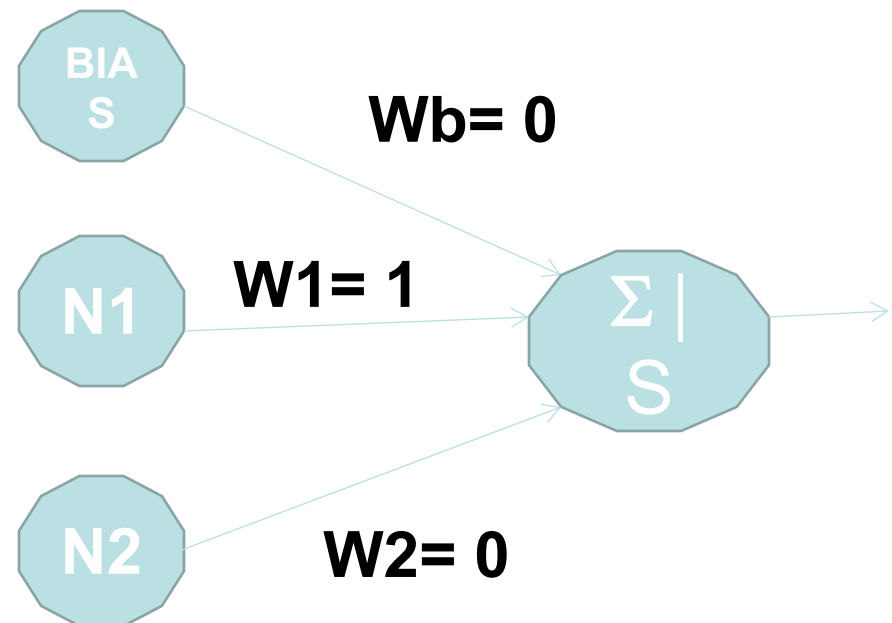
Testando para Bach, o sinal gerado também é correto:

$$(\text{Bias} * W_b) + (N1 * W1) + (N2 * W2) \Rightarrow$$

$$(1 * 0) + (0 * 1) + (0 * 0) = 0$$



	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1

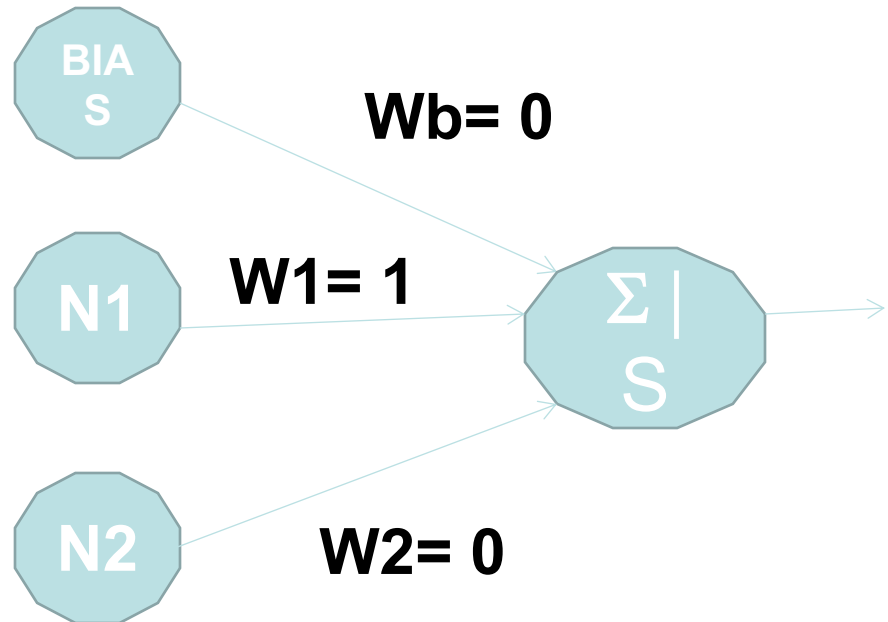


Perceptron – Exemplo (XIII)

Os pesos já funcionaram para 3 entradas, testando Kepler novamente:

$(1*0) + (1*1) + (1*0) = 1 \rightarrow$ Funciona! A rede aprendeu!

	N1	N2	SAÍDA
BACH	0	0	0
BEETHOVEN	0	1	0
EINSTEIN	1	0	1
KEPLER	1	1	1



Perceptrons – Prática 1

- Implemente o experimento anterior.

Questões:

- 1.O programa de treinamento funciona sempre ou depende da seqüência de valores informados durante o treinamento?
- 2.Qual o número máximo de interações para corrigir os pesos?

Perceptrons – Prática 2 (I)

Considere a seguinte base de treinamento, sobre o diagnóstico de Gripe x Resfriado:

	VÍRUS	BACTÉRIA	DOR CABEÇA	CORISA
GRIFE	1	0	1	1
RESFRIADO	0	1	0	1
GRIFE	1	0	1	0
RESFRIADO	0	1	1	1
GRIFE	0	0	1	1
RESFRIADO	0	0	0	1

Perceptrons – Prática 2 (II)

- 1) Quantas iterações são necessárias para encontrar os pesos corretos da base de treinamento?
- 2) Qual é o resultado para outros valores?

	VÍRUS	BACTÉRIA	DOR CABEÇA	CORISA
?	1	1	1	1
?	1	0	0	0
?	0	1	0	0
?	0	1	1	1
?	1	0	0	1
?	0	0	0	0

- 3) O programa parece ter alguma inteligência? Como ele se comporta?

Perceptrons - Limitação

- Minsky e Papert: e se não for linear?
 - XOR

X1	X2	X1 XOR X2
0	0	0
0	1	1
1	0	1
1	1	0

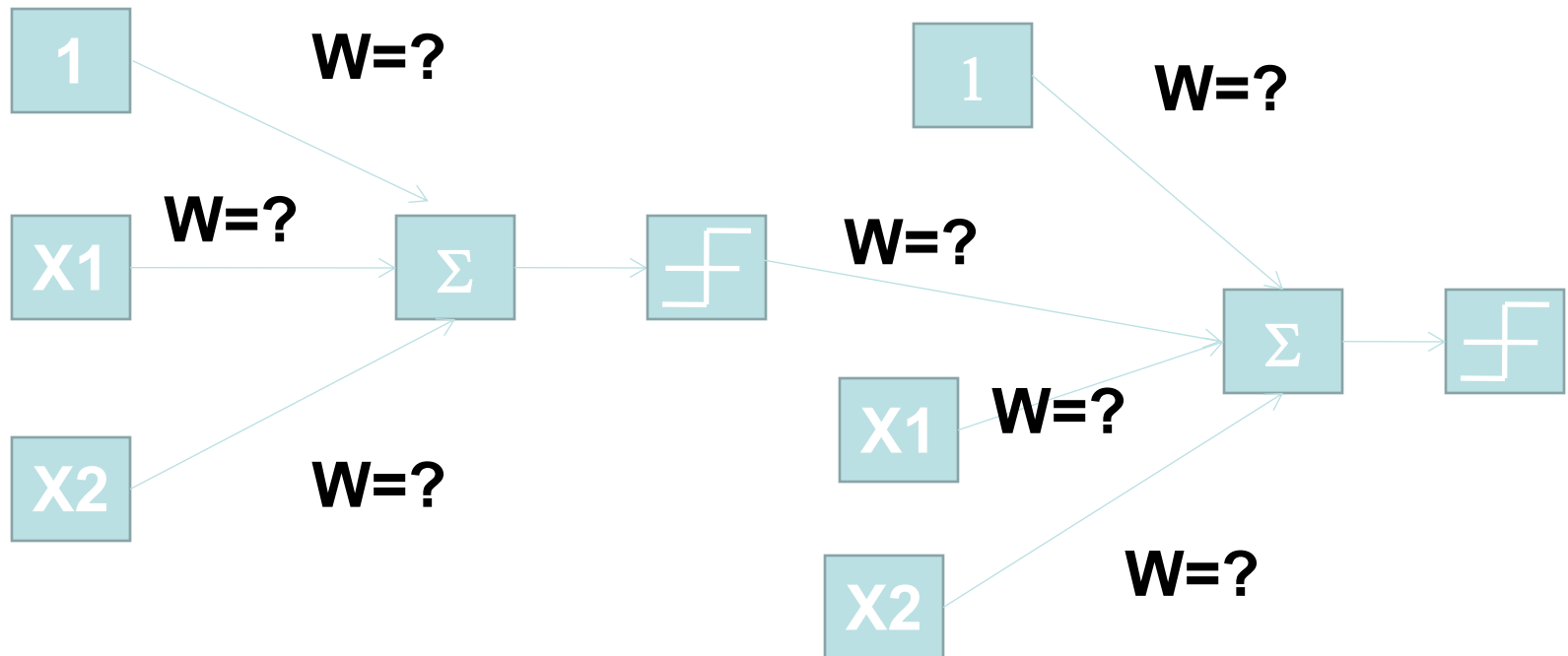
Perceptrons – Prática 3

Com o mesmo Perceptron da prática 1, teste a tabela XOR:

1) O programa consegue achar uma solução?
Por quê?

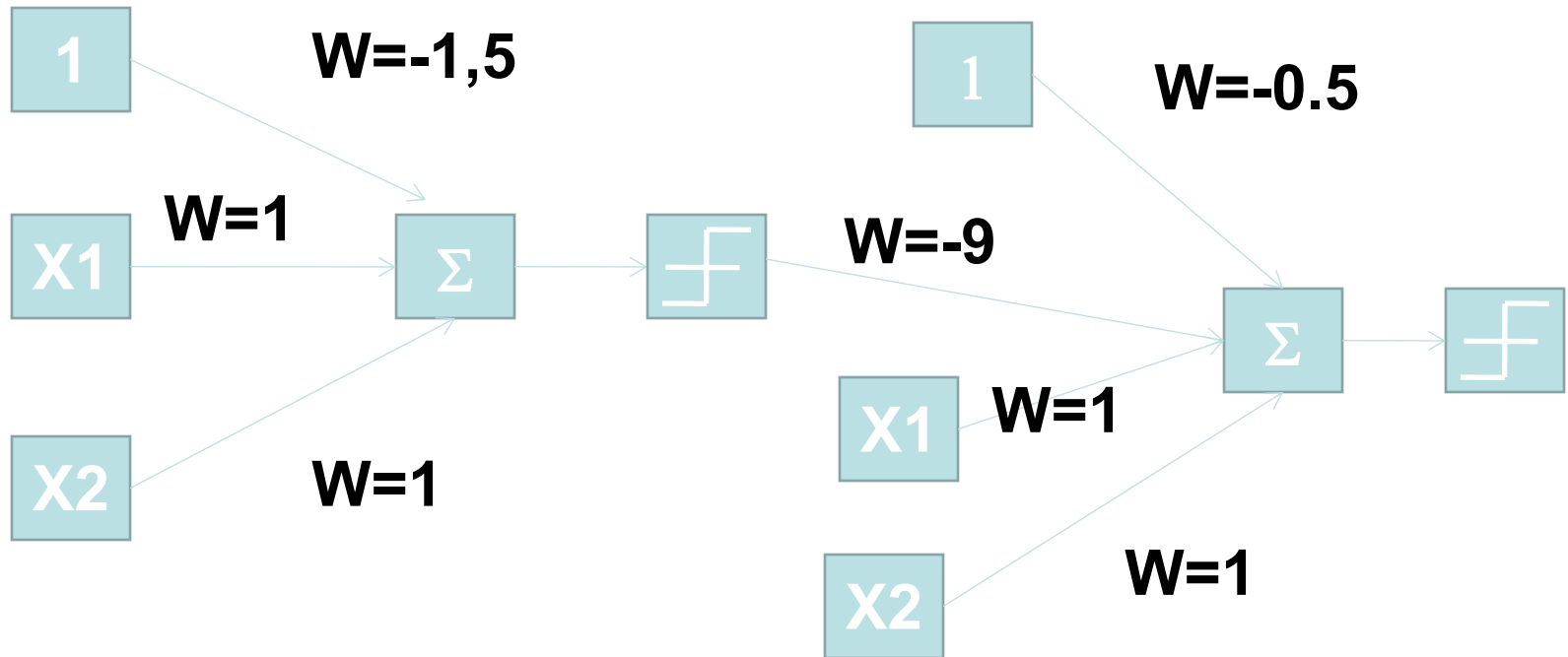
Perceptrons Multicamadas(I)

- Resolvem problemas não lineares, mas introduz um sério problema de aprendizagem;



Perceptrons Multicamadas(II)

- Uma solução para o problema XOR:



Perceptrons – Prática 4 (I)

VALENDO 2 PONTOS DA PARCIAL 2:

Crie seu perceptron, seguindo os seguintes passos:

1. Crie uma base de conhecimento para treinamento. Sua base deve ter pelo menos 5 características, 4 resultados possíveis e 12 exemplos:

	Carac 1	Carac 2	Carac 3	Carac 4	Carac 5
RESULTADO A	1	1	1	1	1
RESULTADO B	1	0	0	0	0
RESULTADO C	0	1	0	0	1
RESULTADO D	0	1	1	1	0
RESULTADO B	1	0	0	1	1
Até a linha 12...

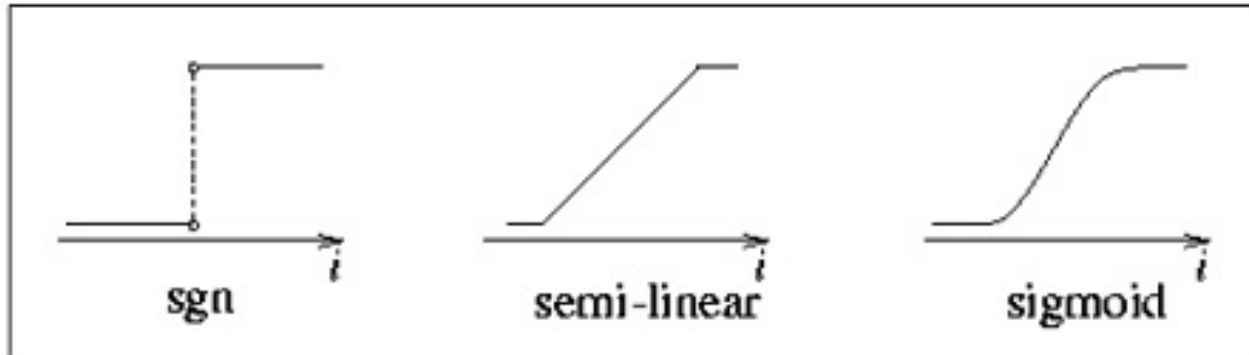
Perceptrons – Prática 4 (II)

2. A base de conhecimento criada deve ser baseada em um problema concreto, ou seja, as características e resultados devem ter significado. Dica: características -> sintomas do problema, resultados -> soluções do problema;
3. Implemente seu Perceptron em C++ (usando o programa da prática 2), de forma a alimentar a matriz de leitura com os exemplos da base de treinamento;
4. Verifique e responda:
 - a) Sua RNA conseguiu aprender os 12 elementos do treinamento? Eles são linearmente separáveis?;
 - b) Crie mais quatro exemplos e teste no seu programa. Quais eram o resultados esperados e o que a rede respondeu?
 - c) A RNA criada teria alguma utilidade prática? Por que?

Referências

- <http://www.intelliwise.com/reports/info2001.htm>
- <http://www.ucs.louisiana.edu/%7Eisb9112/dept/phil341/wisai/WhatisAI.html>
- <http://www.cs.bham.ac.uk/%7Eaxs/courses/ai.html>
- http://pt.wikipedia.org/wiki/Intelig%C3%A2ncia_artificial
- <http://pt.wikiquote.org/wiki/Intelig%C3%A2ncia>
- <http://www.ucb.br/prg/professores/rogerio/FIA/fundia.html>
- <http://to-campos.planetaclix.pt/neural/hop.html>
- http://codebetter.com/photos/greg_young/images/169874/320x394.aspx
- <http://www.dcc.fc.up.pt/~jpp/cia/node54.html>>
- http://www.livinginternet.com/i/ii_ai.htm
- <http://www.conpet.gov.br/ed/>
- <http://www.inbot.com.br/sete/>
- <http://alicebot.org>
- <http://www.cin.ufpe.br/~in1006/2003/AIParadigms.ppt>
- <http://www.stdwizard.org/>
- <http://expertise2go.com/webesie/>
- <http://www.din.uem.br/ia/neurais/>>
- <http://www.icmc.usp.br/~andre/research/neural/index.htm>
- <http://www.inf.unisinos.br/~osorio/neural.html>
- Inteligência Artificial – Elaine Rich e Kevin Knight – 2ª edição

Funções de Transferência



$$F(X) = \text{Sgn}(X)$$

If $X \geq 0$
 Then $Y = 1$
 Else $Y = 0$ (ou -1)

ou

IF $X \geq \text{Limiar}$
 Then $Y = 1$
 Else $Y = 0$ (ou -1)

$$F(X) = \text{Linear}(X, \text{Min}, \text{Max})$$

If $X < \text{Min}$
 Then $Y = 0$

If $X \geq \text{Min}$ and $X \leq \text{Max}$
 Then $Y = X$

If $X > \text{Max}$
 Then $Y = 1$

Obs.: $Y = X$ ou $Y = \text{Normaliza}(X)$

$$F(X) = \text{Sigmoid}(X)$$

Assimétrica

$$Y = \frac{1}{1 + \text{Exp}(-x)}$$

Simétrica

$$Y = \text{TanHip}(X)$$