
UNIVASF
FUNDAÇÃO UNIVERSIDADE FEDERAL
DO VALE DO SÃO FRANCISCO

# Redes Neurais Artificial

# Inteligência Artificial

**Professor**: Rosalvo Ferreira de Oliveira Neto

# Estrutura

1. Definições
2. Histórico
3. Conceitos Básicos
4. Aprendizado em RNA
5. Exemplo de Aprendizado com Perceptron

# Definições

Redes Neurais Artificiais são técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência.

# Definições

- Redes Neurais Artificiais (RNA) são modelos de computação com propriedades particulares

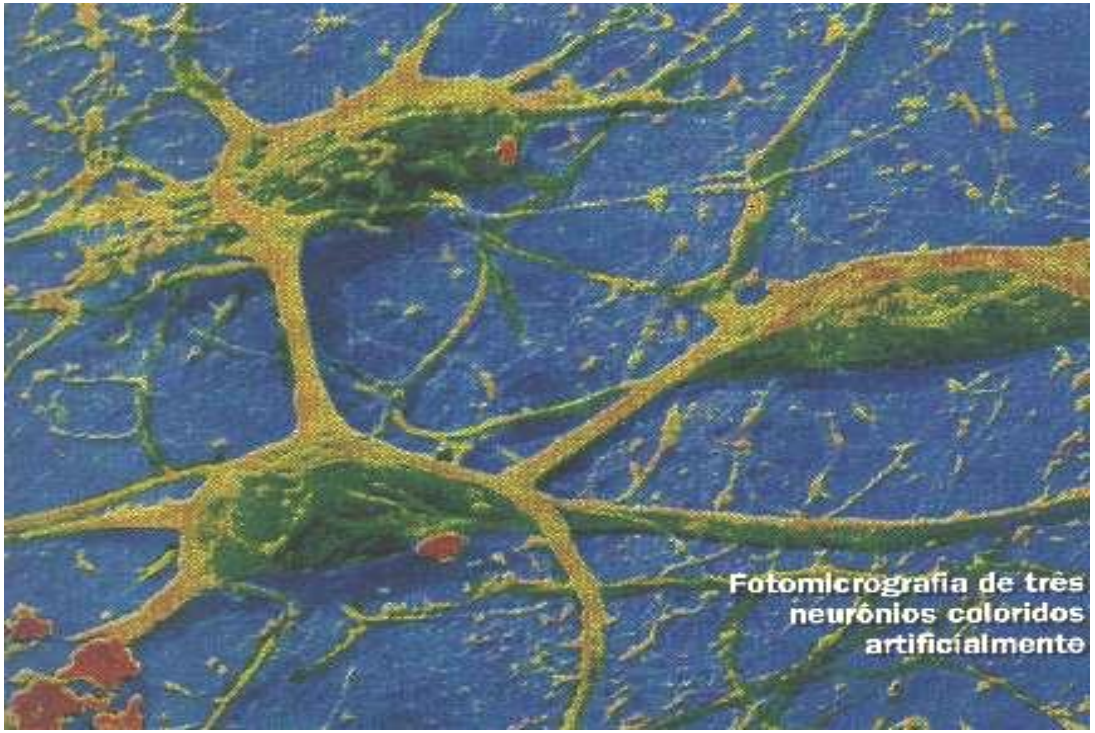
  - Capacidade de se adaptar ou aprender
  - Generalizar
  - Agrupar ou organizar dados

# Definições

- Modelos inspirados no cérebro humano
  - Compostas por várias unidades de processamento (“neurônios”)
  - Interligadas por um grande número de conexões (“sinapses”)
- Eficientes onde métodos tradicionais têm se mostrado inadequados

# Inspiração biológica

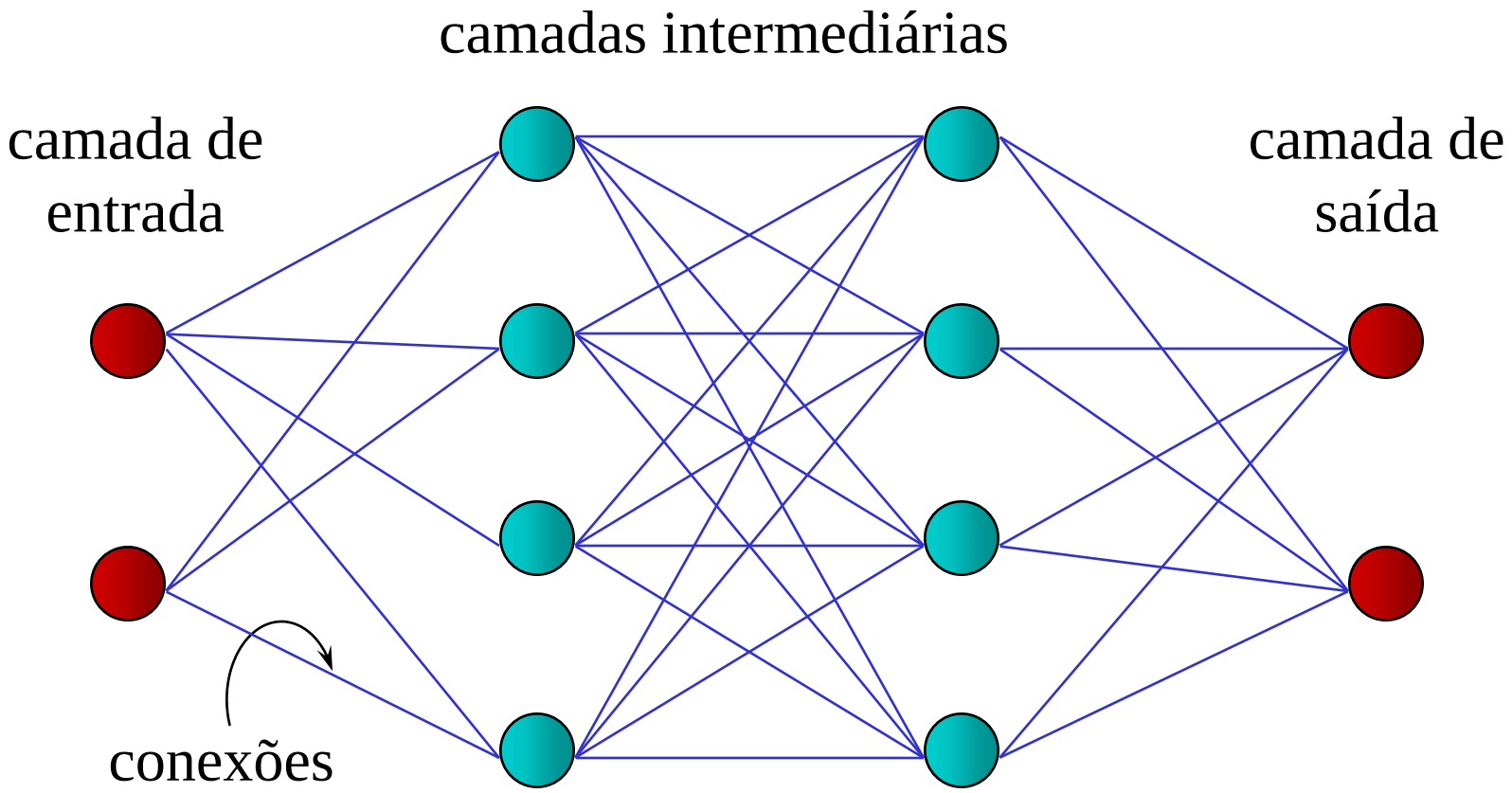
Em média, cada neurônio forma entre mil e dez mil sinapses. O cérebro humano possui cerca de $10^{11}$ neurônios, e o número de sinapses é maior que $10^{14}$, possibilitando a formação de redes muito complexa.



# Histórico

- 1940-1950: início das Redes neurais com McCulloch e Pitts
- 1950-1960: Perceptron de aprendizado supervisionado.
- 1960-1970: Pouco avanço
- 1980-2000: Redes simétricas e o Backpropagation

# Redes Neurais Artificiais



# Características das RNAs

- Aprendem através de exemplos
- Adaptabilidade
- Capacidade de generalização
- Tolerância a falhas
- Implementação rápida

# Conceitos básicos

- Estrutura geral das RNA

1. Unidades de processamento (neurônios)
2. Conexões
3. Topologia

# Conceitos básicos

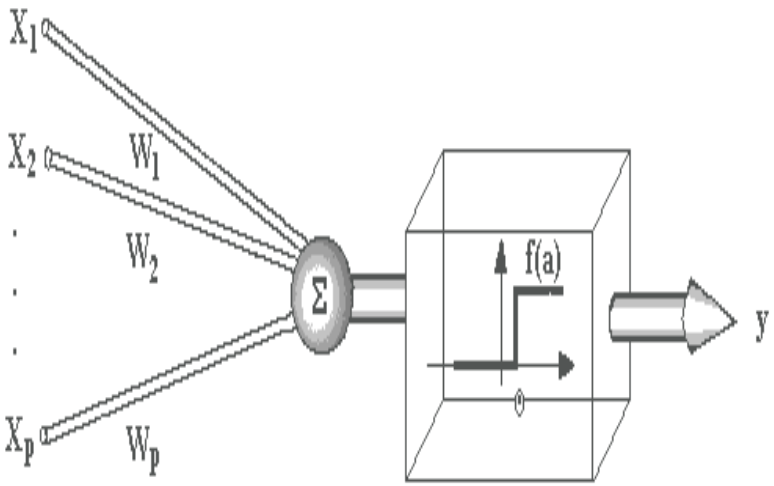
## Unidades de processamento $n_i$ (nós)

1. Estado de ativação $a_i$
2. Função de ativação $F_i$
3. Função de saída $f_i$

# Conceitos básicos

## Unidades de processamento

Objetivo: receber entradas de conjunto de unidades A, computar função sobre entradas e enviar resultado para conjunto de unidades B



# Conceitos básicos

## Unidades de processamento - Estado de ativação

- Representa o estado dos neurônios da rede
- Pode assumir valores
  - Binários (0 e 1)
  - Bipolares (-1 e +1)
  - Reais
- Definido através de funções de ativação

# Conceitos básicos

## Unidades de processamento - Funções de ativação

• Processa conjunto de entradas recebidas e o transforma em estado de ativação – Atualiza o estado de ativação do neurônio

•Funções de ativação típicas envolvem:

- Adições
- Comparações
- Transformações matemáticas

# Conceitos básicos

- 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)

# Conceitos básicos

- 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

# Conceitos básicos

- **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

# Conceitos básicos

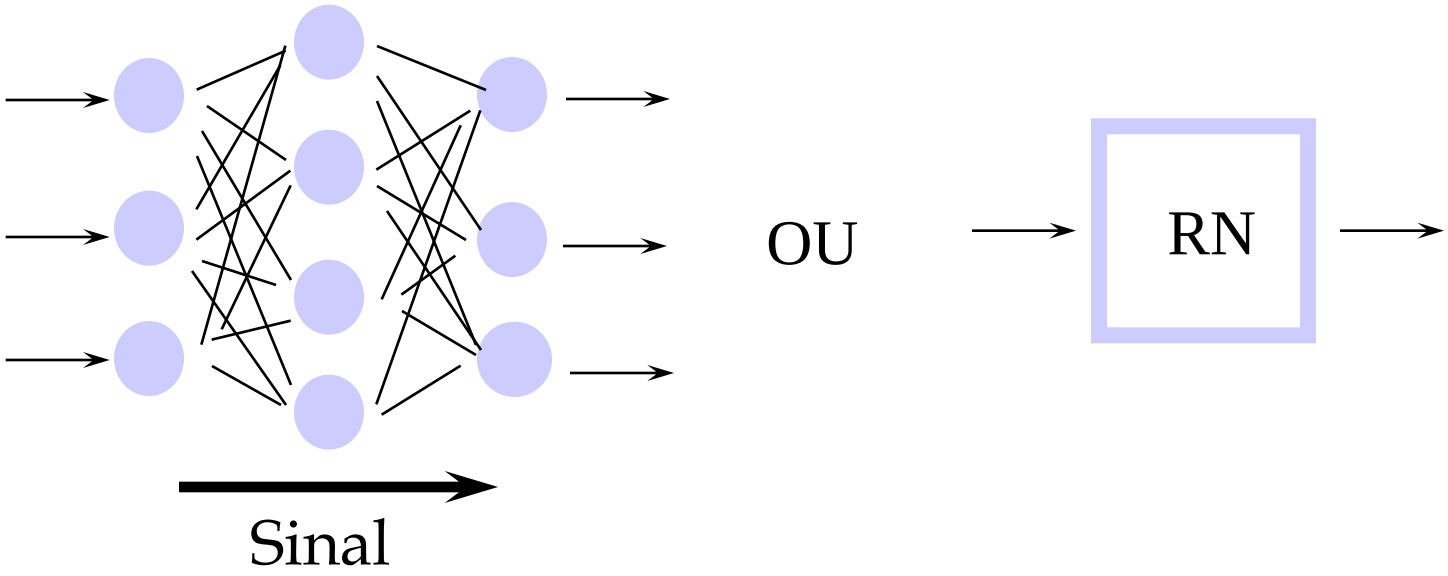
- Tipos de conexões ($w_{ik}(t)$)
  - Excitatória: ($w_{ik}(t) > 0$)
  - Inibitória: ($w_{ik}(t) < 0$)
  - Conexão inexistente: ($w_{ik}(t) = 0$)
- Número de conexões de um nó
  - Fan-in: número de conexões de entrada
  - Fan-out: número de conexões de saída

# Conceitos básicos

- **Topologia**
  - Número de camadas
    - Uma camada (Ex Perceptron, Adaline)
    - Multi-camadas (Ex MLP)
      - Completamente conectada
      - Parcialmente conectada
      - Localmente conectada

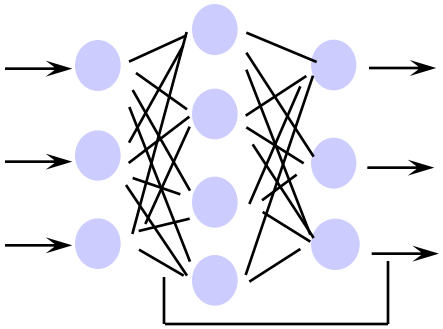
# Redes feedforward

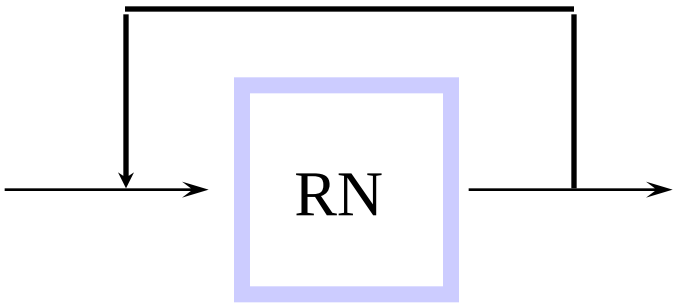
- Sinais seguem em uma única direção



# Redes recorrentes

- Possuem conexões ligando saída da rede a sua entrada





# Aprendizado em RNAs

- 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 em RNAs

- 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 em RNAs

- 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

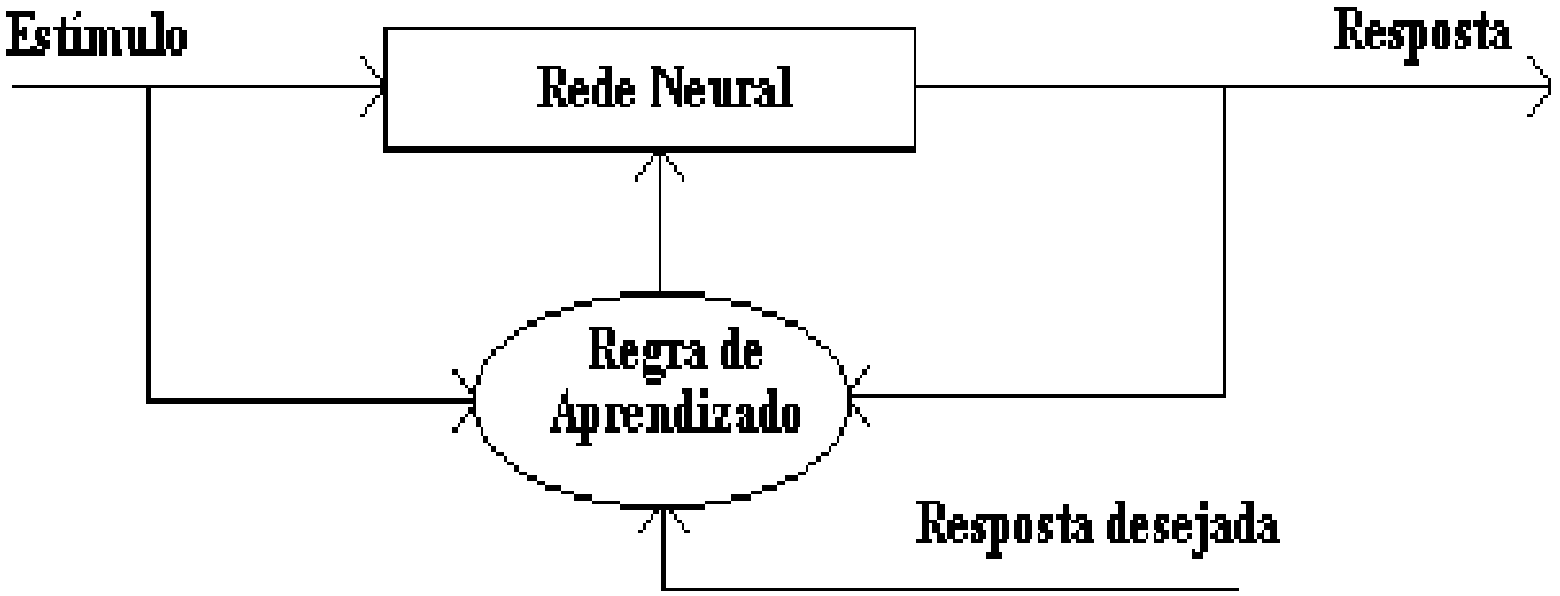
# Tipos de Aprendizado em RNAs

- Supervisionado
- Não Supervisionado

# 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

# Aprendizado supervisionado

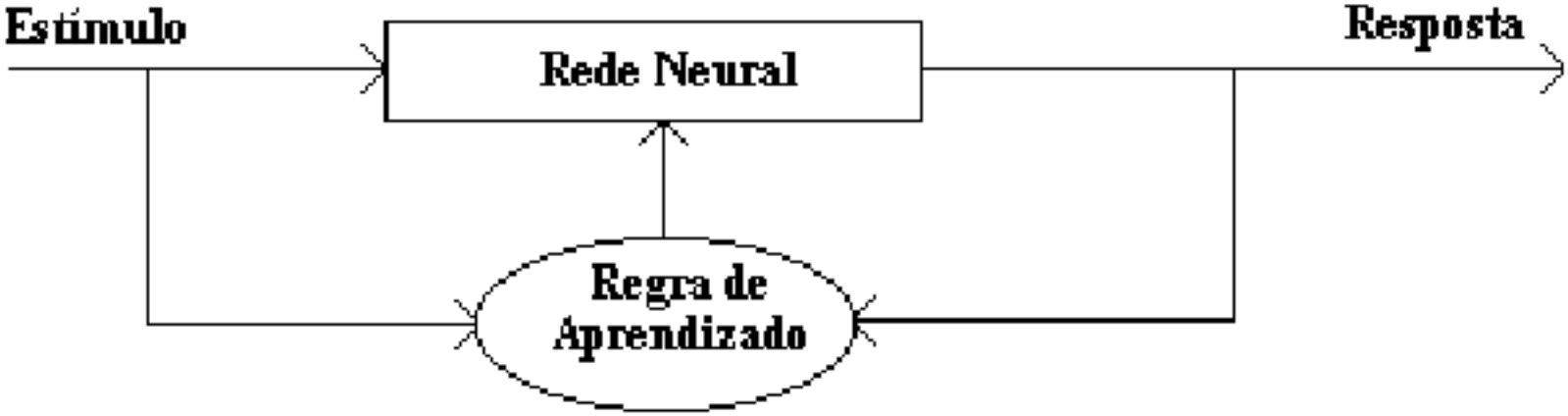



Aprendizado Supervisionado

# Aprendizado não supervisionado

- Não tem crítico ou professor externo
- Extração de características estatisticamente relevantes
  - Cria classes automaticamente

# Aprendizado não supervisionado




Aprendizado Não-supervisionado

# Aprendizado por correção de erro

- Regra Delta (Widrow e Hoff 1960)
- Erro: $e_k(t) = d_k(t) - y_k(t)$
- Minimizar função de custo baseada em $e_k(t)$
- Função de custo
  - $c(t) = -1/2 \text{å} e^2_k(t)$
  - Aprendizado atinge solução estável quando os pesos não precisam mudar muito

# Aprendizado por correção de erro

- Após seleção da função de custo, aprendizado se torna um problema de otimização
  - RNA é otimizada pela minimização de c(t) com respeito aos pesos da rede
- Modelo matemático:
  - $\Delta w_{ik}(t) = \eta e_k(t)\, x_i(t)$

# Aprendizado por correção de erro

- Taxa de aprendizado ($\eta$)
  - $0 < h < 1$
  - Taxas pequenas
    - Média das entradas anteriores
    - Estimativas estáves de peso
    - Aprendizado lento
  - Taxas grandes
    - Captação de mudanças no processo
    - Instabilidade
  - Taxas variáveis

# Perceptrons

- Desenvolvido por Rosemblat, 1958
- Rede mais simples para classificação de padrões linearmente separáveis
- Utiliza modelo de McCulloch-Pitts como neurônio

# Perceptrons

- Estado de ativação
  - 1 = ativo
  - -1 = inativo

- Função de ativação
  - $a_i(t+1) = \begin{cases} -1, & \text{se } u_i(t) < 0 \\ +1, & \text{se } u_i(t) \geq 0 \end{cases}$

$$u = \sum_{j=1}^{N} x_j w_j - \theta$$

# Perceptrons

- Função de saída = função identidade
- Duas camadas
  - Camada de pré-processamento
    - M máscaras fixas para extração de características
  - Camada de discriminação
    - Uma unidade de saída para discriminar padrões de entrada
    - Pesos determinados através de aprendizado

# Perceptrons

- Treinamento
  - Supervisionado
  - Correção de erro
    - $\Delta w_{ij} = \eta x_i (d_j - y_j)$ $(d \neq y)$
    - $\Delta w_{ij} = 0$ $(d = y)$
- Teorema de convergência: se é possível classificar um conjunto de entradas, uma rede Perceptron fará a classificação

# Algoritmo de treinamento

1) Iniciar todas as conexões com $w_{ij} = 0$;

2) Repita

Para cada par de treinamento (X, d)

Calcular a saída y

Se ($d \neq y$)

Então

Atualizar pesos dos neurônios

Até o erro ser aceitável

# Algoritmo de teste

1) Apresentar padrão X a ser reconhecido

2) Calcular a saída y

3) Se (y=-1)

Então

X $\in$ classe 0

Senão

X $\in$ classe 1

# Exemplo

Dada uma rede do tipo Perceptron formada por um neurônio com três terminais de entrada, utilizando pesos iniciais $w_0 = 0.4$, $w_1 = -0.6$ e $w_2 = 0.6$, limiar $\theta = 0.5$ e uma taxa de aprendizado $= 0.4$, responda os itens abaixo:

a) Ensinar a rede a gerar a saída -1 para o padrão 001 e a saída +1 para os padrão 110

b) A que classe pertencem os padrões 111, 000, 100 e 011?

# Exemplo 1: resposta a

a) Treinar a rede

a.1) Para o padrão 001 (d = -1)

Passo 1: definir a saída da rede

$u = 0(0.4) + 0(-0.6) + 1(0.6) - 1(0.5) = 0.1$

$y = u = +1$ (uma vez $0.1 \neq 0$)

Passo 2: atualizar os pesos

$w_0 = 0.4 + 0.4(0)(-1 - (+1)) = 0.4$

$w_1 = -0.6 + 0.4(0)(-1 - (+1)) = -0.6$

$w_2 = 0.6 + 0.4(1)(-1 - (+1)) = -0.2$

$w_3 = 0.5 + 0.4(-1)(-1 - (+1)) = 1.3$

# Exemplo 1: resposta a

a) Treinar a rede

a.2) Para o padrão 110 (d = 1)

Passo 1: definir a saída da rede

$u = 1(0.4) + 1(-0.6) + 0(-0.2) -1(1.3) = -1.5$

$y = u = -1$ (uma vez $-1.5 < 0$)

Passo 2: atualizar pesos

$w_0 = 0.4 + 0.4(1)(1 - (-1)) = 1.2$

$w_1 = -0.6 + 0.4(1)(1 - (-1)) = 0.2$

$w_2 = -0.2 + 0.4(0)(1 - (-1)) = -0.2$

$w_2 = 1.3 + 0.4(-1)(1 - (-1)) = 0.5$

# Exemplo 1: resposta a

a) Treinar a rede

a.3) Para o padrão 001 (d = -1)

Passo 1: definir a saída da rede

$u = 0(1.2) + 0(0.2) + 1(-0.2) -1(0.5) = -0.7$

$y = u = -1$ (uma vez $-0.7 < 0$)

Passo 2: atualizar pesos

Como d = y, os pesos não precisam ser modificados

# Exemplo 1: resposta a

a) Treinar a rede

a.4) Para o padrão 110 (d = 1)

Passo 1: definir a saída da rede

$u = 1(1.2) + 1(0.2) + 0(-0.2) -1(0.5) = 0.9$

$y = u = 1$ (uma vez $0.9 \geq 0$)

Passo 2: atualizar pesos

Como d = y, os pesos não precisam ser

# Exemplo 1: resposta b

b) Testar a rede

b.1) Para o padrão 111

$u = 1(1.2) + 1(0.2) + 1(-0.2) -1(0.5) = 0.7$

$y = u = 1$ (porque $0.7 \geq 0$) ) -> classe 1

b.2) Para o padrão 000

$u = 0(1.2) + 0(0.2) + 0(-0.2) -1(0.5) = -0.5$

$y = u = -1$ (porque $-0.5 < 0$) -> classe 0

# Exemplo 1: resposta b

b) Testar a rede

b.3) Para o padrão 100

$u = 1(1.2) + 0(0.2) + 0(-0.2) + 1(-0.5) = 0.7$

$y = u = 1$ (porque $0.7 \geq 0$) -> classe 1

b.4) Para o padrão 011

$u = 0(1.2) + 1(0.2) + 1(-0.2) - 1(0.5) = -0.5$

$y = u = -1$ (porque $-0.5 < 0$) -> classe 0

# Aplicação das RNAs

- Classificação de padrões
- Clustering/categorização
- Aproximação de funções
- Previsão
- Otimização
- etc...


UNIVASF
FUNDAÇÃO UNIVERSIDADE FEDERAL
DO VALE DO SÃO FRANCISCO