Universidade do Sul de Santa Catarina
Ciência da Computação

Técnicas de Inteligência Artificial

Aula 06
Redes Neurais Artificiais

Max Pereira

Formas de Aprendizado

Aprendizado Supervisionado

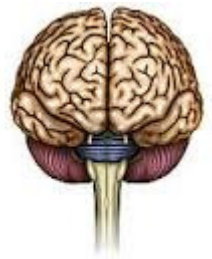
- ✓ Árvores de decisão
- ✓ K-Nearest Neighbor (KNN)
- ✓ Support Vector Machines (SVM)
- ✓ Redes Neurais

Aprendizado não Supervisionado

Aprendizado por Reforço

- Cérebro humano:
  Processador Ideal

Capaz de:
- -Aprender
- -Generalizar
- -Memorizar
- -Criar



- Maior órgão do Sistema Nervoso humano.
- Possui de 10 a 100 bilhões de células nervosas

- No cérebro, o **comportamento inteligente** é uma propriedade emergente de um grande número de **unidades simples** (ao contrário do que acontece com regras e algoritmos simbólicos).

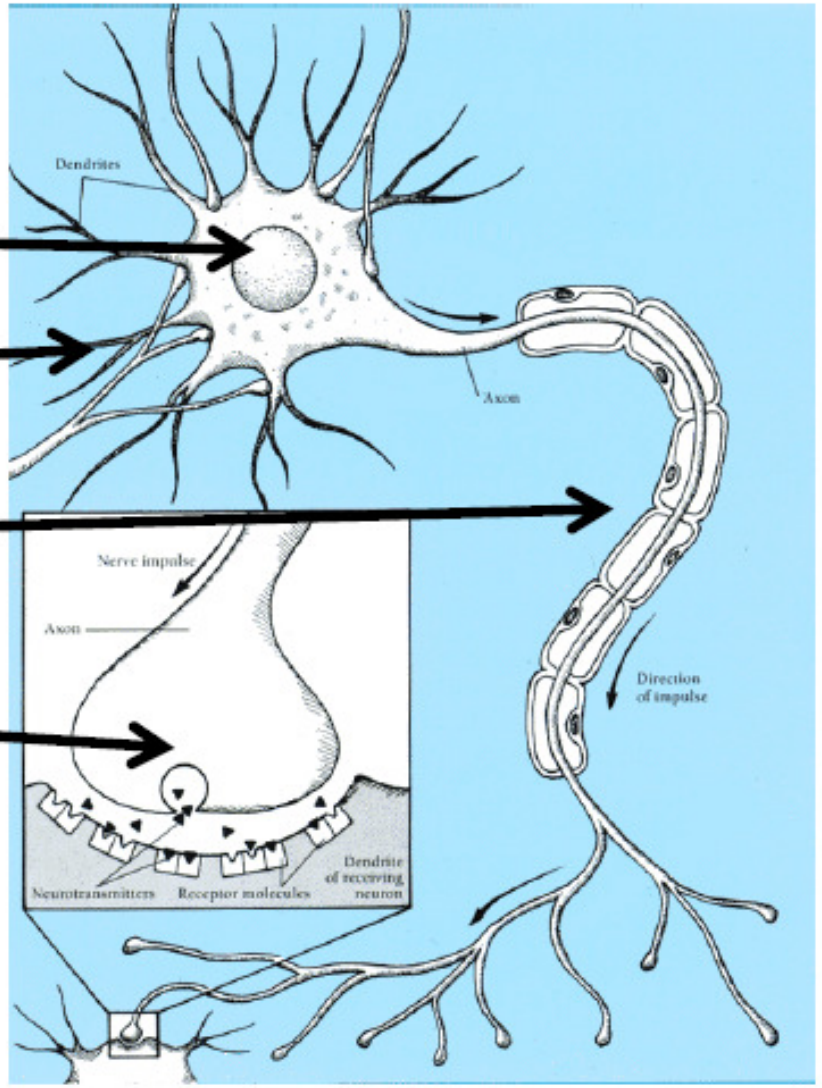
- Neurônios ligam e desligam em alguns milissegundos, enquanto o hardware atual faz o mesmo em nano segundos.

  - Entretanto, o cérebro realiza tarefas cognitivas complexas (visão, reconhecimento de voz) em décimos de segundo.

- O cérebro deve estar utilizando um **paralelismo massivo**.

Estrutura de um Neuronio:

- Corpo celular
- Dendritos
- Axônio
- Terminais sinápticos



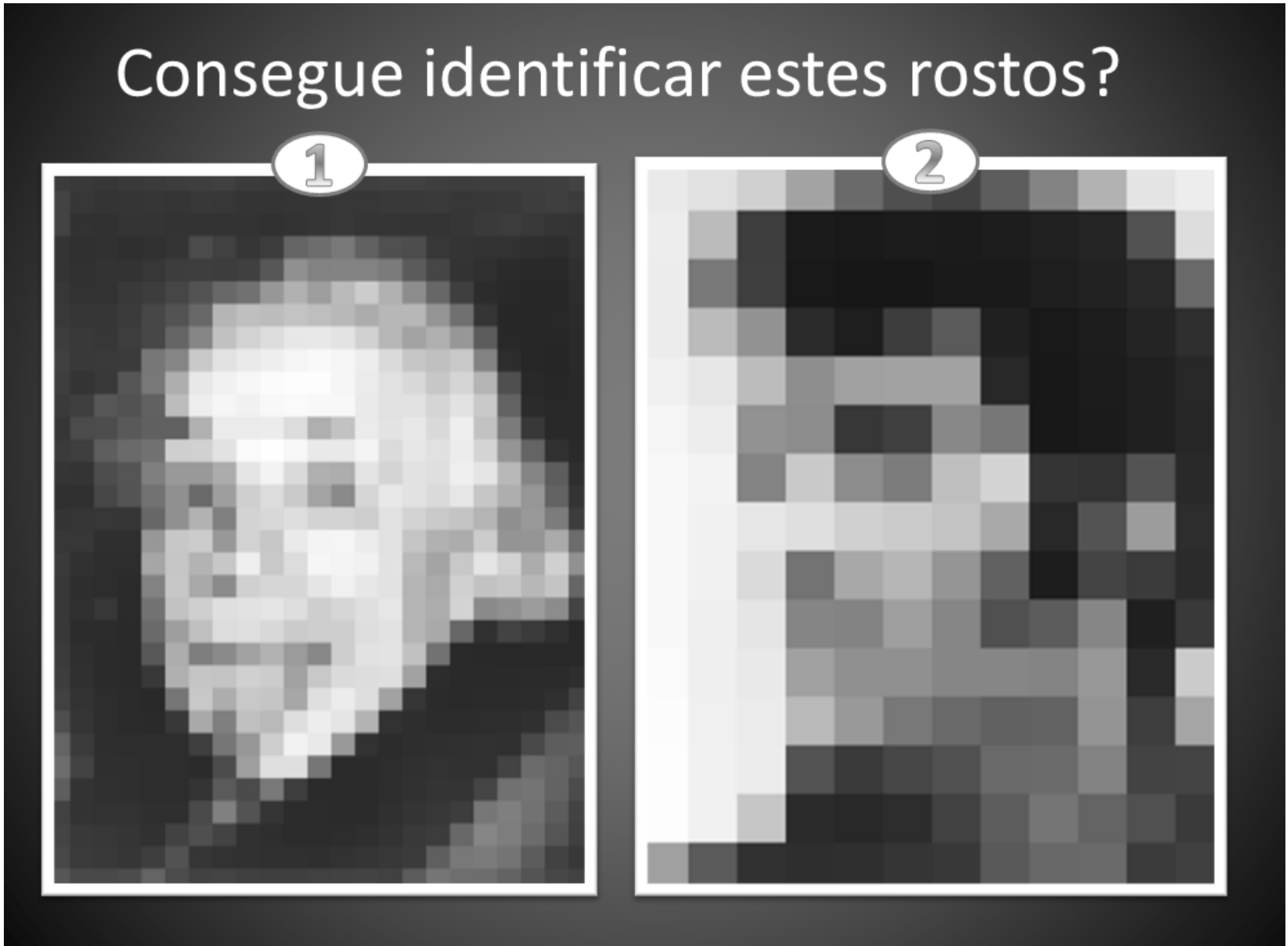
- Através dos **dentritos**, o neurônio recebe sinais de outros neurônios a ele conectados por meio das **sinapses**.

- Os sinais são acumulados no **corpo** do neurônio.

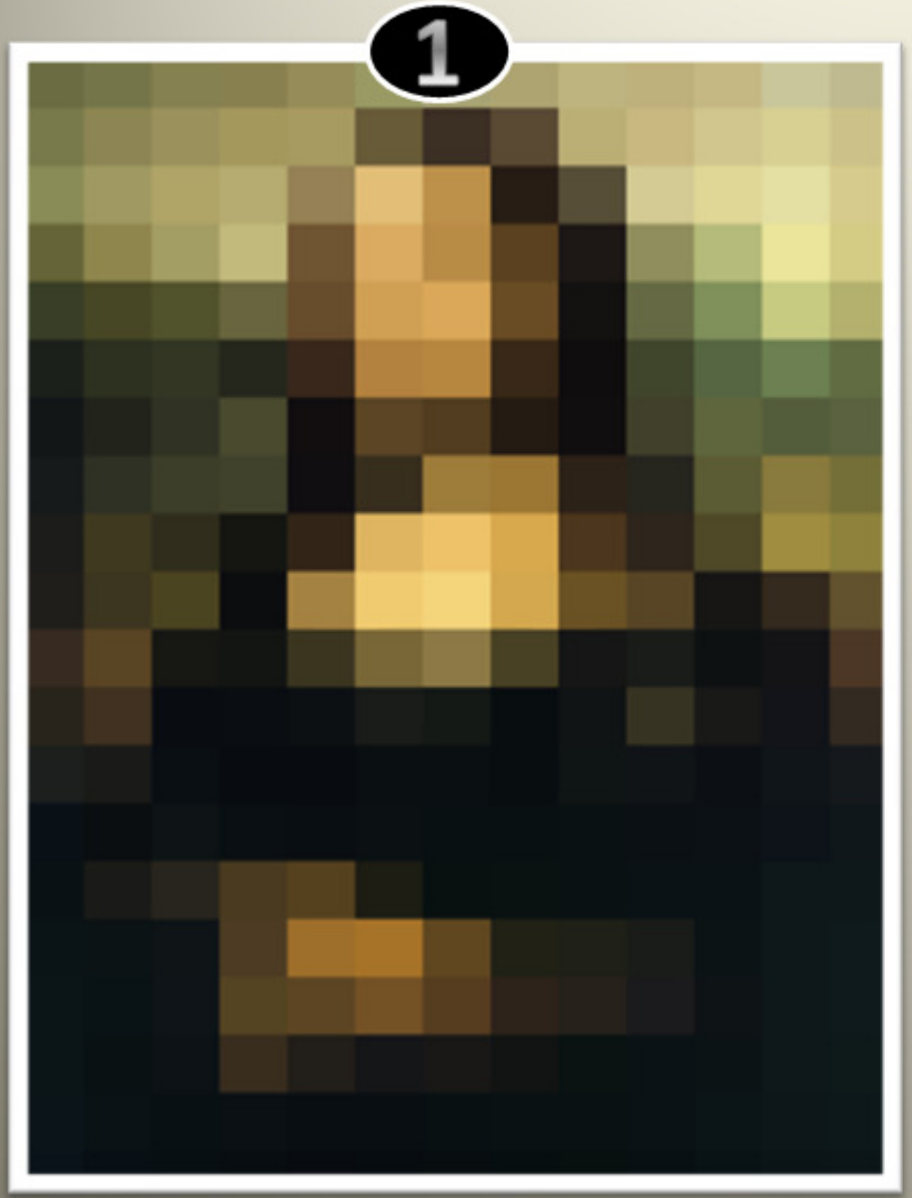
- Quando a soma dos sinais passa de um certo limiar (~ 50mV) um sinal é propagado no **axônio**.

- As **sinapses** tem um peso que pode ser:
  - excitatório: incrementam a soma dos sinais.
  - inibidor: decrementam.

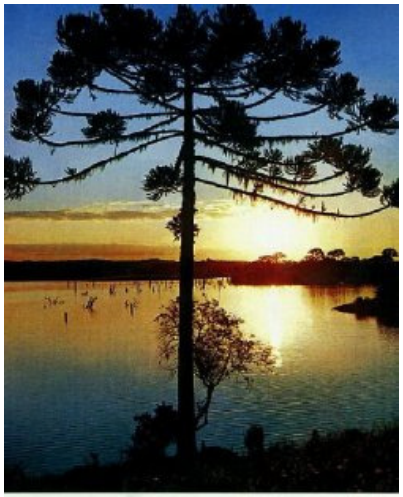
Capacidade de processar informação incompleta ou com ruído.

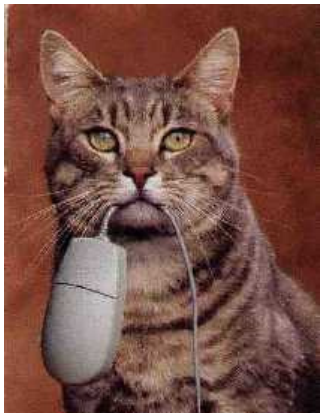


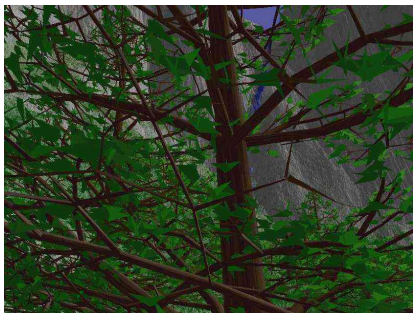
Consegue identificar estes rostos?



Qual é o diferente?







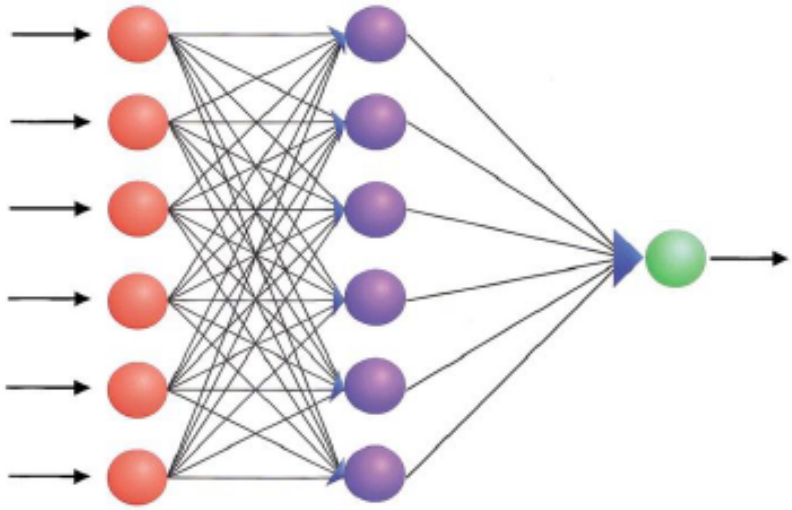
Capacidade de associar, extrapolar, inferir.

	Computador Von Neumann	**Sistema neural biológico**
Processador	complexo alta velocidade um ou poucos	simples baixa velocidade grande número
Memória	Separada do processador localizada não-endereçavel pelo conteúdo	integrada com processador distribuída endereçável pelo conteúdo
Computação	centralizada seqüencial programas armazenados	distribuída paralela aprendizado
Confiabilidade	muito vulnerável	robusto
Adequação	manipulações num. e simbólica	Problemas de percepção
Ambiente operacional	bem definido muito restrito	pouco definido não restrito

- **Redes Neurais** podem ser consideradas um paradigma diferente de computação.

- Inspirado na **arquitetura paralela** do cérebro humano.

  - Elementos de processamento simples.
  - Grande grau de interconexões.
  - Interação adaptativa entre os elementos.



O Modelo MCP (McCulloch e Pitts)

- Uma RNA é composta por várias unidades de processamento (nós), cujo funcionamento é bastante simples
- Essas unidades geralmente são ligadas por conexões (links) que estão associados a um determinado peso
- As unidades fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões
- O comportamento inteligente de uma RNA vem das interações entre as unidades de processamento da rede

O Modelo MCP (McCulloch e Pitts)

- Uma ligação de uma unidade j para unidade i serve para propagar a ativação a_j de j para i
- Cada ligação possui um peso $w_{j,i}$ associado, que determinar a força e o sinal da conexão
- Cada unidade i primeiro computa a soma dos pesos de suas entradas:

$$in_i = \sum_{j=0}^{n} W_{j,i} a_j$$

O Modelo MCP (McCulloch e Pitts)

Função de Ativação

- Então se aplica uma função de ativação g nesta soma para derivar a saída:

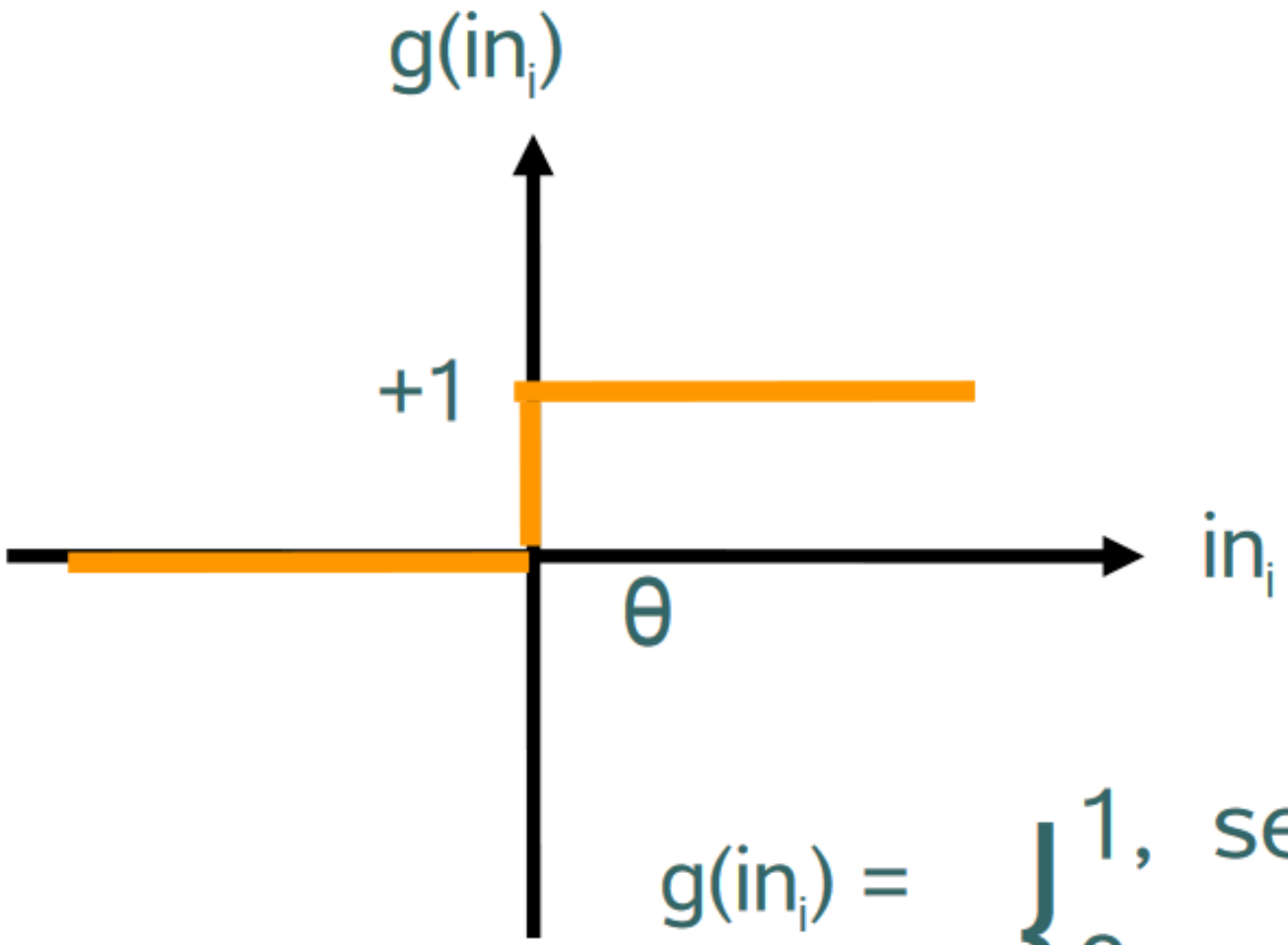
$$a_i = g(in_i) = g\left(\sum_{j=0}^{n} W_{j,i} a_j\right)$$

- Se este nível de atividade exceder um certo **limite ou limiar** (*threshold*) a unidade produz uma determinada resposta de saída

$$a_i \geq \theta$$

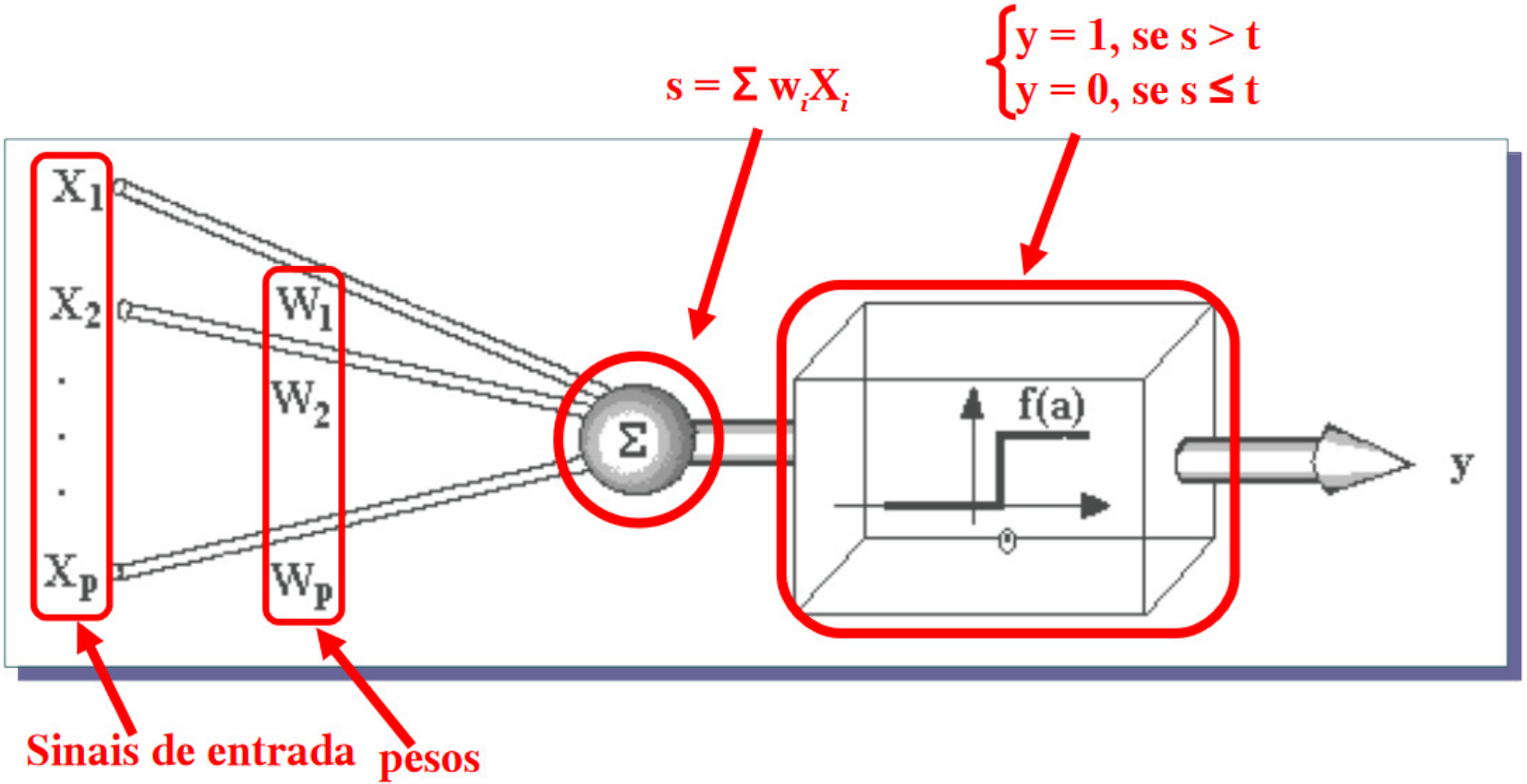
O Modelo MCP (McCulloch e Pitts)

Função limiar (*threshold*)



$$g(in_i) = \begin{cases} 1, & \text{se } in_i \geq \theta \\ 0, & \text{se } in_i < \theta \end{cases}$$

O Modelo MCP (McCulloch e Pitts)

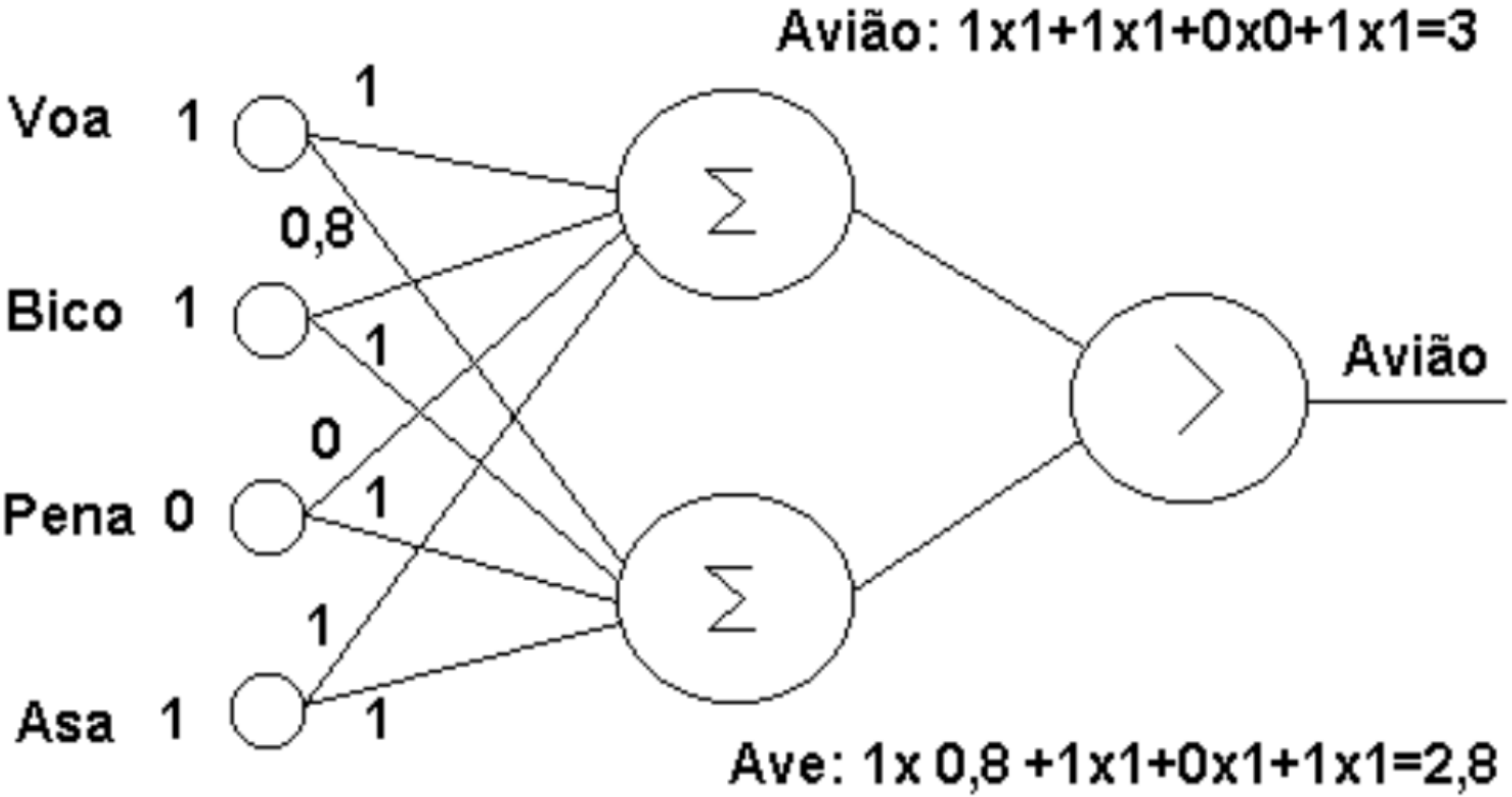


O Modelo MCP (McCulloch e Pitts)

- Este modelo foi simplificado
  - os nós em cada camada da rede disparam **sincronicamente**
    - as entradas em um instante t produzem a sua saída no tempo $t+1$
    - diferente do biológico, onde não existe sincronismo e nem ativação em tempo discreto

- Além disso, possuem outras limitações
  - com apenas uma camada só conseguem implementar funções linearmente separáveis
  - pesos fixos, não ajustáveis, não há aprendizado

Exemplo de funcionamento



Características das RNAs

- O comportamento inteligente vem das interações entre as unidade de processamento da rede
- Elas aprendem através de exemplos
- Processo de treinamento a partir dos casos reais
- Capaz de extrair regras básicas a partir de dados reais, diferindo da computação programada

Aprendizagem das RNAs

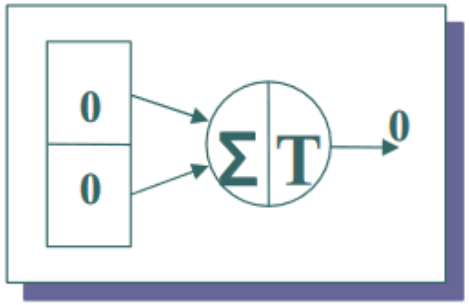
- A maioria dos modelos de redes neurais possui alguma **regra de treinamento**
  - onde os pesos de suas conexões são ajustados de acordo com os padrões apresentados
  - elas **aprendem através de exemplos**

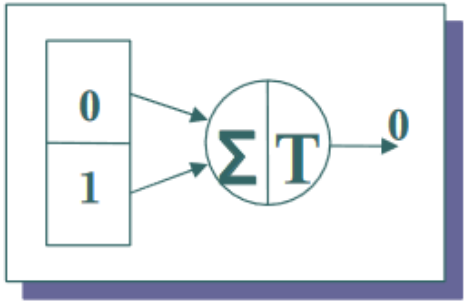
Implementação da porta lógica AND

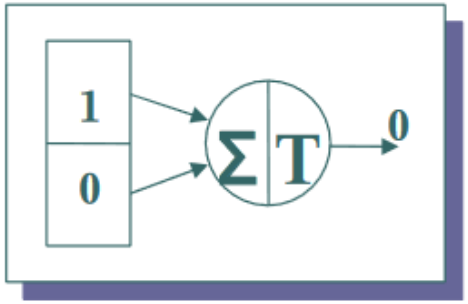
- Certificar-se de que todas as respostas estão corretas para cada conjunto de entradas pela tabela-verdade

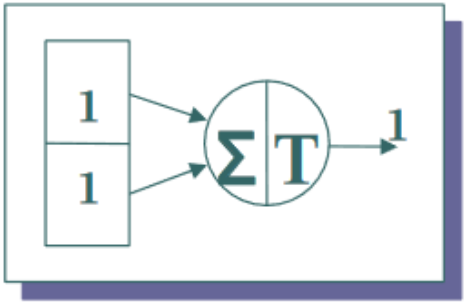
- A RNA possui um único neurônio de duas entradas e uma saída

Tabela Verdade - AND		
Entrada 1	Entrada 2	Saída
1	1	1
1	0	0
0	1	0
0	0	0





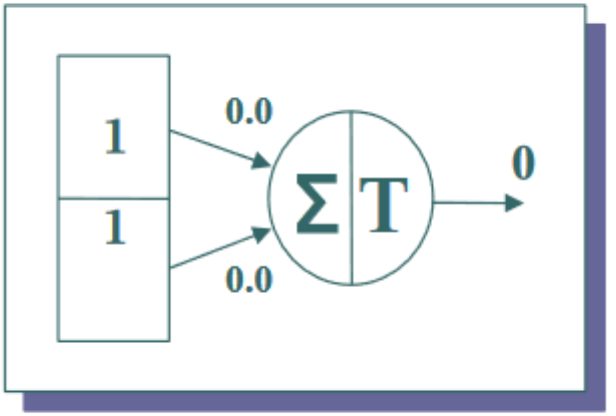




Implementação da porta lógica AND

- Para treinar a rede vamos seguir alguns passos:

para as entradas [1, 1]
→
pesos iniciais [0, 0]



$$\theta = 0{,}5$$

Função Soma

$$\sum_{i-1}^{n} x_i w_i \geq \theta$$

$y = 1$, se $s > \theta$
$y = 0$, se $s \leq \theta$

limiar

Implementação da porta lógica AND

- Para treinar a rede vamos seguir alguns passos:
  - Passo 1: Aplicar a função *Soma*

$$\sum_{i-1}^{n} x_i w_i \geq \theta$$

Soma = 1*0 + 1*0 = 0

  - Passo 2: Aplicar a função de *Transferência*

Soma ≤ 0,5 → *y* = 0

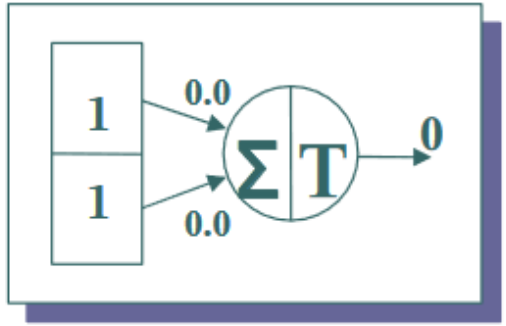
Soma > 0,5 → *y* = 1

  - Transferido 0 para a saída. Erro!!!!!

$$\theta = 0,5$$

Função Soma

$$\sum_{i-1}^{n} x_i w_i \geq \theta$$



y = 1, se s > θ

y = 0, se s ≤ θ

limiar

Implementação da porta lógica AND

- Passo 3: Ajuste do peso

Equação do erro:

$$E = S_d - S_o$$

onde

S_d é a saída desejada

S_o é a saída obtida

Fator de correção:

$$F = c*x*E$$

onde

c = 0,5 (constante)

x é a entrada

E é o erro

Equação do ajuste:

$$w_{novo} = w + F$$

Implementação da porta lógica AND

- Passo 3: Ajuste do peso

Calcular o erro: $E = 1 - 0 = 1$

Calcular o fator de correção:

$F_1 = c*E*x_1$ $\quad$ $F_2 = c*E*x_2$

$F_1 = 0{,}5*1*1$ $\quad$ $F_2 = 0{,}5*1*1$

$F_1 = 0{,}5$ $\quad$ $F_2 = 0{,}5$

Calcular o novo peso:

$w_{1novo} = w_1 + F_1$ $\quad$ $w_{2novo} = w_1 + F_2$

$w_{1novo} = 0 + 0{,}5$ $\quad$ $w_{2novo} = 0 + 0{,}5$

$w_{1novo} = 0{,}5$ $\quad$ $w_{2novo} = 0{,}5$

Equação do erro:

$$E = S_d - S_o$$

onde
- S_d é a saída desejada
- S_o é a saída obtida

Fator de correção:

$$F = c*x*E$$

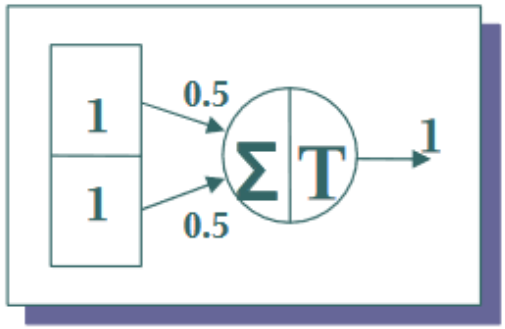
onde
- c = 0,5 (constante)
- x é a entrada
- E é o erro

Equação do ajuste:

$$w_{novo} = w + F$$

Implementação da porta lógica AND

- Para treinar a rede vamos seguir alguns passos:
  para as entradas [1, 1] ...
  pesos iniciais [0,5, 0,5]



  - Passo 1: Aplicar a função *Soma*
    $Soma = 1*0,5 + 1*0,5 = 1$
  - Passo 2: Aplicar a função de *Transferência*
    $Soma \leq 0,5 \rightarrow y = 0$
    $Soma > 0,5 \rightarrow y = 1$

  - Transferido 1 para a saída. **Correto!!!!!**

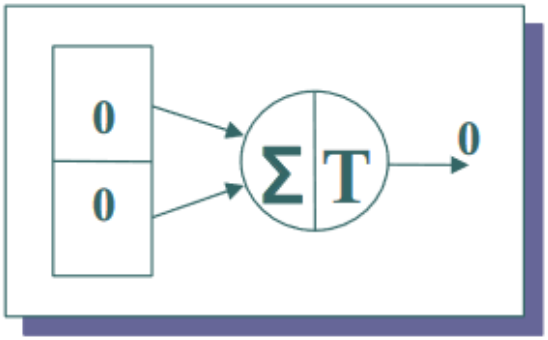
Implementação da porta lógica AND

Exercício

- Continuar treinando a rede
  - para as entradas [0, 0] e pesos [0,5, 0,5]
- Testar a rede
  - Para as entradas [0, 1] e [1, 0]

Implementação da porta lógica AND

- Para treinar a rede vamos seguir alguns passos:
  para as entradas [0, 0] e
  pesos [0,5, 0,5]



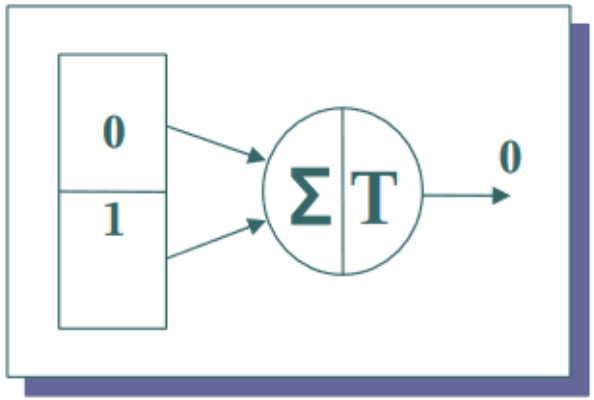
  - Passo 1: Aplicar a função *Soma*
    $Soma = 0*0,5 + 0*0,5 = 0$
  - Passo 2: Aplicar a função de *Transferência*
    $Soma \leq 0,5 \rightarrow y = 0$
    $Soma > 0,5 \rightarrow y = 1$

  - Transferido 0 para a saída. **Correto!!!!!**

Implementação da porta lógica AND

- Testar a rede: para as entradas [0, 1] e pesos [0,5, 0,5]
  - Passo 1: Aplicar a função *Soma*

    $Soma = 0*0,5 + 1*0,5 = 0,5$



  - Passo 2: Aplicar a função de *Transferência*

    $Soma \leq 0,5 \rightarrow y = 0$

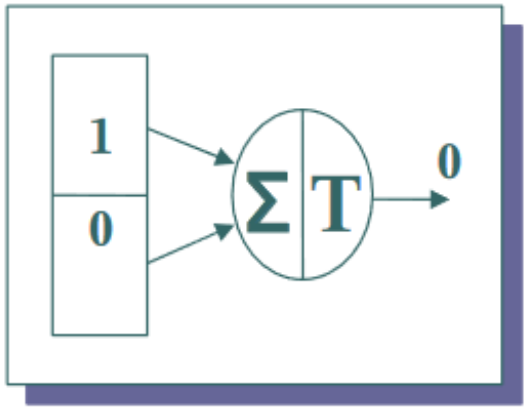
    $Soma > 0,5 \rightarrow y = 1$

  - Transferido 0 para a saída. **Correto!!!!!**

Implementação da porta lógica AND

- Testar a rede: para as entradas [1, 0] e pesos [0,5, 0,5]
  - Passo 1: Aplicar a função *Soma*

    Soma = 1*0,5 + 0*0,5 = 0,5



  - Passo 2: Aplicar a função de *Transferência*

    Soma ≤ 0,5 → *y* = 0

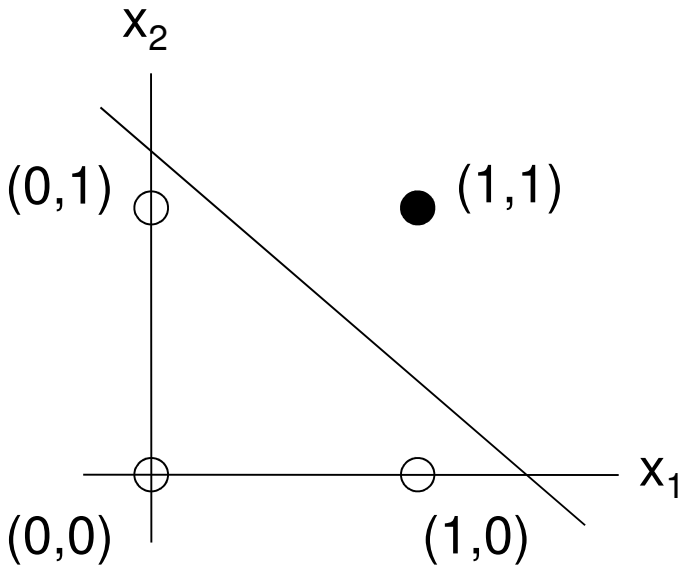
    Soma > 0,5 → *y* = 1

  - Transferido 0 para a saída. **Correto!!!!!**

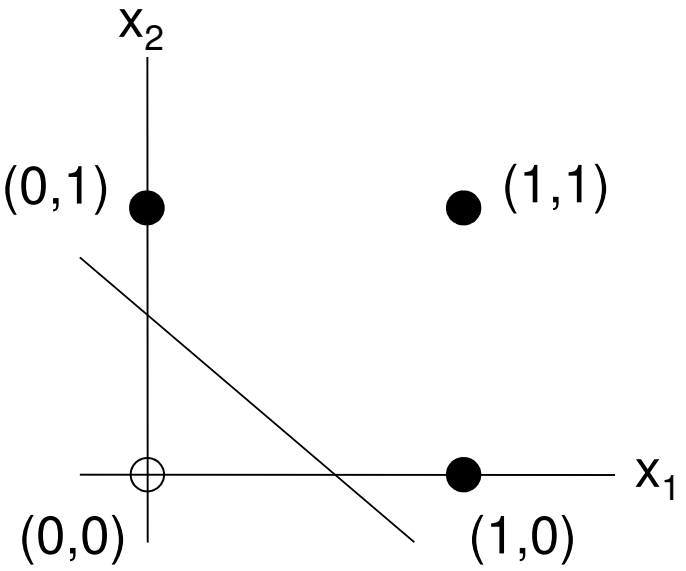
Modelo MCP

- O Modelo MCP é um discriminador linear que pode ser usado, em certos casos, como *classificador de padrões*.
- As funções lógicas **E** e **OU** são linearmente separáveis (implementáveis com o modelo MCP)
- A função **XOR** ou **ou-exclusivo** não é linearmente separável.

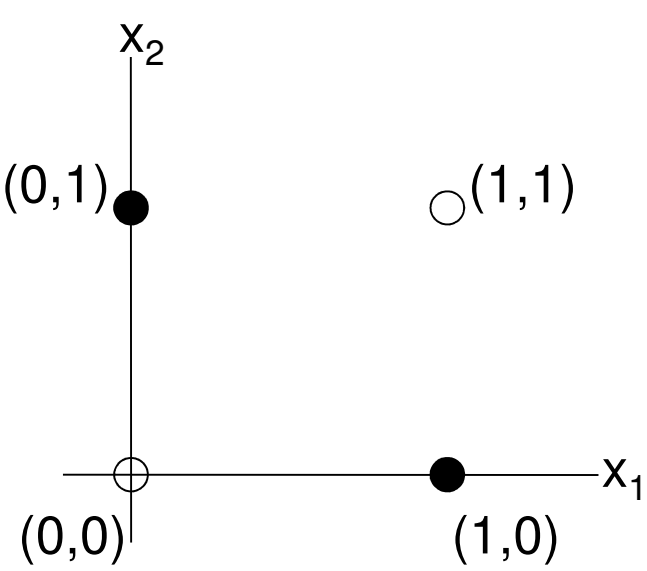
Funções Booleanas representadas no plano binário



AND

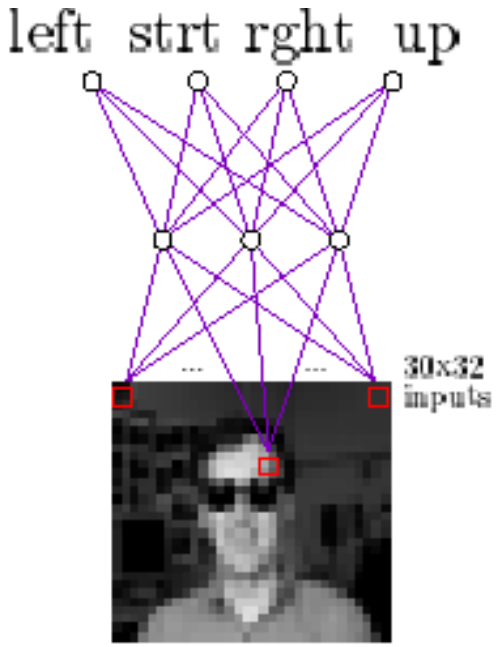


OR



XOR

Face Recognition



Typical input images

90% accurate learning head pose, and recognizing 1-of-20 faces

Handwritten digit recognition

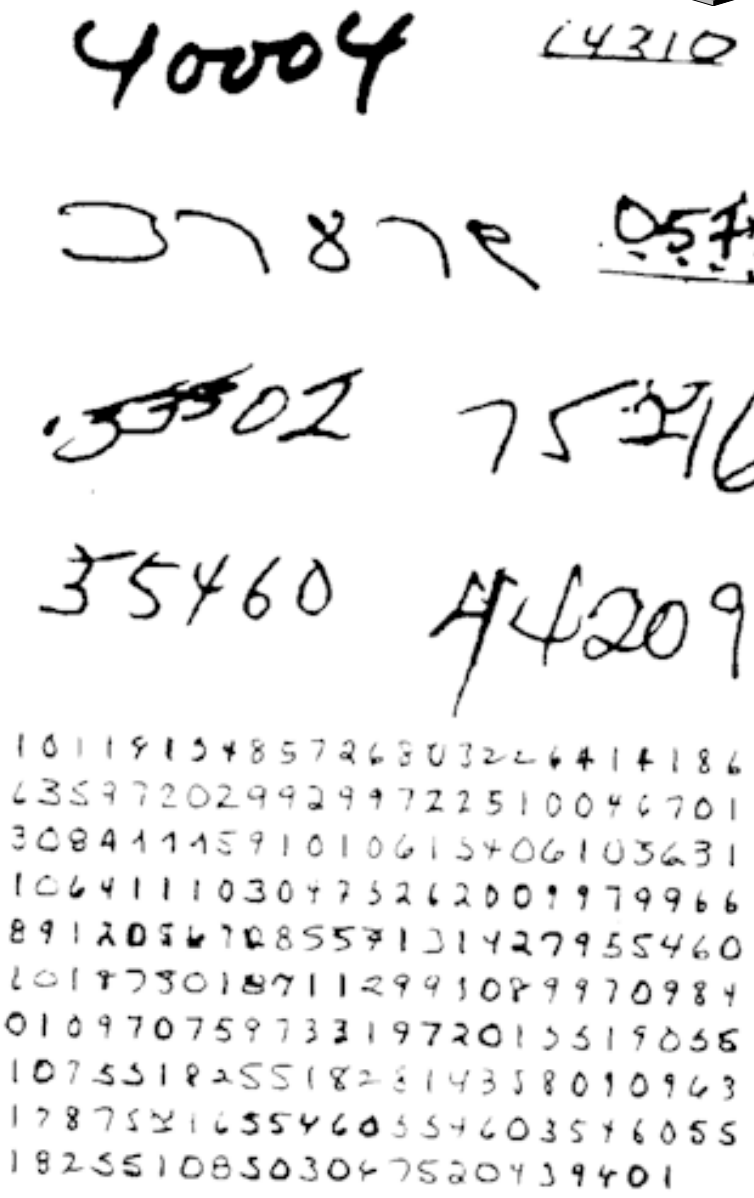


FIGURE 10.8
Examples of ZIP code image, and segmented and normalized numerals from the testing set. (*Source:* Reprinted with permission from Y. Le Cun, et al., "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, 1:541–551, 1989. ©1989 The MIT Press.)