

# Aprendizado de Máquina: habilidade fundamental da IA

prof<sup>a</sup> Dr<sup>a</sup> Carine G. Webber

# Aprendizado de Máquina

Pesquisa métodos computacionais relacionados à aquisição de (Mitchel, 1990):

- Novos conhecimentos
- Novas habilidades
- Novas formas de organizar o conhecimento existente

Principais objetivos:

- Automatizar aquisição de conhecimento
- Melhorar o desempenho do sistema pela experiência.

# Descoberta de Conhecimento



Efeito desejado do Aprendizado de Máquina.

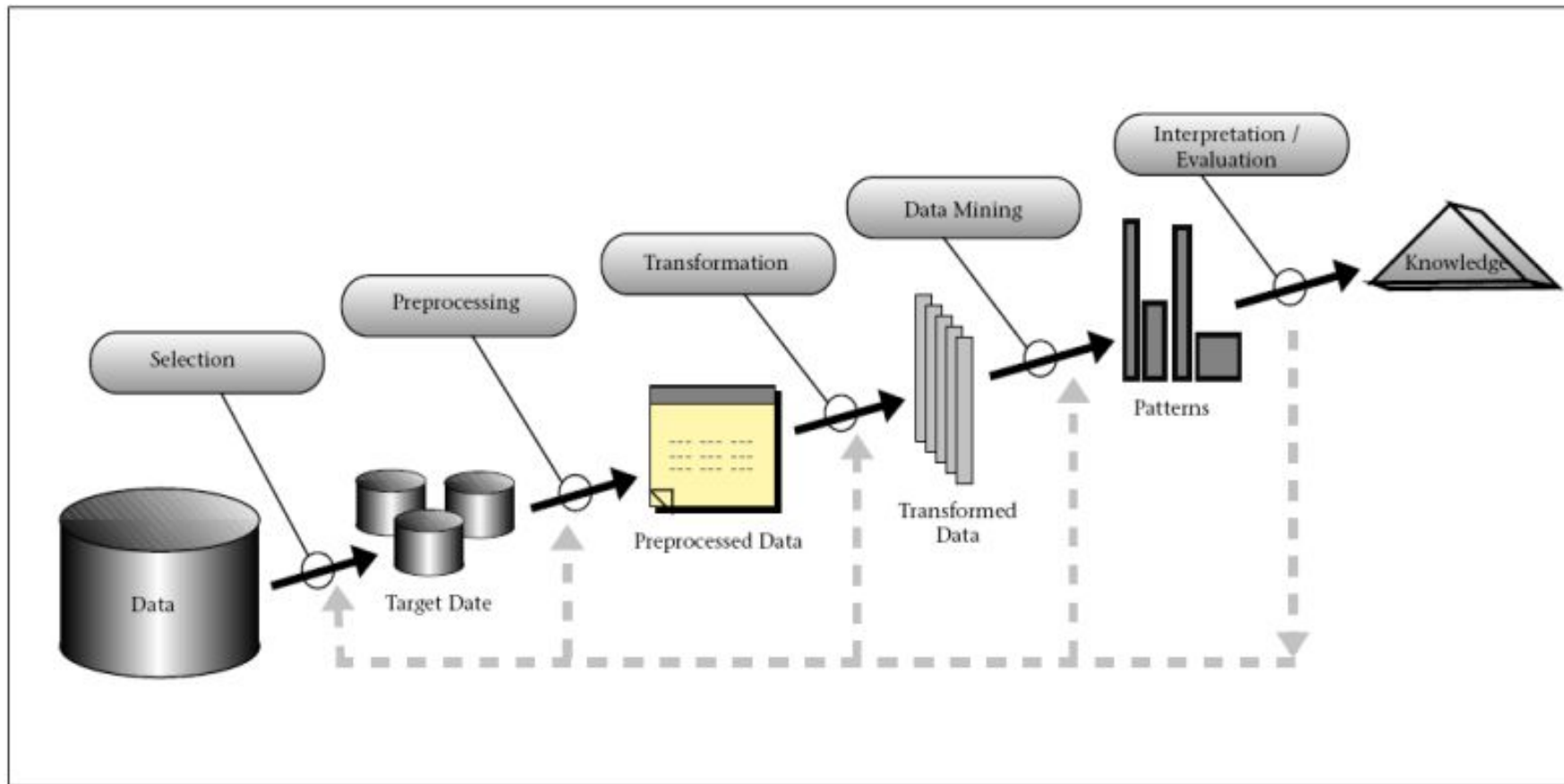
Introduzido por Fayaad (1996).

É vista como uma aplicação da IA para bancos de dados.

É um processo semiautomático.

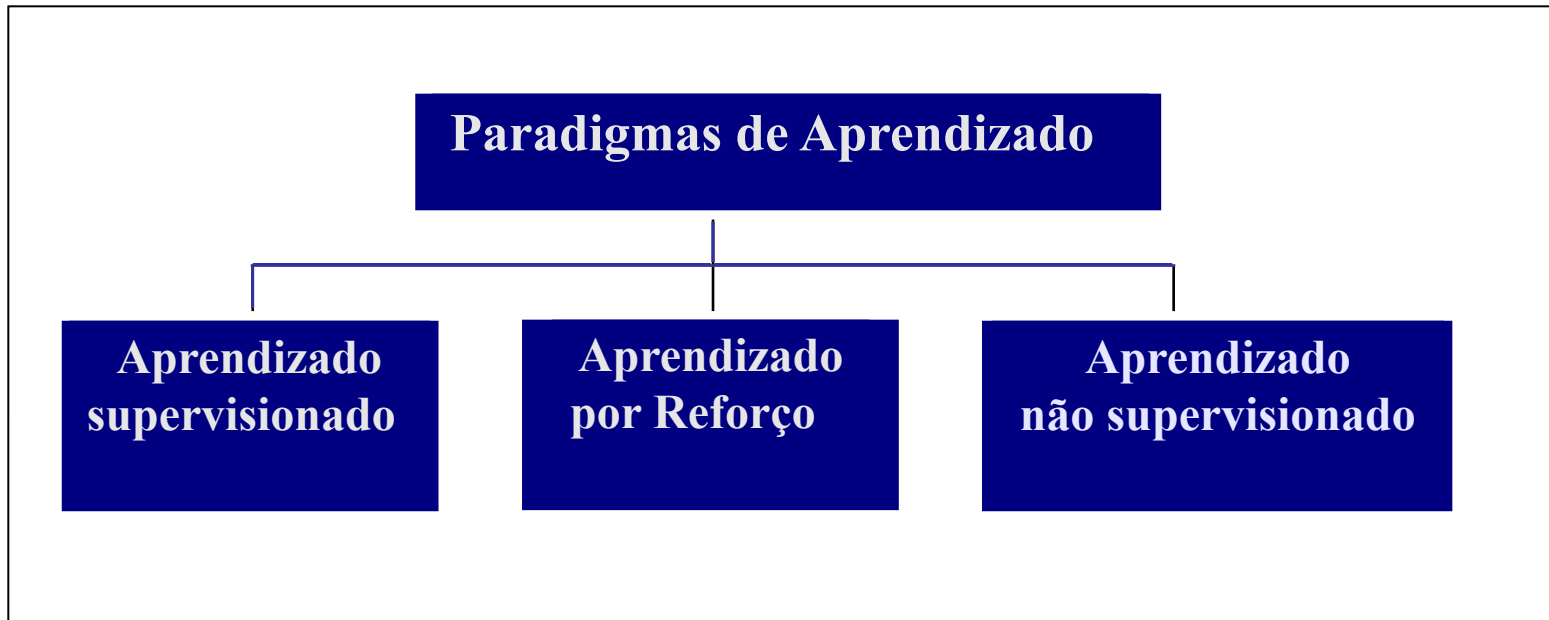
É independente de método de aprendizagem.

# Processo de Descoberta de Conhecimento

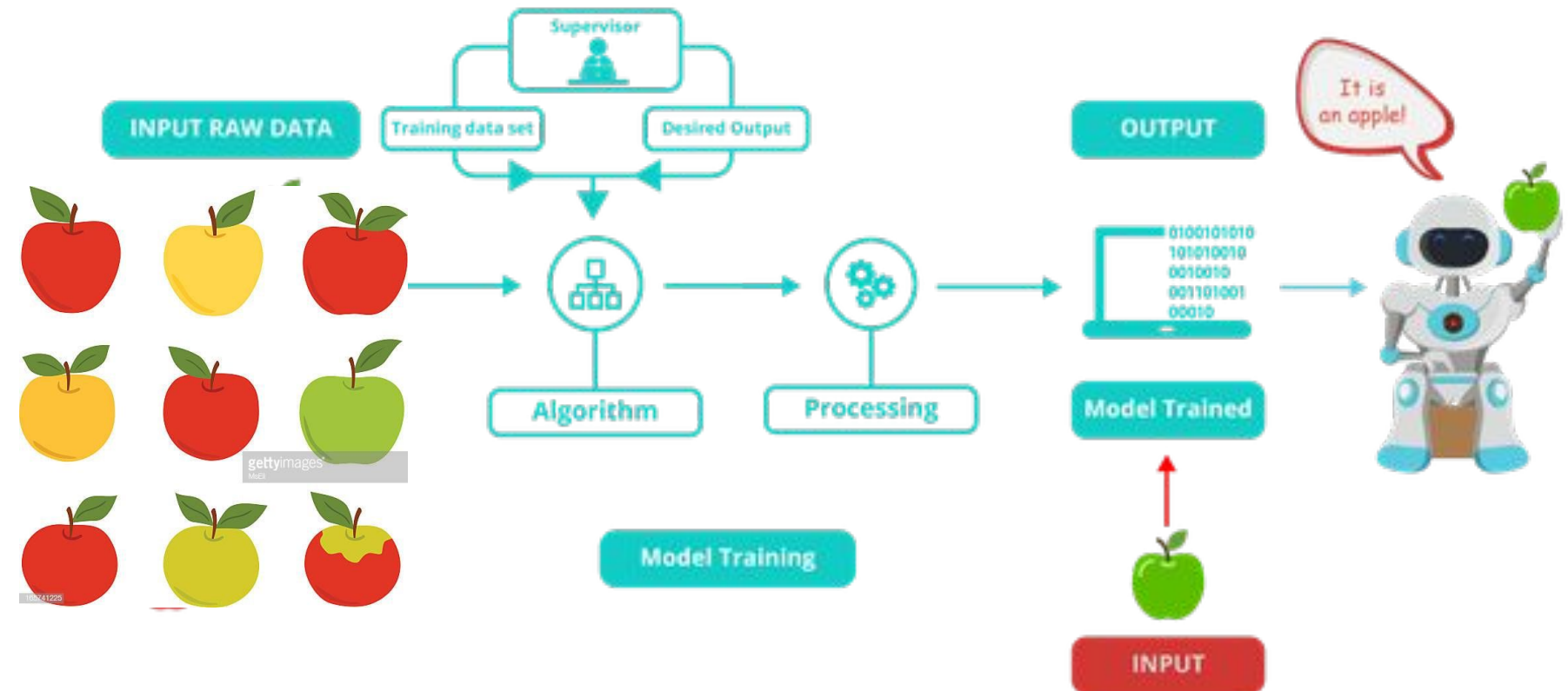


(Fayyad, 1996)

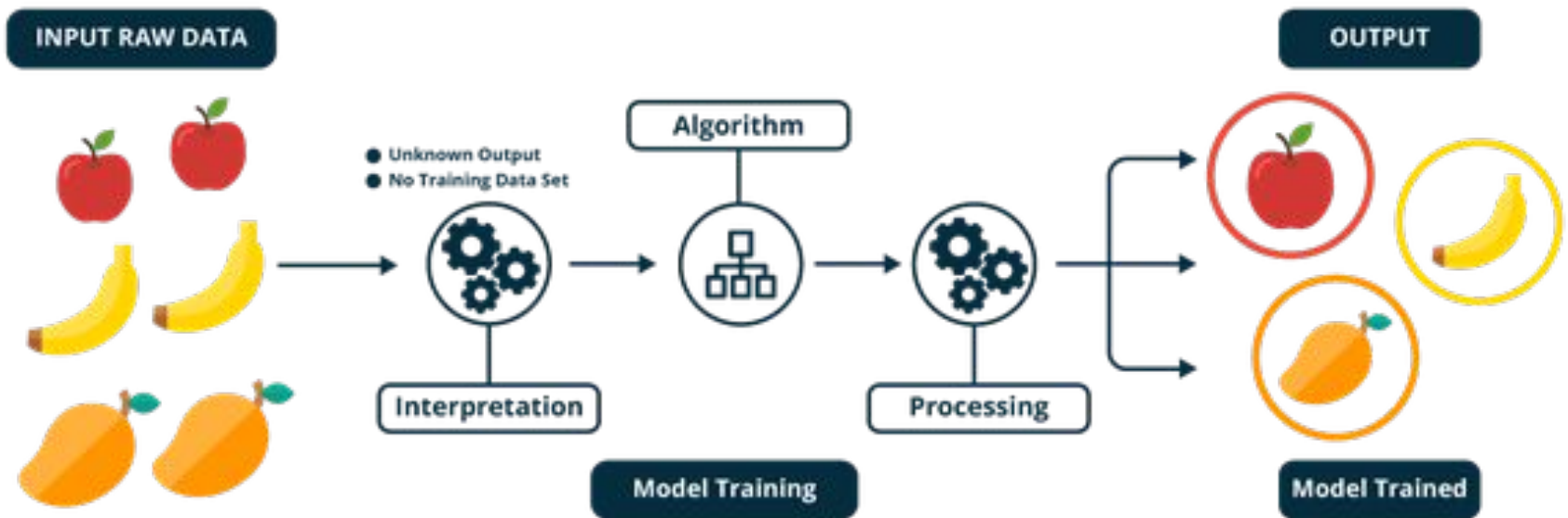
# Paradigmas Algorítmicos



# Com supervisão



# Sem supervisão



# Tipos de Aprendizado



## Supervisionado

- Resolve problemas de classificação de dados
- Ocorre a partir de exemplos previamente classificados
- Modelo dos dados é conhecido

## Não supervisionado

- Resolve problemas de agrupamento de dados similares
- As categorias estão implícitas e subjacentes aos dados
- Modelo dos dados é desconhecido



# Aprendizado por Reforço

○ agente aprende através de interações contínuas com o ambiente.

○ aprendizado ocorre por meio de

a) recompensas altas dadas a cada ação de sucesso

b) recompensas baixas para ações neutras

c) recompensas nulas ou negativas para

ações indesejadas ou proibitivas.





**UCS**

UNIVERSIDADE  
DE CAXIAS DO SUL



# Redes Neurais Artificiais

Universidade de Caxias do Sul  
Área de Exatas e Tecnologias

# Surgimento das Redes Neurais



## (1943) McCulloch & Pitts

- Provam, teoricamente, que qualquer função lógica pode ser implementada utilizando unidades de soma ponderada e *threshold*.

## (1949) Hebb desenvolve algoritmo para treinar RNA (aprendizado Hebbiano)

- Se dois neurônios estão simultaneamente ativos, a conexão entre eles deve ser reforçada.

# Surgimento das Redes Neurais



(1958) Von Neumann mostra interesse em modelagem do cérebro (RNA)

- The Computer and the Brain, Yale University Press

(1959) Rosenblatt implementa primeira RNA, a rede Perceptron

- Ajuste iterativo de pesos
- Prova teorema da convergência

# Surgimento das Redes Neurais

(1969) Minsky & Papert analisam Perceptron e mostram suas limitações

- Não poderiam aprender a resolver problemas simples como o OU-exclusivo
- Causou grande repercussão

# Surgimento das Redes Neurais

(1969) Minsky & Papert analisam Perceptron e mostram suas limitações

- Não poderiam aprender a resolver problemas simples como o OU-exclusivo
- Causou grande repercussão

# Evolução das Redes Neurais

(1971) Aleksander propõe redes Booleanas

(1972) Kohonen e Anderson trabalham com RNA associativas

(1975) Grossberg desenvolve a Teoria da Ressonância Adaptiva (redes ART)

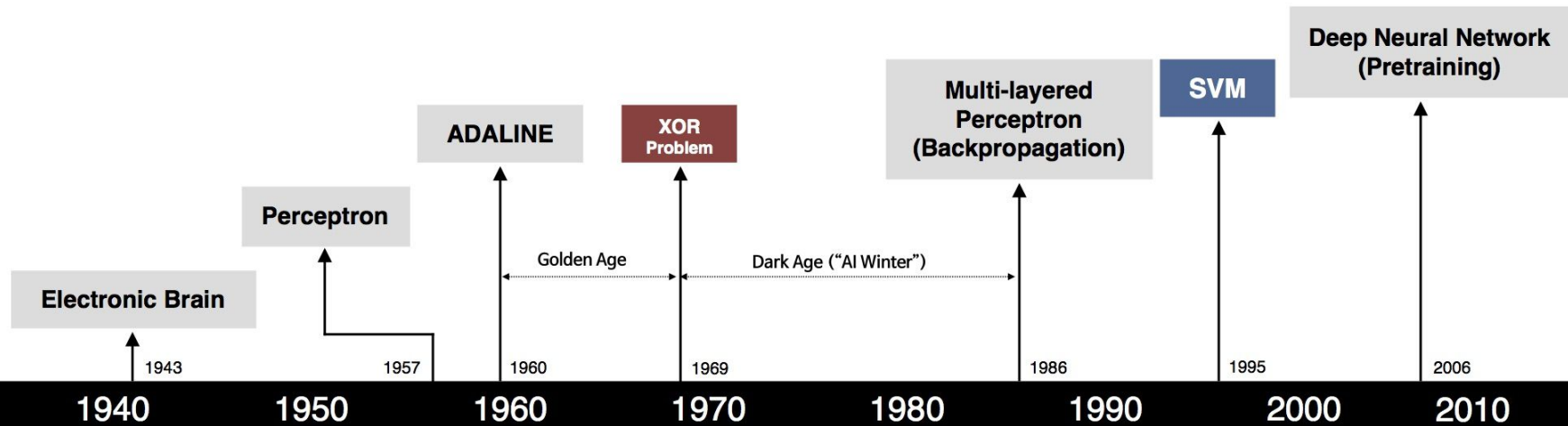
(1982) Hopfield mostra que Redes Neurais podem ser tratadas como sistemas dinâmicos

(1986) Hinton, Rumelhart e Williams, propõem algoritmo de aprendizagem para redes multi-camadas

Anos 90 – momento de compreensão e estudos (aparente estagnação)

Anos 2000 – ampla expansão e uso de redes neurais

# Timeline



S. McCulloch – W. Pitts



F. Rosenblatt



B. Widrow – M. Hoff



M. Minsky – S. Papert



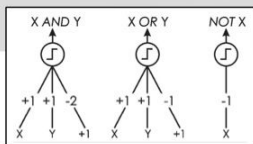
D. Rumelhart – G. Hinton – R. Williams



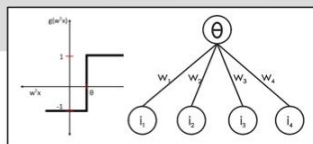
V. Vapnik – C. Cortes



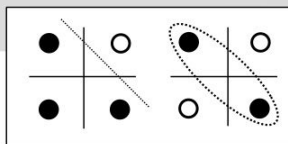
G. Hinton – S. Ruslan



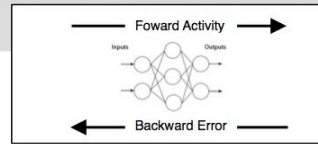
- Adjustable Weights
- Weights are not Learned



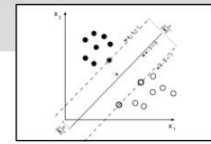
- Learnable Weights and Threshold



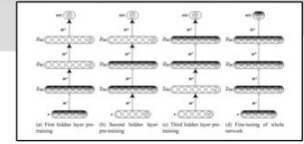
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



- Limitations of learning prior knowledge
- Kernel function: Human Intervention



- Hierarchical feature Learning



# Características do Cérebro

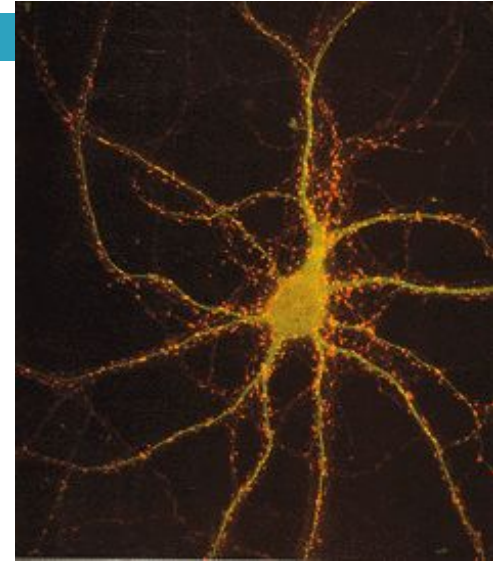
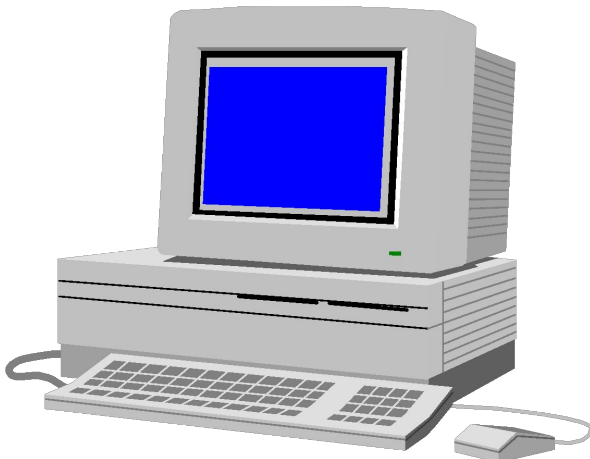


- Dez bilhões ( $10^{10}$ ) de neurônios
- Tempo de ativação do neurônio  $>10^{-3}$ seg
- Reconhecimento de uma face  $\sim 0.1$  seg
- Em média cada neurônio tem vários milhares de conexões
- Realiza centenas de operações por segundo
- Alto nível de computação paralela
- Mantém representações distribuídas
- Alta capacidade para Reconhecimento de padrões, associação, e tolerância a ruídos.

# O cérebro e o computador

18

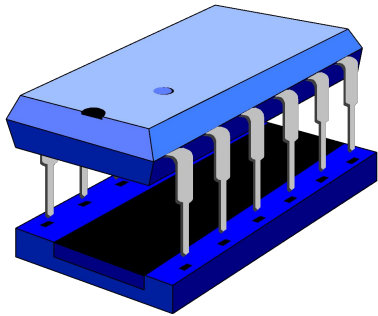
- O cérebro
  - Reconhecimento de padrões
  - Associação
  - Complexidade
  - Tolerância a ruídos



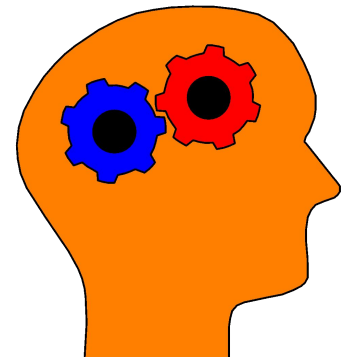
- O computador
  - Cálculos
  - Precisão
  - Lógica

# O contraste nas arquiteturas

19

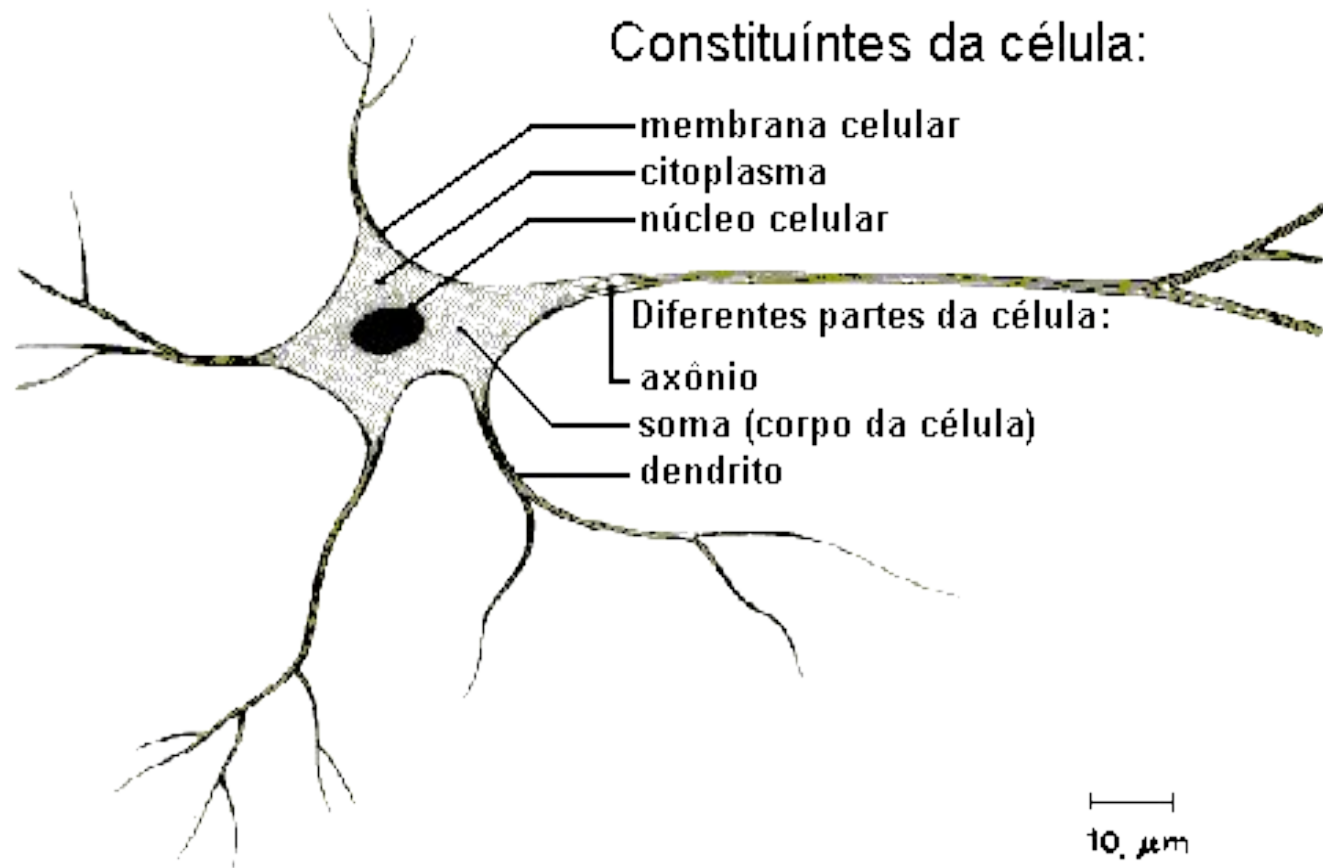


- Arquitetura Von Neumann usa uma única unidade de processamento;
  - Dez milhões de operações por seg
  - Absoluta precisão aritmética
- O cérebro usa muitos processadores lentos agindo em paralelo.



# A estrutura de um neurônio

20



# A estrutura de um neurônio

21

- Um neurônio se ativa somente se seu sinal de entrada excede um certo valor limiar (*threshold*) em um curto período de tempo.
- Sinapses variam em força:
  - Boas conexões permitem um sinal amplo.
  - Conexões fracas geram um sinal fraco.
  - Sinapses podem ser **excitatórias** ou **inibitórias**.

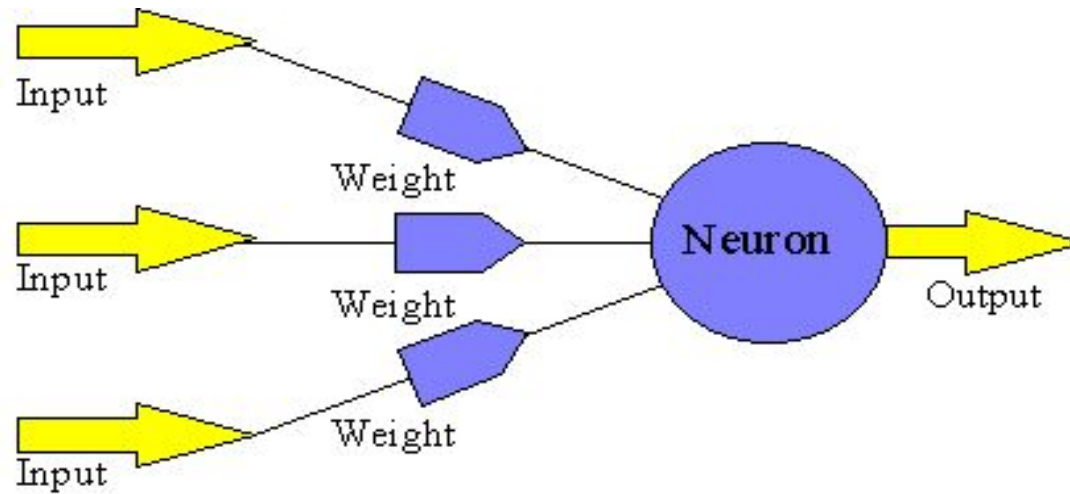
# A estrutura de um neurônio

22

Um neurônio tem um corpo celular, uma entrada conectada (o dendrito) e uma saída conectada (o axônio).

- Axônios se conectam aos dendritos via sinapses.
- Sinais eletro-químicos são propagados das entradas, através do corpo celular, e pelo axônio aos outros neurônios.

# Neurônio Artificial



# Propriedades dos Neurônios

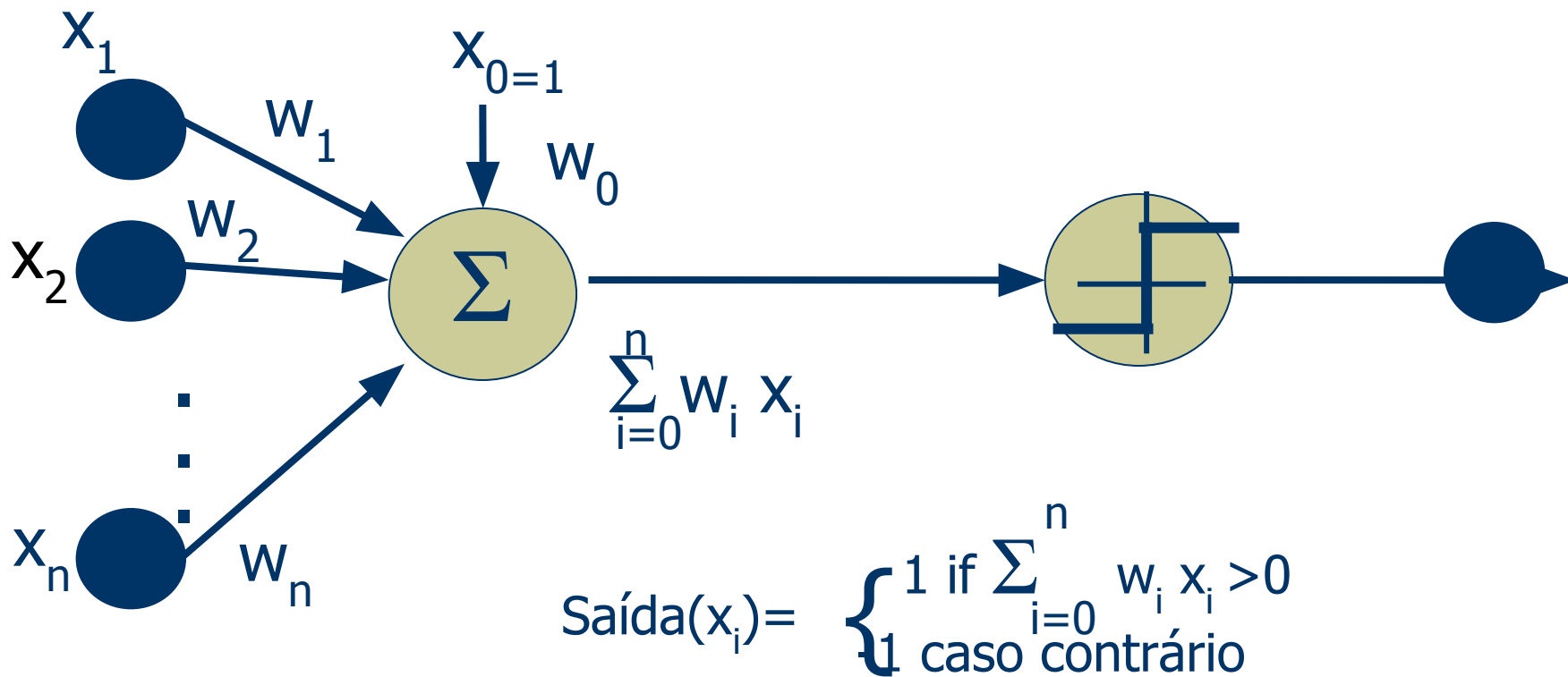
- ◆ Várias unidades de neurônios com limiares de ativação;
- ◆ Vários pesos entre as conexões de neurônios;
- ◆ Processamento altamente paralelo e distribuído;
- ◆ Aprendizagem pela correção dos pesos entre as conexões de neurônios.



# Domínio de problemas

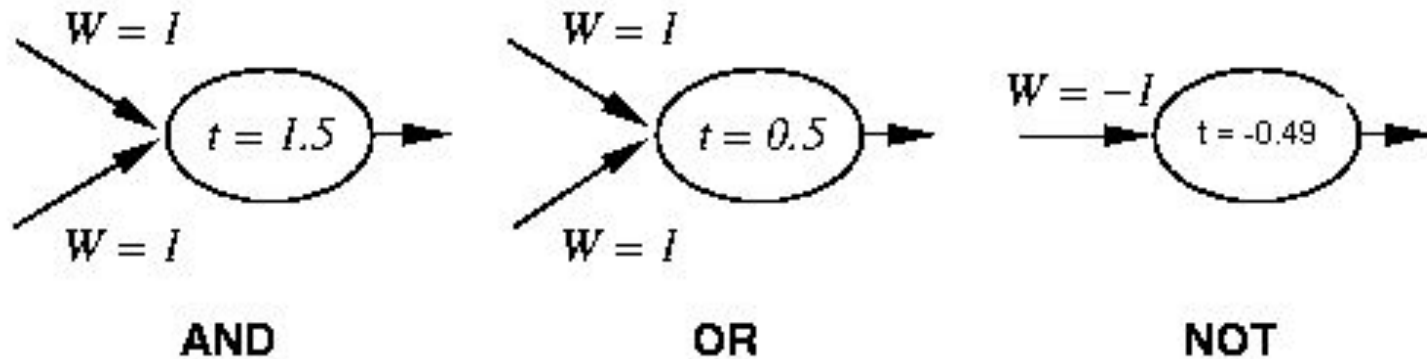
- ◆ Entrada é realizada por valores discretos ou contínuos (sensor de entrada)
- ◆ Saída é composta por valores discretos ou contínuos (reais)
- ◆ Saída é um vetor de valores
- ◆ Forma da função objetivo é desconhecida

# Perceptron



# Treinamento do Perceptron

27

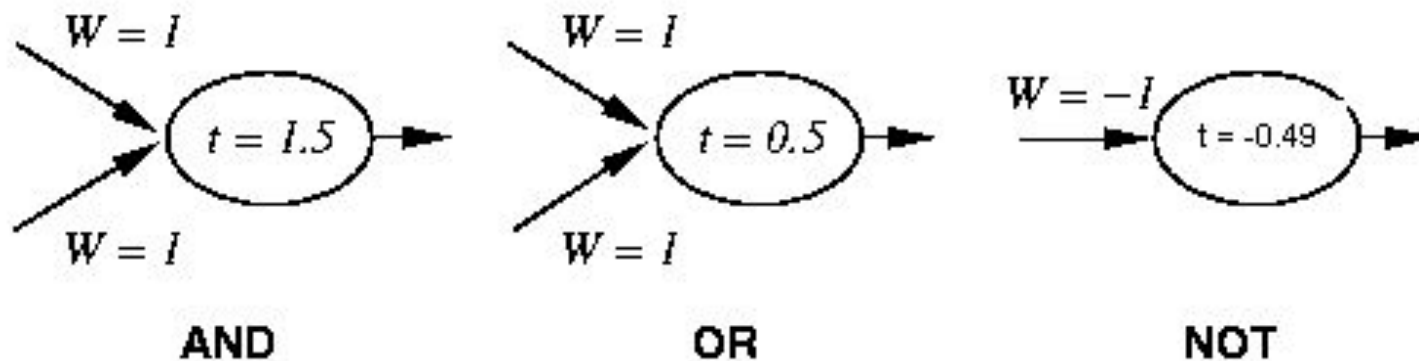


$$\text{Saída} = \begin{cases} 1 & \text{se } \sum w_i x_i > t \\ 0 & \text{caso contrário} \end{cases}$$

- ◆ Limiar (*threshold*) linear é usado.
- ◆  $W$  – valor do peso
- ◆  $t$  – valor do limiar (*threshold*)

# Treinamento do Perceptron

28

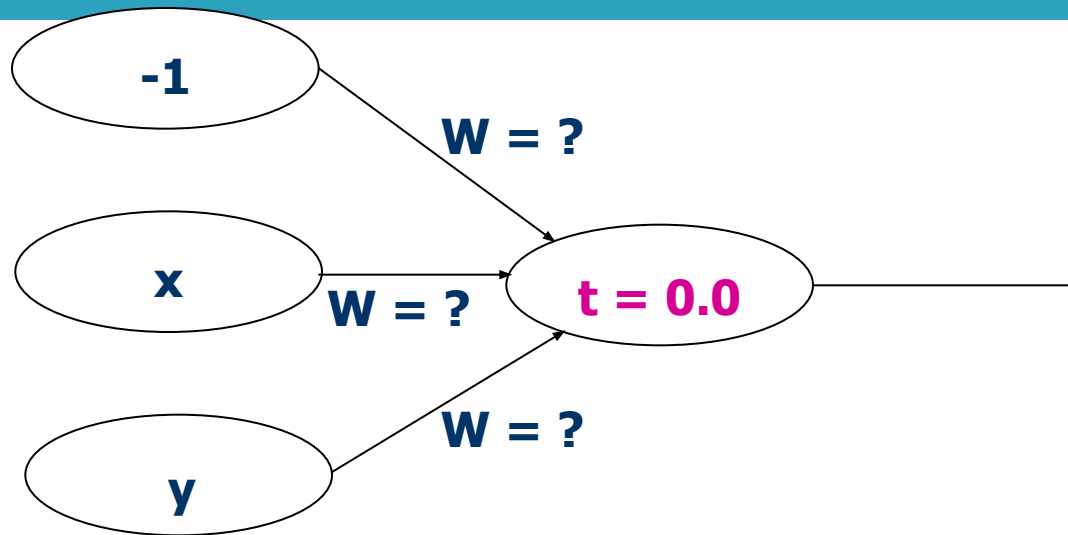


$$\text{Saída} = \begin{cases} 1 & \text{se } \sum w_i x_i > t \\ 0 & \text{caso contrário} \end{cases}$$

0

# Treinamento do Perceptron

29



## E lógico

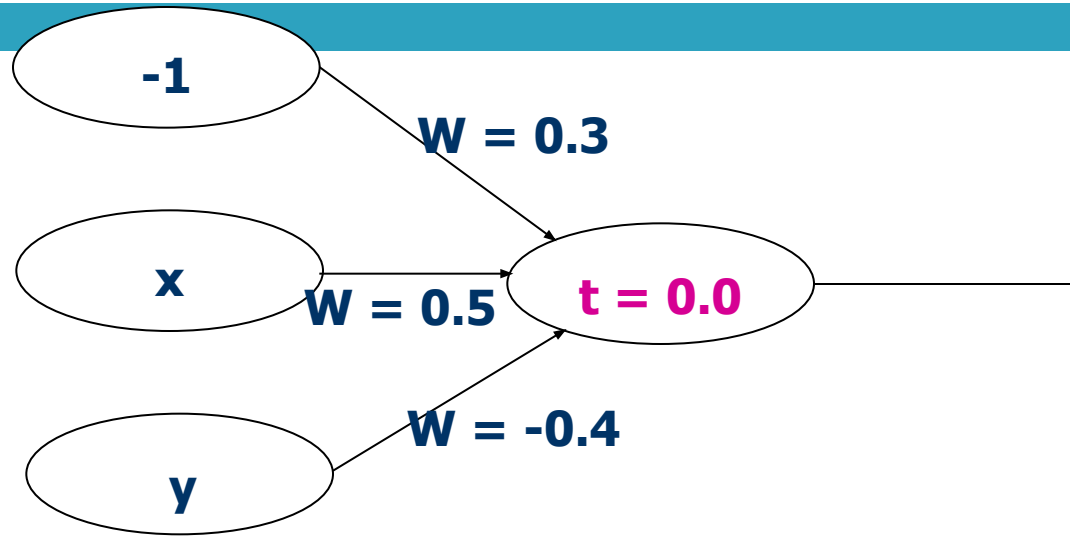
A	B	Saída
0	0	0
0	1	0
1	0	0
1	1	1

Quais os valores dos pesos?

Devem ser inicializados com valores randômicos.

# Treinamento do Perceptron

30



**E lógico**

A	B	Saída
0	0	0
0	1	0
1	0	0
1	1	1

-1	0	0	$(-1)*0.3 + 0*0.5 + 0*(-0.4) = -0.3$	0
-1	0	1	$(-1)*0.3 + 0*0.5 + 1*(-0.4) = -0.7$	0
-1	1	0	$(-1)*0.3 + 1*0.5 + 0*(-0.4) = 0.2$	1
-1	1	1	$(-1)*0.3 + 1*0.5 + 1*(-0.4) = -0.2$	0


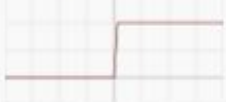

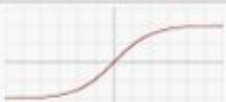





←

←

# Funções de ativação

## Funções de ativação mais comuns

- $a(t + 1) = u(t)$  (linear)
- $a(t + 1) = \begin{cases} 1, & \text{se } u(t) \geq \theta \\ 0, & \text{se } u(t) < \theta \end{cases}$  *threshold* ou limiar
- $a(t + 1) = 1 / (1 + e^{-\lambda u(t)})$  (sigmoid logística)
- $a(t + 1) = \frac{(1 - e^{-\lambda u(t)})}{(1 + e^{-\lambda u(t)})}$  (tangente hiperbólica)

Name	Plot	Equation
Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$

## Resumo das funções de ativação



# Funções de saída

## Função de saída

- Transforma estado de ativação de uma unidade em seu sinal de saída

$$y_i(t) = f_i(a_i(t))$$

- Geralmente é uma função identidade

# Valores de entrada e saída

Sinais de entrada e saída de uma RNA geralmente são números reais

- Números devem estar dentro de um intervalo
  - Tipicamente entre -1 e +1 ou 0 e 1
  - Codificação realizada pelo projetista da rede
- Técnica de codificação mais simples é a binária.

# Conexões



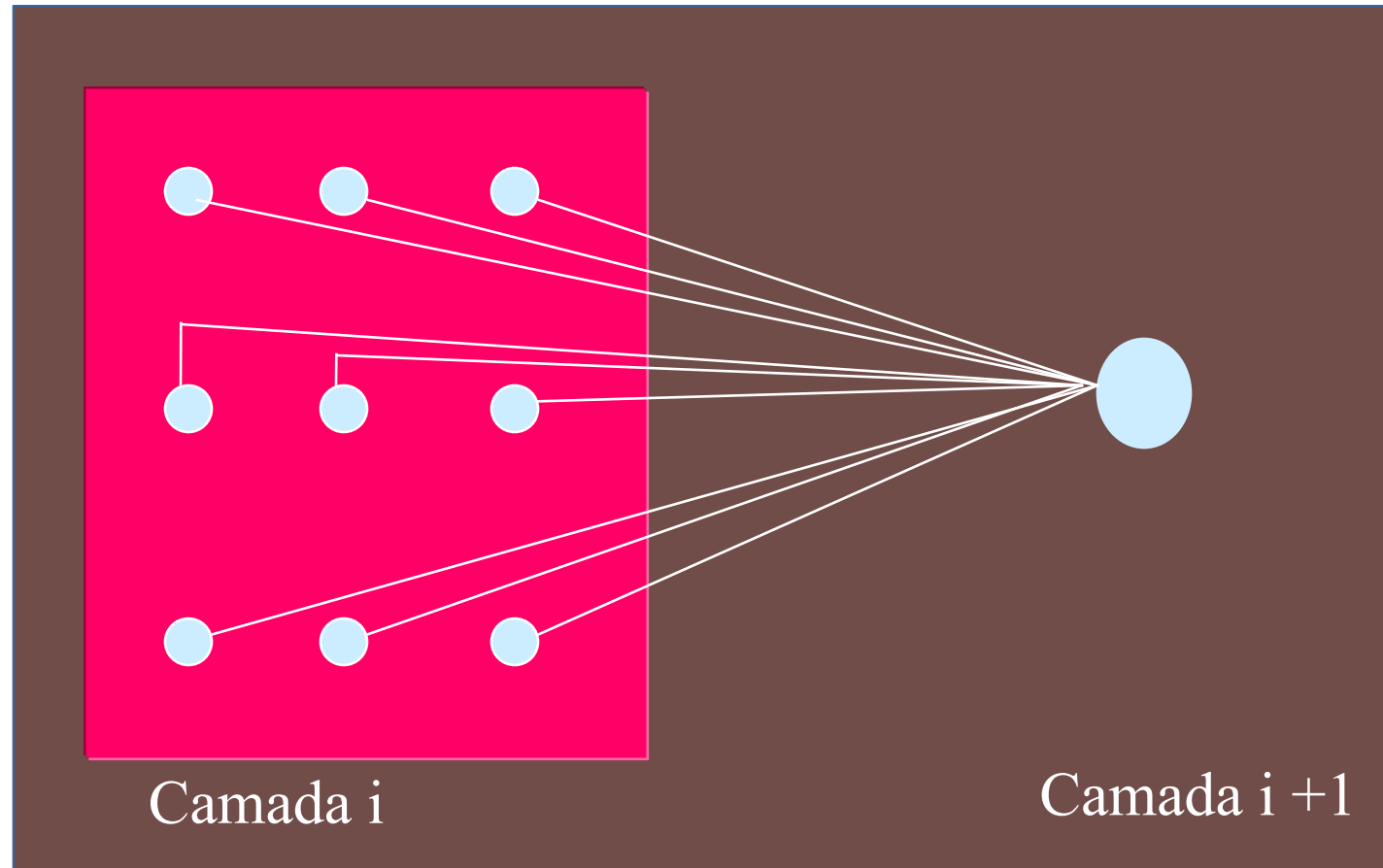
Definem como neurônios estão interligados

- Nós são conectados entre si através de conexões específicas.

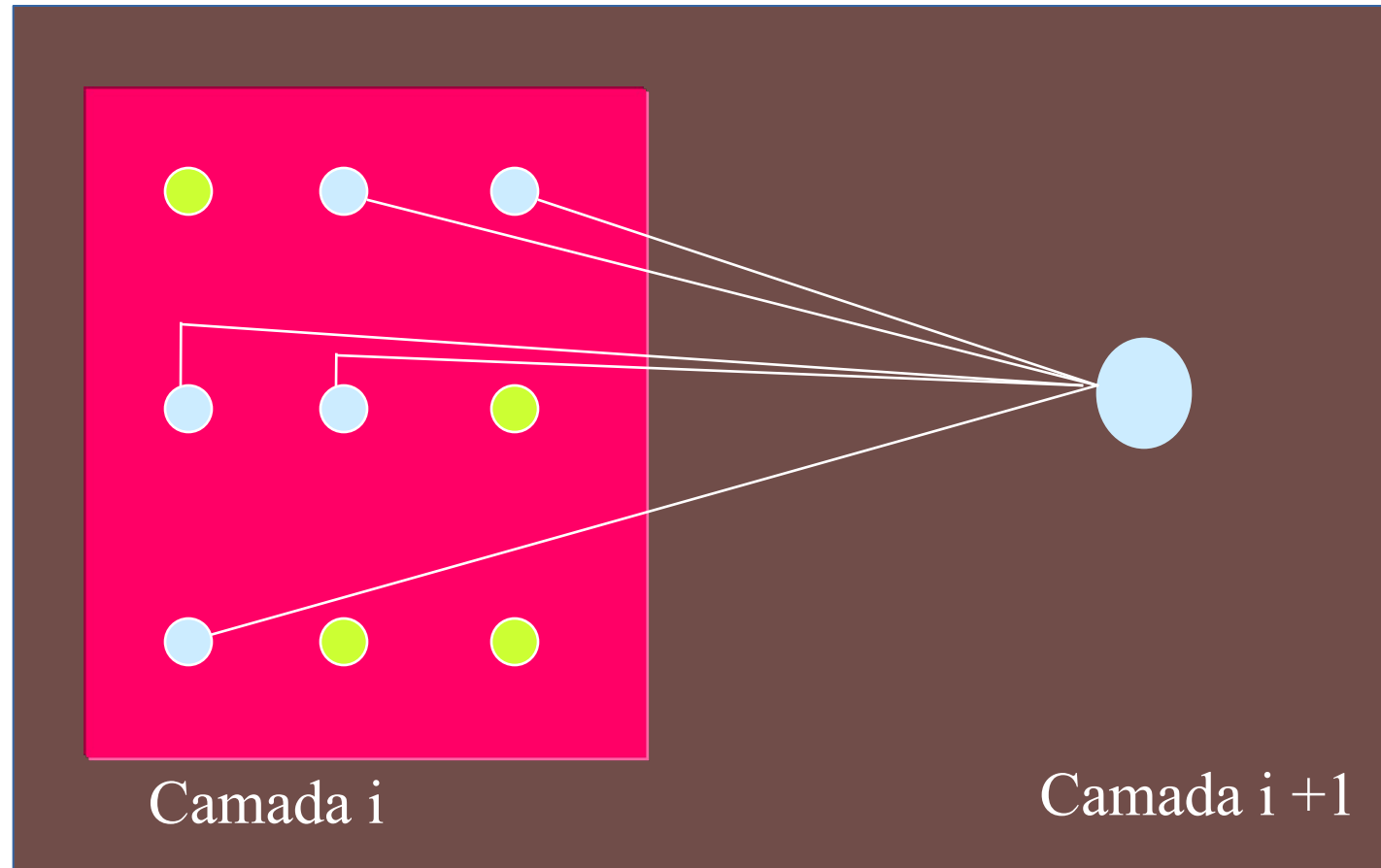
Codificam conhecimento da rede

- Uma conexão geralmente tem um valor de ponderamento ou **peso** associada a ela.

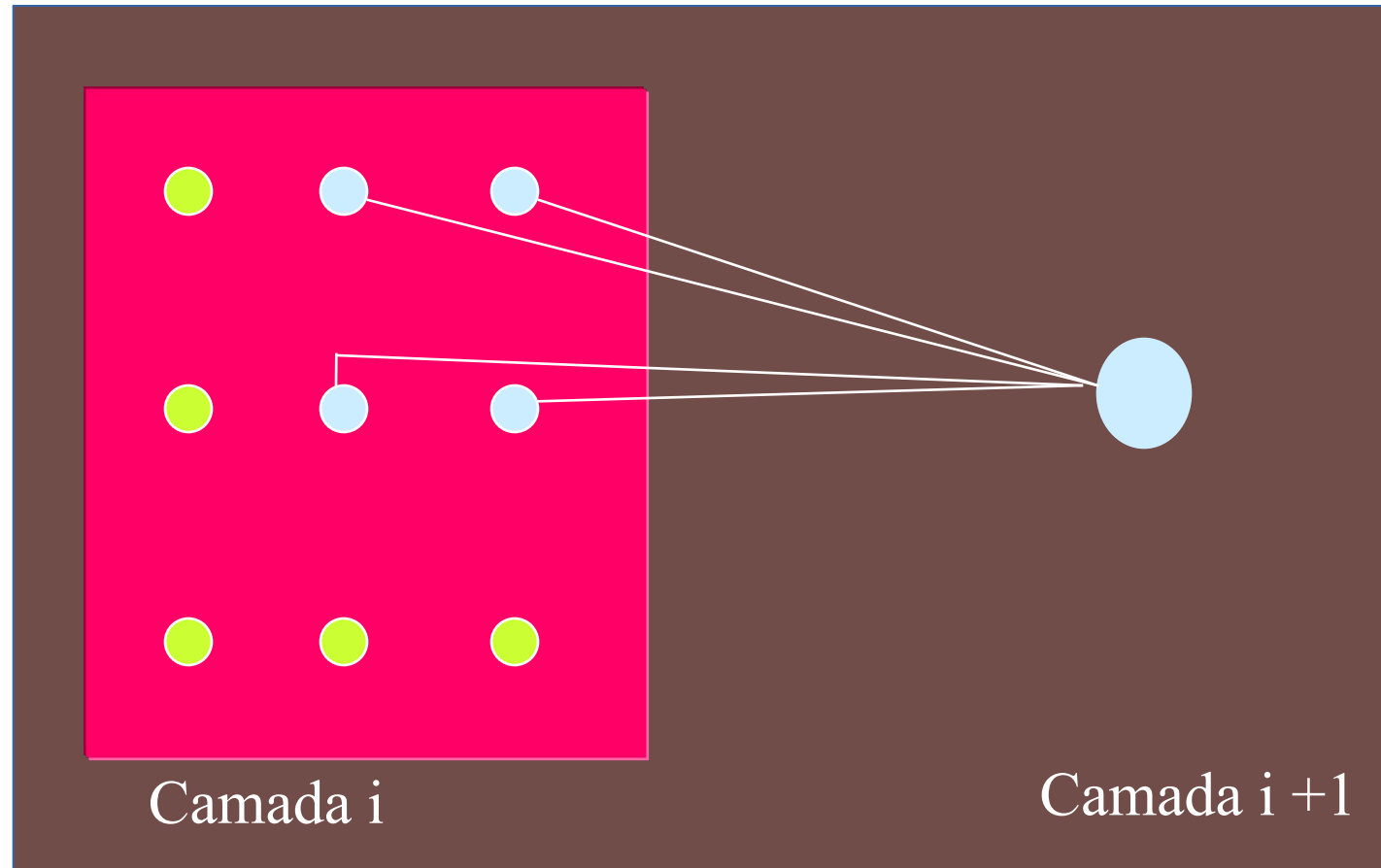
# Completamente conectada



# Parcialmente conectada



# Localmente conectada



# Topologia

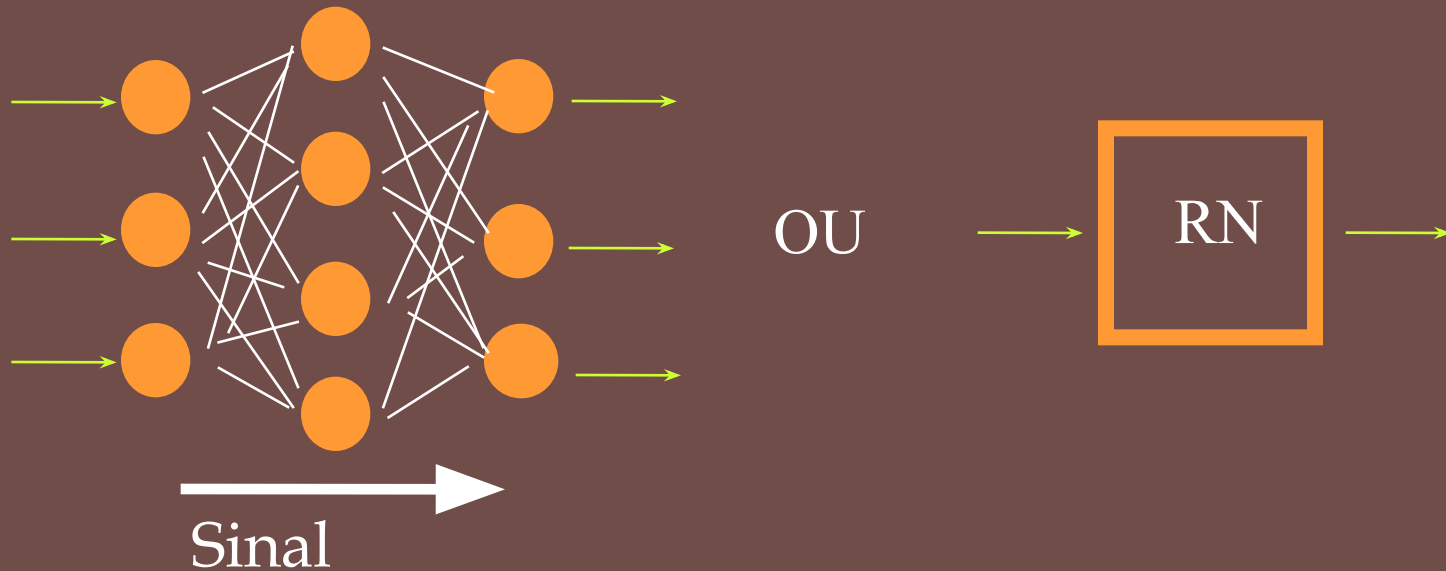


## Arranjo das conexões

- Redes feedforward
  - Não existem loops de conexões
- Redes recorrentes
  - Conexões apresentam loops
  - Mais utilizadas em sistemas dinâmicos
- Lattices
  - Matriz  $n$ -dimensional de neurônios

# Redes feedforward

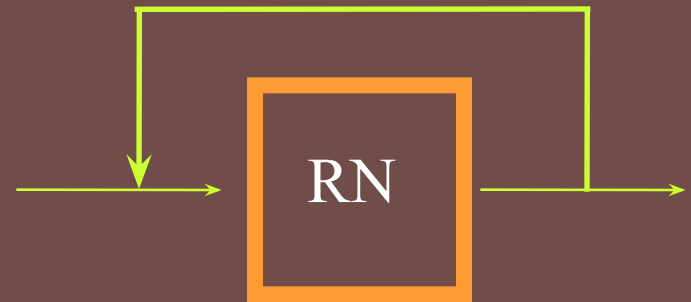
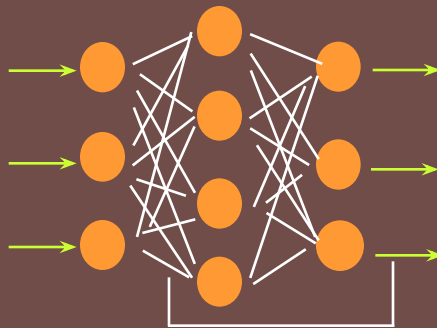
Sinais seguem em uma única direção





# Redes recorrentes

Possuem conexões ligando saída da rede a sua entrada



Podem aplicar entradas passadas e, conseqüentemente, processar seqüência de informações (no tempo ou espaço).

# Projeto de Redes Neurais



## Projeto de sistemas convencionais

- Formular modelo matemático a partir de observações do ambiente
- Validar o modelo com dados reais
- Construir o sistema utilizando o modelo

## Projeto de Redes Neurais

- Baseado apenas nos dados
- Exemplos para treinar uma rede devem ter padrões positivos e negativos

# Conjunto de dados



Tamanho depende da complexidade dos dados

- Quanto maior a complexidade, maior a quantidade necessária
- Pré-processamento dos dados
  - Dados numéricos
  - Presença de valores em todos os campos
  - Balanceamento entre classes

# Pré-processamento dos dados



## Estimativa de valores ausentes

- Média de todos os valores do mesmo campo
- Média entre anterior e posterior
- Criação de um novo valor

## Normalização de valores numéricos

- Normalizar cada campo individualmente
- Assegurar que todos os valores de um dado campo estejam dentro de um intervalo (Ex.  $[0.0, \dots, 1.0]$ )

# Projeto da rede



Escolha do modelo

Selecionar arquitetura adequada para a rede

- Número de camadas
- Número de nós da camada de entrada igual ao de atributos ou campos do vetor de entrada
  - Pré-processamento pode aumentar ou diminuir número de campos

# Aprendizado



Capacidade de aprender a partir de seu ambiente e melhorar sua performance com o tempo

Parâmetros livres de uma RNA são adaptados através de estímulos fornecidos pelo ambiente

- Processo iterativo de ajustes aplicado a sinapses e thresholds
- Idealmente, a RNA sabe mais sobre seu ambiente após cada iteração

# Aprendizado

RNA deve produzir para cada conjunto de entradas apresentado o conjunto de saídas desejado

- $w_{ik}(t+1) = w_{ik}(t) + \Delta w_{ik}(t)$

# Aprendizado



## Mecanismos de aprendizado

- Modificação de pesos ( $\Delta w_{ij}(t)$ ) associados às conexões
- Armazenamento de novos valores em conteúdos de memória
- Acréscimo e/ou eliminação de conexões/neurônios



# Aprendizado

## Algoritmos de aprendizado

- Conjunto de regras bem definidas para a solução de um problema de aprendizado
- Grande variedade
  - Cada um com suas vantagens
  - Diferem na maneira como ajuste  $\Delta w_{ik}(t)$  é realizado

## Paradigmas de aprendizado

- Diferem na maneira como RNA se relaciona com seu ambiente

# Aprendizado supervisionado

## Professor externo

- Possui conhecimento sobre ambiente
  - Representado por conjunto de pares  $(x, d)$
  - Geralmente, a rede não possui informações prévias sobre ambiente
- Parâmetros da rede são ajustados por  $(x, d)$
- Rede procura emular professor

# Algoritmo de Aprendizagem

58

**Treino** : Apresenta o conjunto de treinamento para a RN. No caso da função E, o conjunto de treino consiste de 4 conjuntos de entradas (i.e.  $[0,0]$ ,  $[0,1]$ ,  $[1,0]$ ,  $[1,1]$ )

**Erro** : O valor do erro é a diferença entre o valor esperado na saída e o valor obtido. Por exemplo, se o valor esperado de saída da RN fosse 0, e o valor obtido 1, então o erro = -1.

# Algoritmo de Aprendizagem

59

**Valor esperado,  $T$**  : Quando se treina uma rede, deve-se apresentar os valores de entrada com o respectivo valor esperado de saída. Por exemplo, para o valor  $[1,1]$  e a função AND, o valor esperado será 1.

**Saída,  $O$**  : O valor de saída do neurônio.

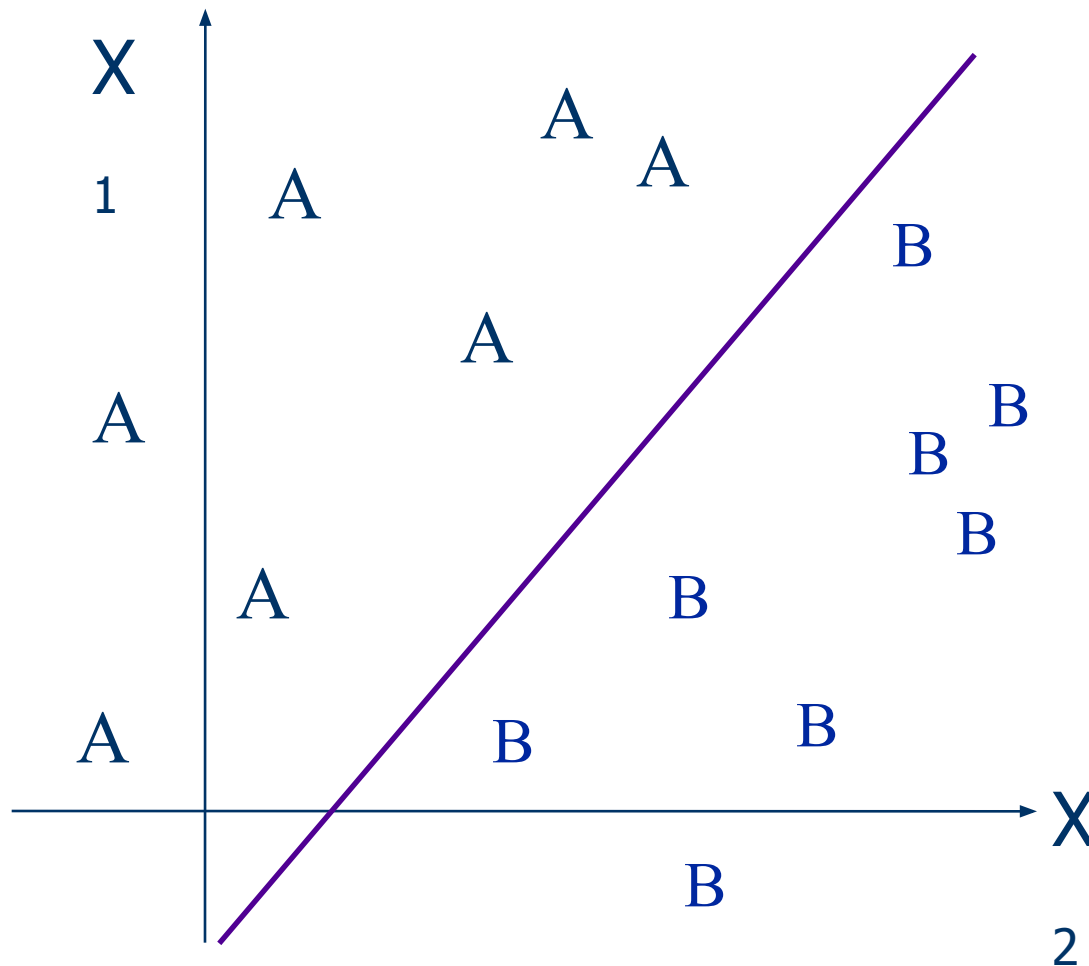
**$I_j$**  : Entrada apresentada ao neurônio.

**$W_j$**  : Peso do neurônio de entrada ( $I_j$ ) para o neurônio de saída.

**$LR$**  : Taxa de aprendizagem. Ela dita o quão rapidamente a rede irá convergir. Este valor é setado através de experimentos. O valor default é de 0,1.

# Separabilidade linear

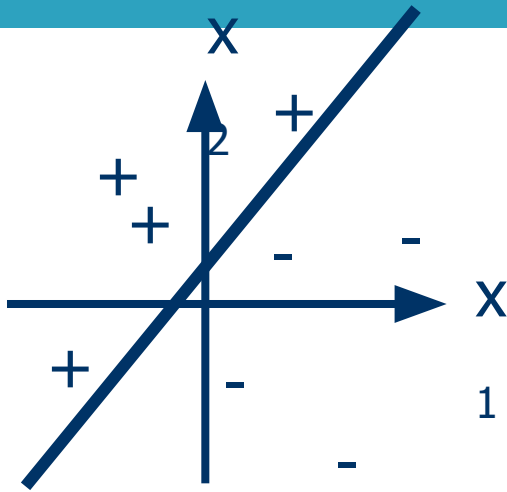
53



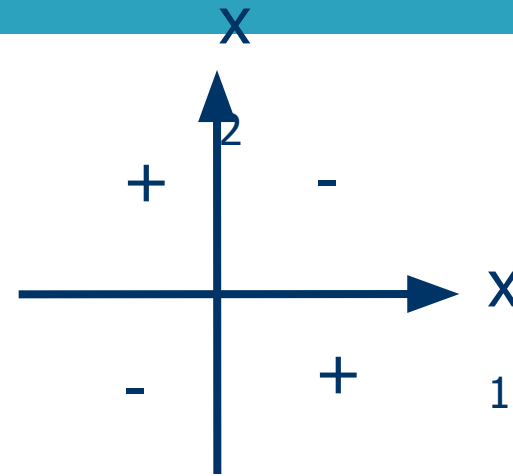
Fronteira  
de Decisão

# Superfície de Decisão de um Perceptron

61



Linearmente separáveis


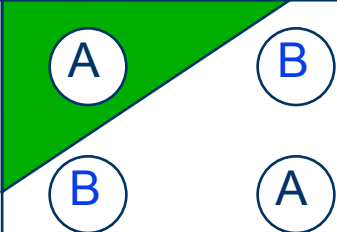
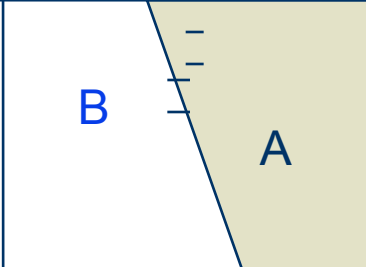
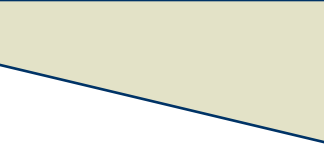
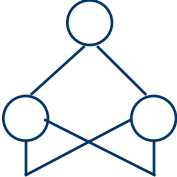
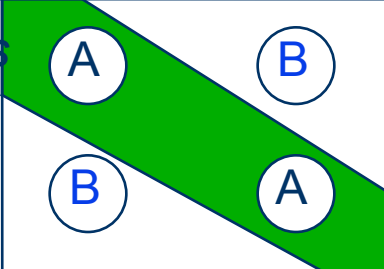
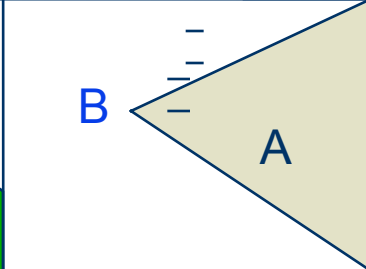
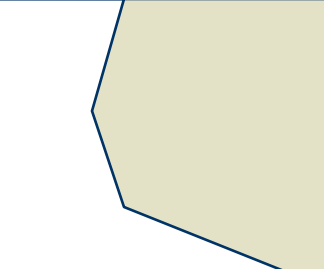
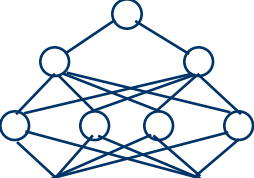
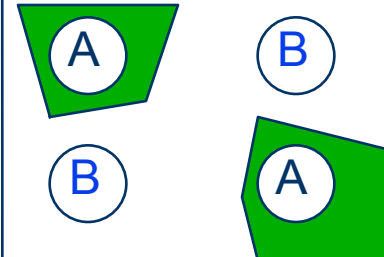
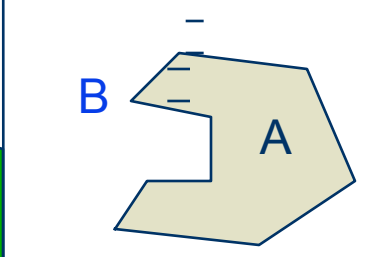
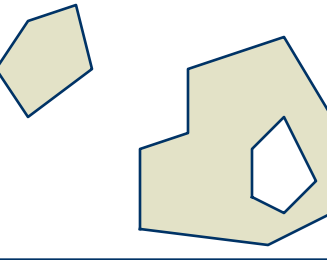


Não-Linearmente separáveis

- Perceptron não opera com funções que não sejam linearmente separáveis (e.g. XOR).

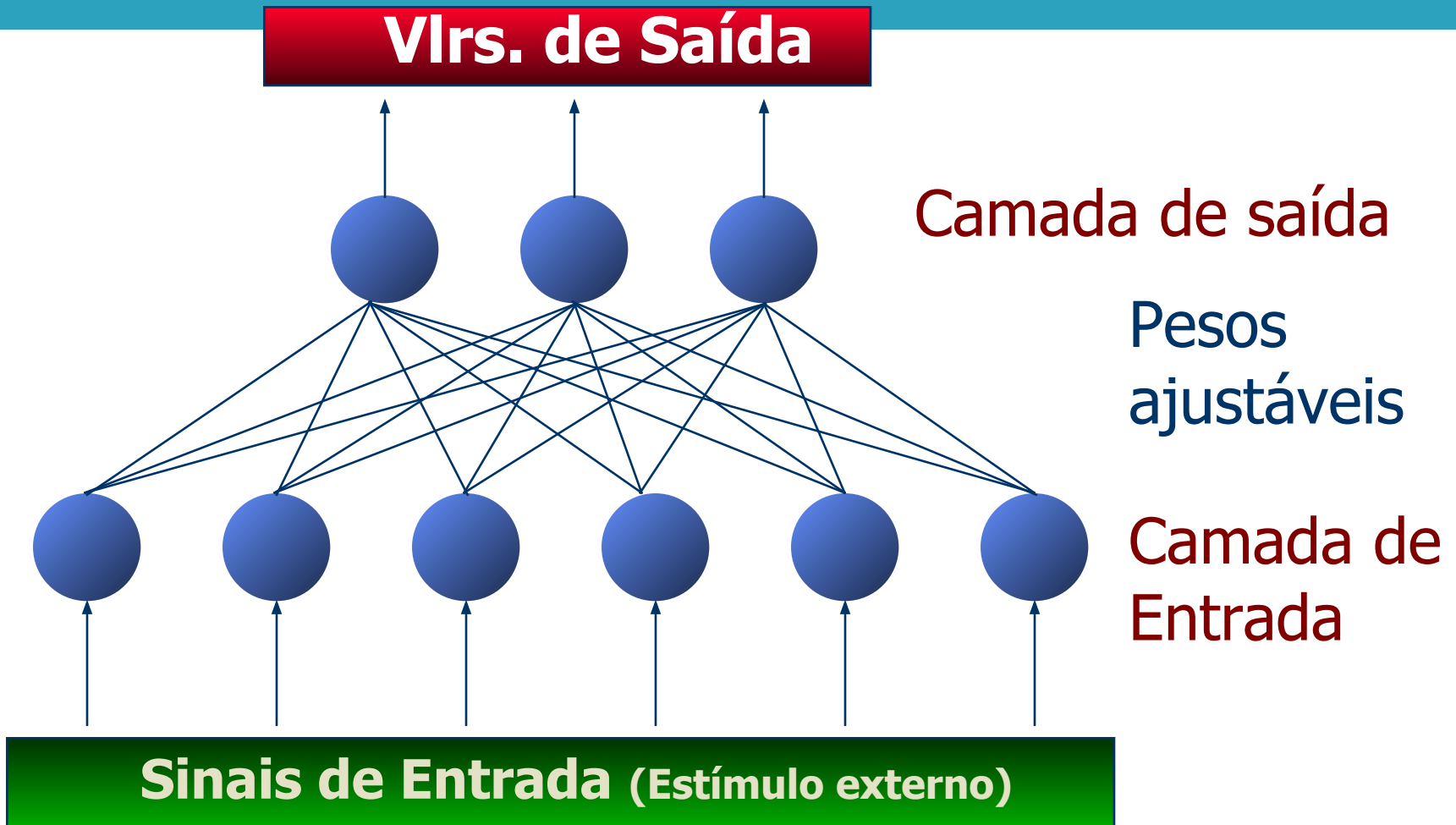
# Problemas não linearmente separáveis

55

<i>Estrutura</i>	<i>Tipos de Regiões</i>	<i>Exclusive-OR Problem</i>	<i>Classes com Regiões misturadas</i>	<i>Regiões em geral</i>
<i>Uma camada</i> 	<i>Plano</i>			
<i>Duas camadas</i> 	<i>Regiões convexas abertas ou fechadas</i>			
<i>Três camadas</i> 	<i>Arbitrárias (Complexidade Limitada pelo No. de Nós)</i>			

# Perceptron Multi-camadas

56





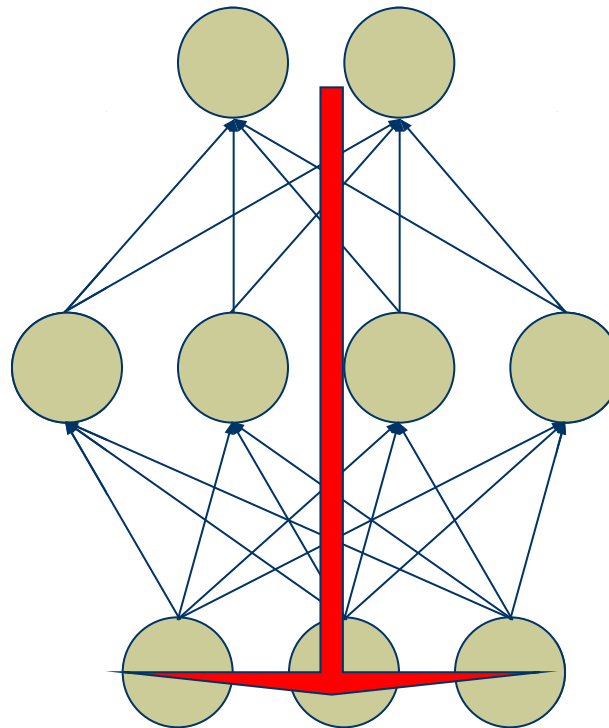
# Algoritmo Backpropagation

57

- Pode teoricamente mapear “qualquer” conjunto de entradas e saídas.
- Aprende a resolver problemas linearmente INSEPARÁVEIS.

# Algoritmo Backpropagation

65



“ativação”

“erros”

# Entradas para Backpropagation

- ◆ Entradas da rede podem ser de dois tipos:
  - Binárias (0 não considerar, 1 considerar).
  - Bipolares (-1 não considerar, +1 considerar).
- ◆ Duas funções de ativação pode ser usadas:
  - Funções Sigmoidais para entradas binárias.
  - Funções Sigmoidais Bipolares para entradas bipolares.

# Entradas para Backpropagation

66

## ◆ Entradas Binárias:

- Um valor maior ou igual a 0.5 é considerado.
- Um valor menor que 0.5 não é considerado.

## ◆ Entradas Bipolares:

- Um valor maior ou igual a 0 é considerado.
- Um valor menor que 0 não é considerado.

# Algoritmo

67

## Definições :

- o par de treinamento (E,S) corresponde à um conjunto de entradas (E) e a sua respectiva saída desejada (S);
- o erro R é definido como: Resposta Desejada - Resposta Obtida ( $D - O$ );
- a taxa de aprendizado A é uma constante positiva, que corresponde à velocidade do aprendizado.
- $\epsilon$  indica o erro aceito no treinamento da rede.

---

1. Iniciar todas as conexões com pesos aleatórios;

2. Enquanto o erro  $R > \epsilon$  faça

Para cada par de treinamento (E,S), faça:

Calcular a resposta obtida O;

Calcular R;

Se o erro R não for satisfatoriamente pequeno

então atualizar pesos:  $W_{\text{novo}} := W_{\text{anterior}} + A$

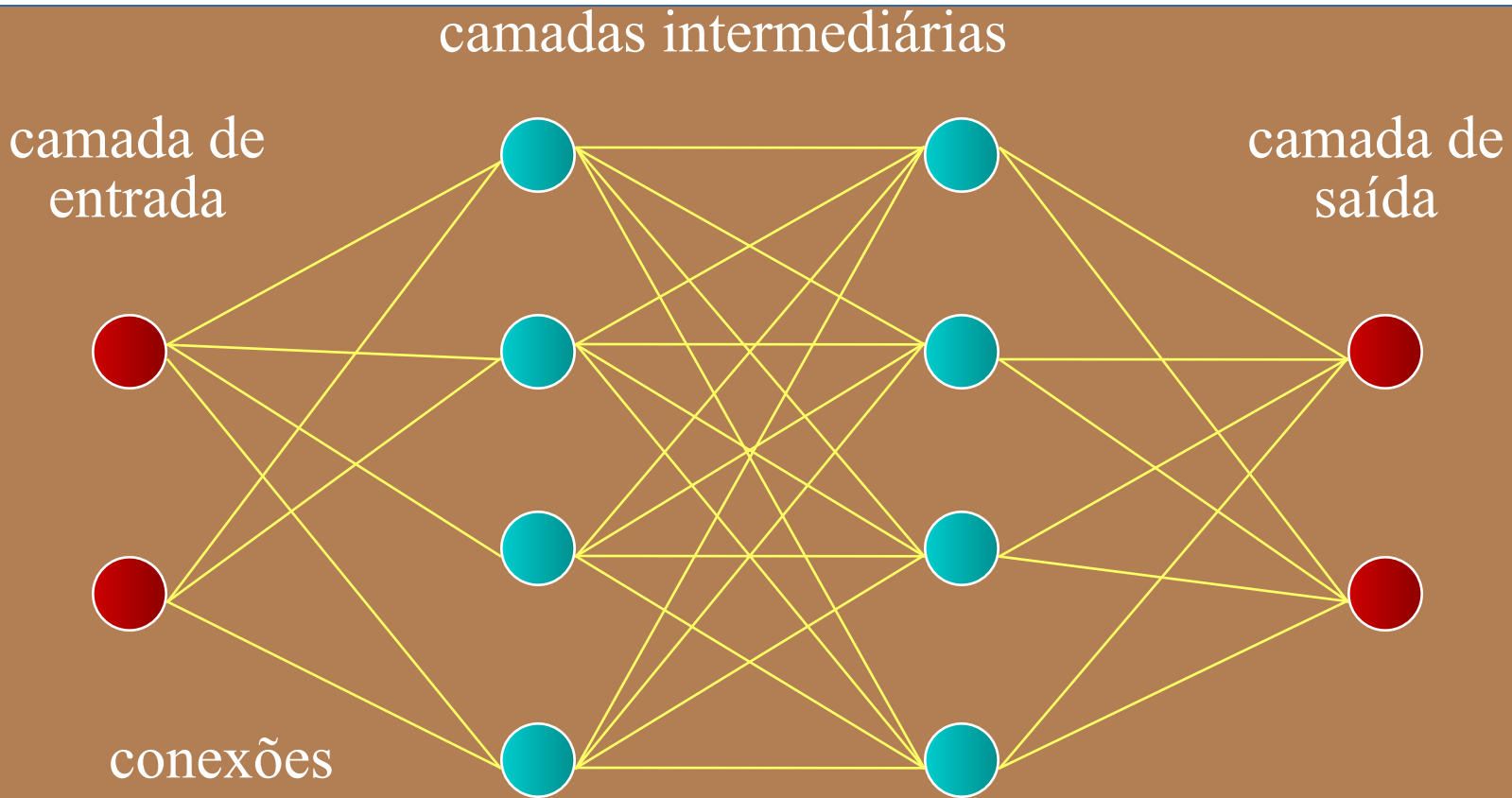
# Backpropagation

68

- ◆ A fase de treinamento pode ser lenta (1 000-10 000 iterações)
- ◆ Porém, o uso da rede após o treinamento é rápido.

# Arquitetura de uma Rede Neural Multi-camadas

70



# Aplicações

64

- As propriedades da rede neural definem onde elas são úteis:
  - Podem aprender mapeamentos complexos a partir de entradas e saídas.
  - Difícil análise: inapropriado para aplicações críticas que envolvem aspectos de segurança;