SICO70
SISTEMAS INTELIGENTES 2

Aula 04 - Perceptron Multicamadas
(*Multilayer Perceptron* - MLP)

Prof. Rafael G. **Mantovani**

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Apucarana - PR, Brasil

Universidade Tecnológica Federal do Paraná (UTFPR)
Especialização em Ciência de Dados

Roteiro

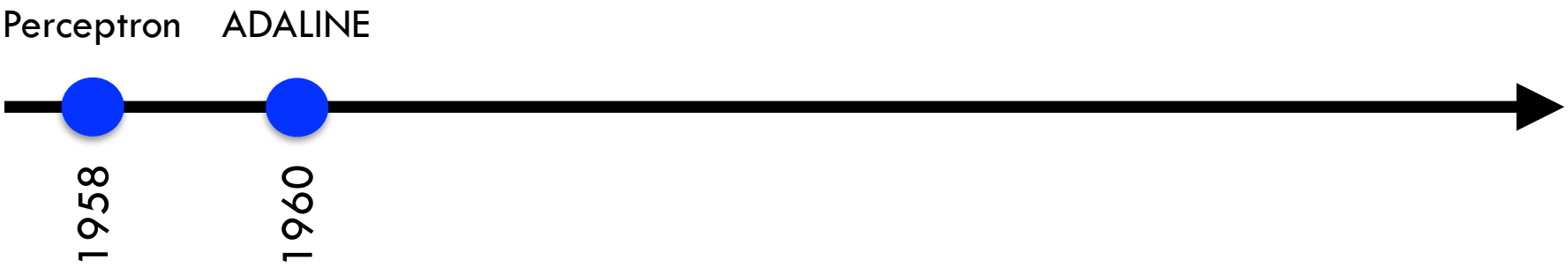
1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Roteiro

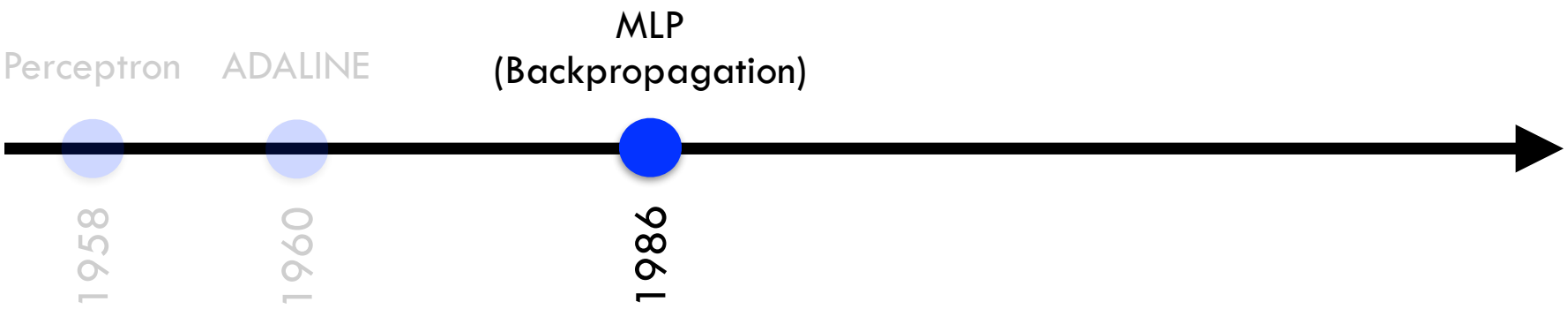
1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Introdução

Introdução

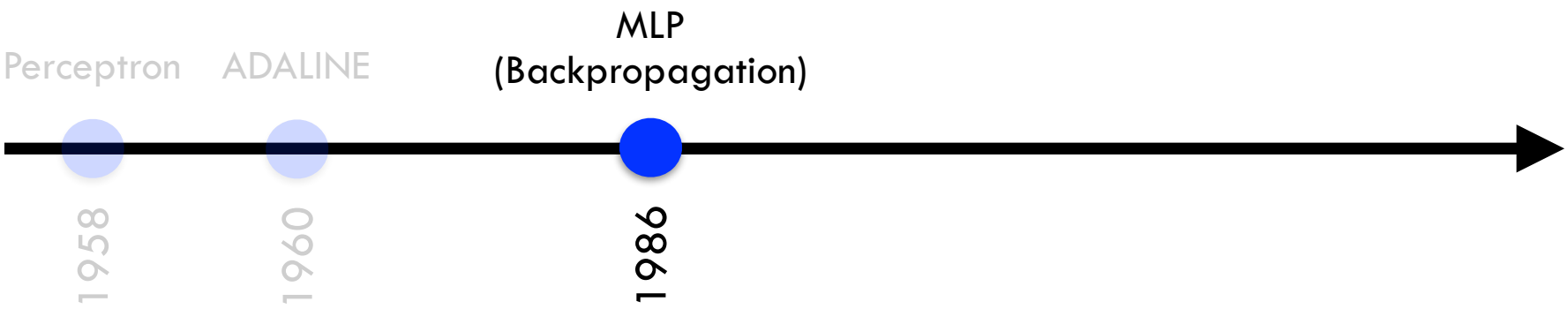


Introdução



Introdução

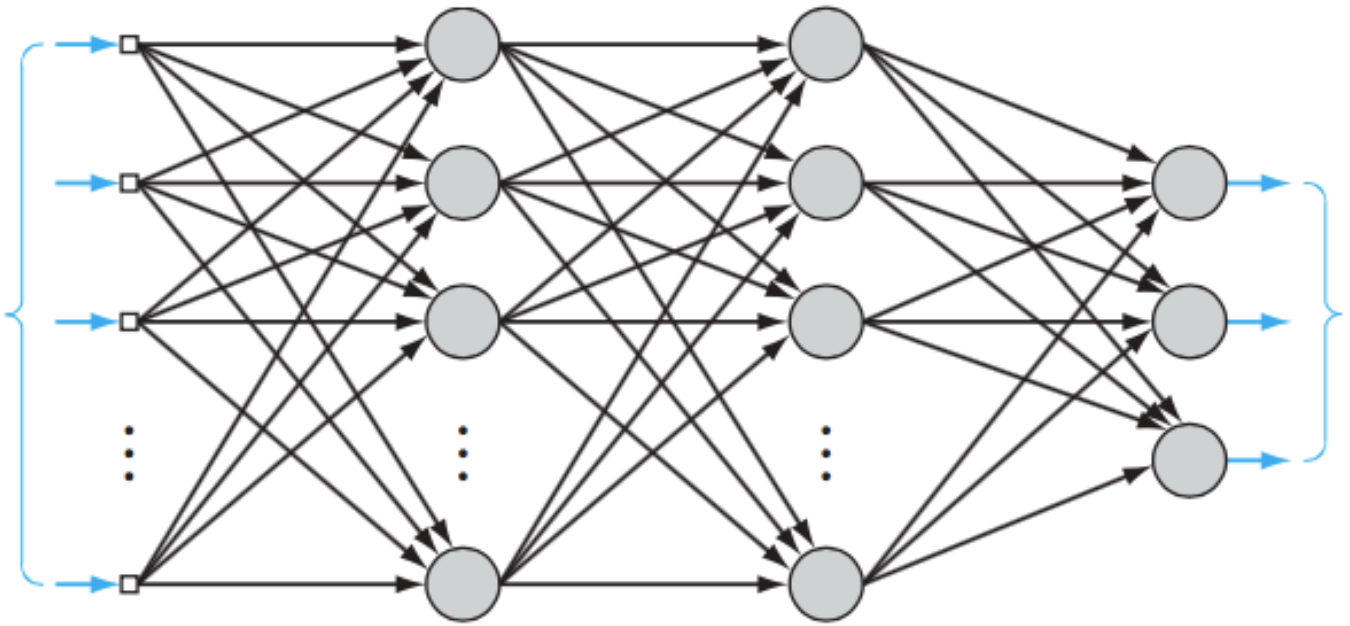
- **MultiLayer Perceptron** (Rosenblatt)
  - Algoritmo de Retropropagação (**backprogation**) [Rumerhalt & et al]
  - Supera as limitações práticas do Perceptron (simples)



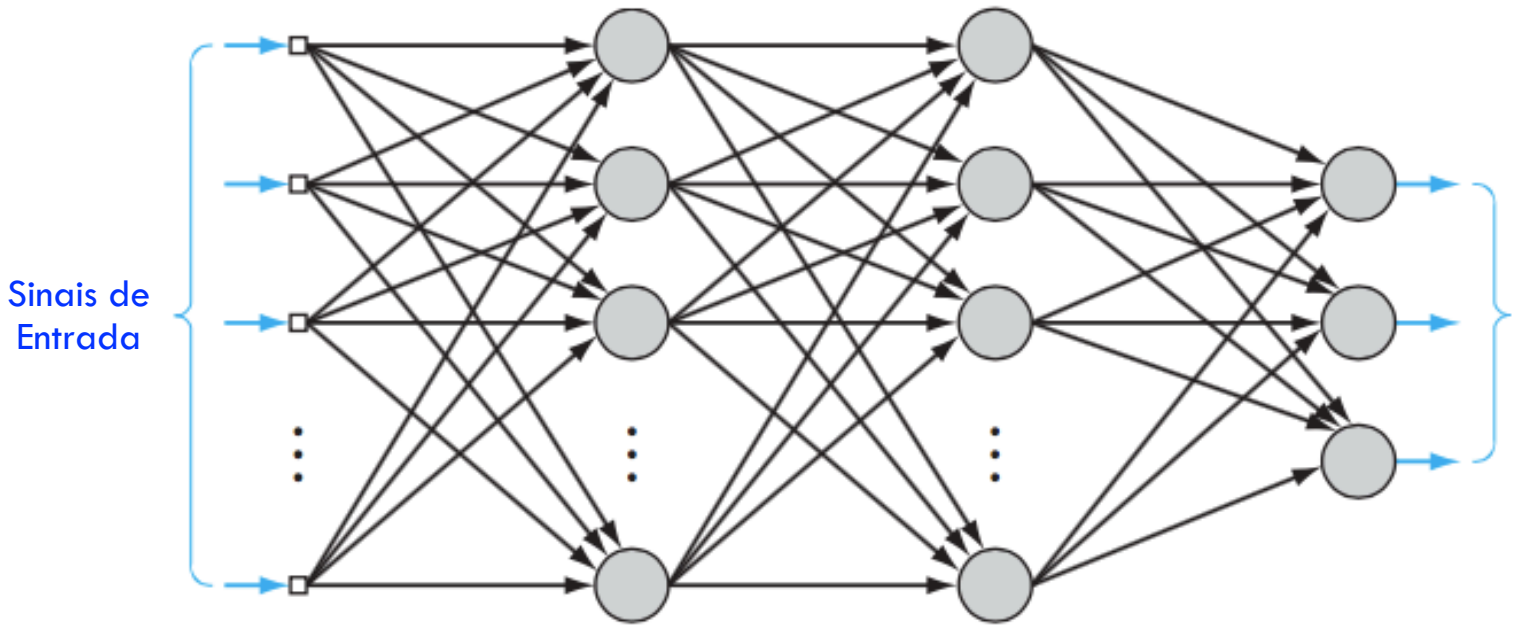
Introdução

- **Multilayer Perceptron:**
  - neurônios possuem uma função de ativação **não-linear e diferenciável**
  - contém uma ou mais camadas escondidas
  - a rede possui alto grau de conectividade

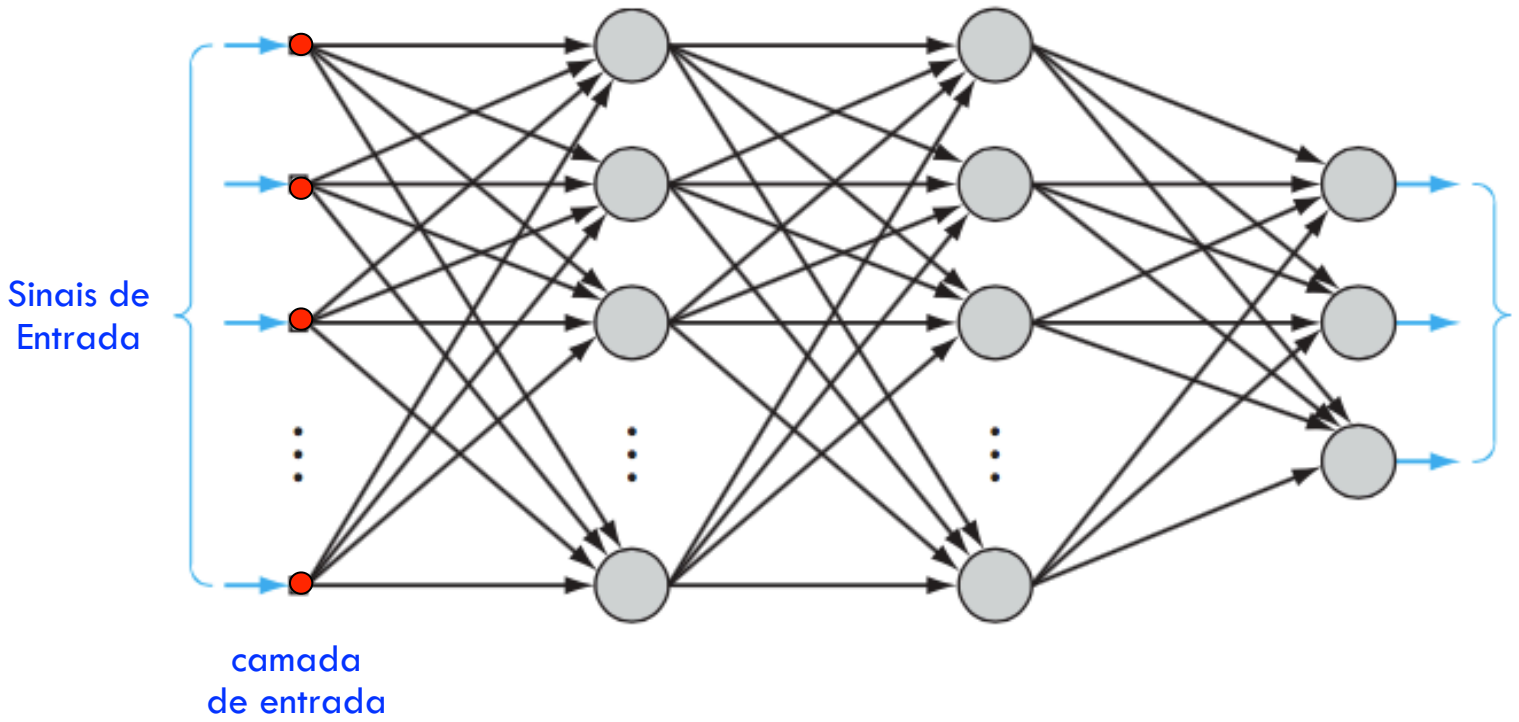
Introdução



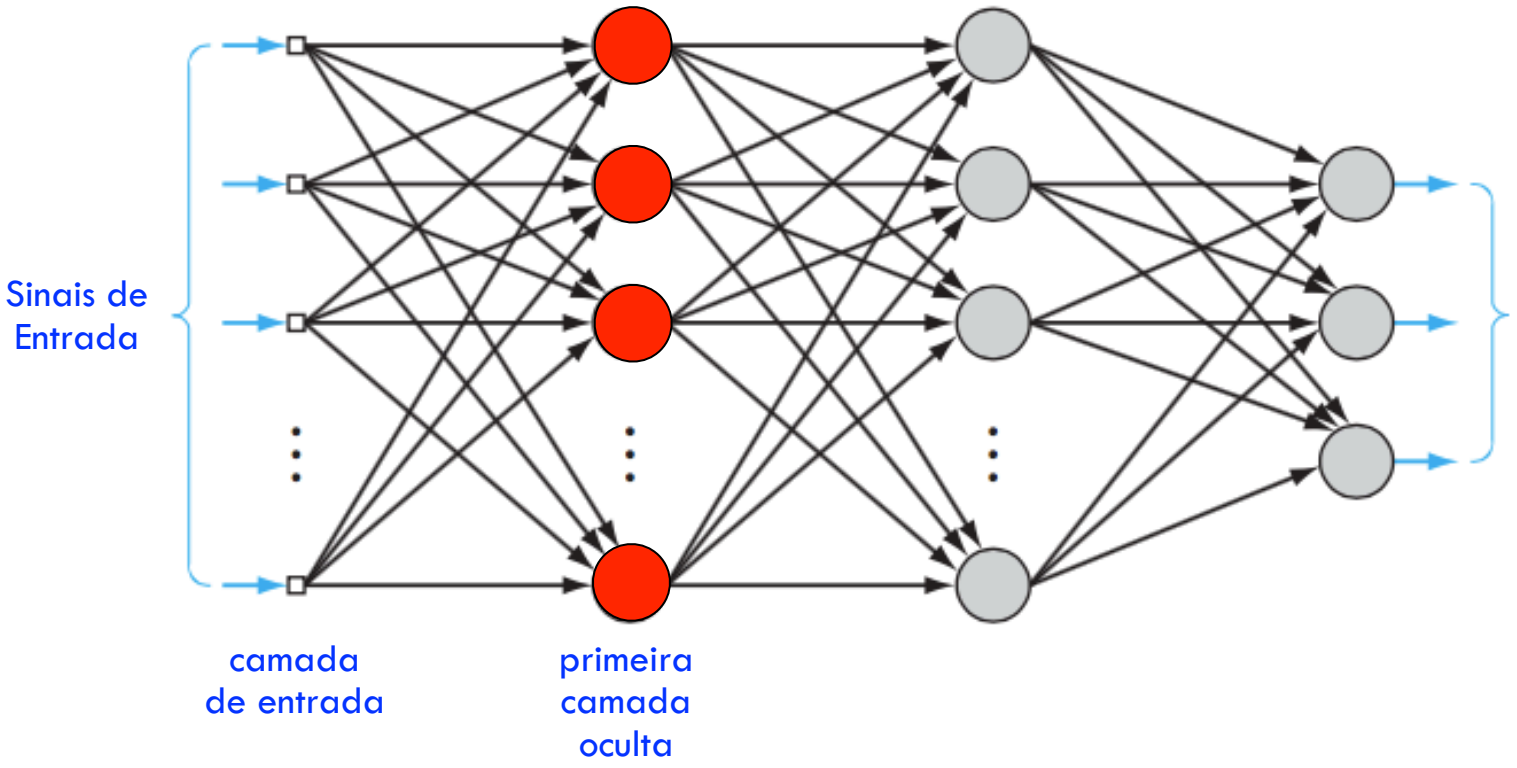
Introdução



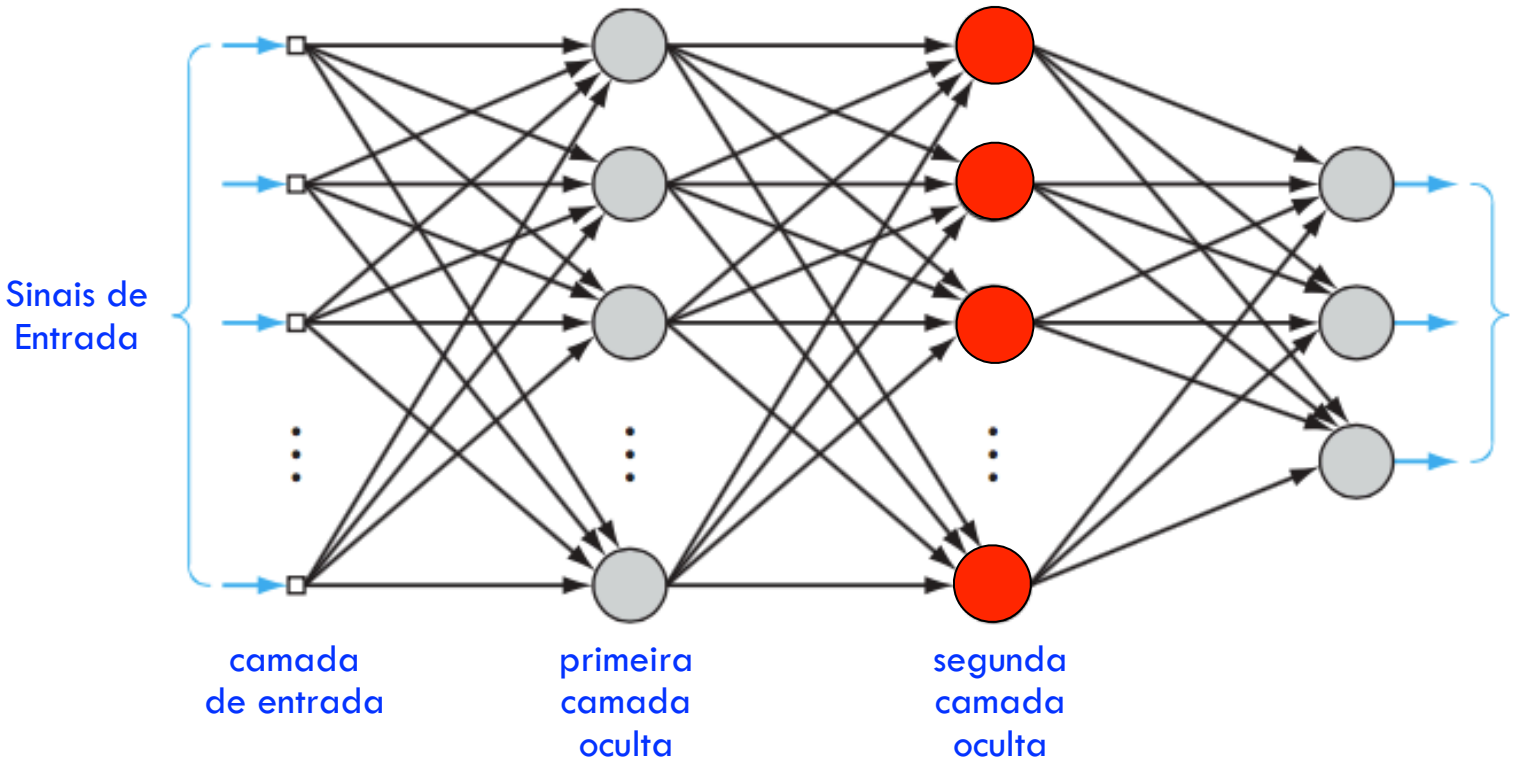
Introdução



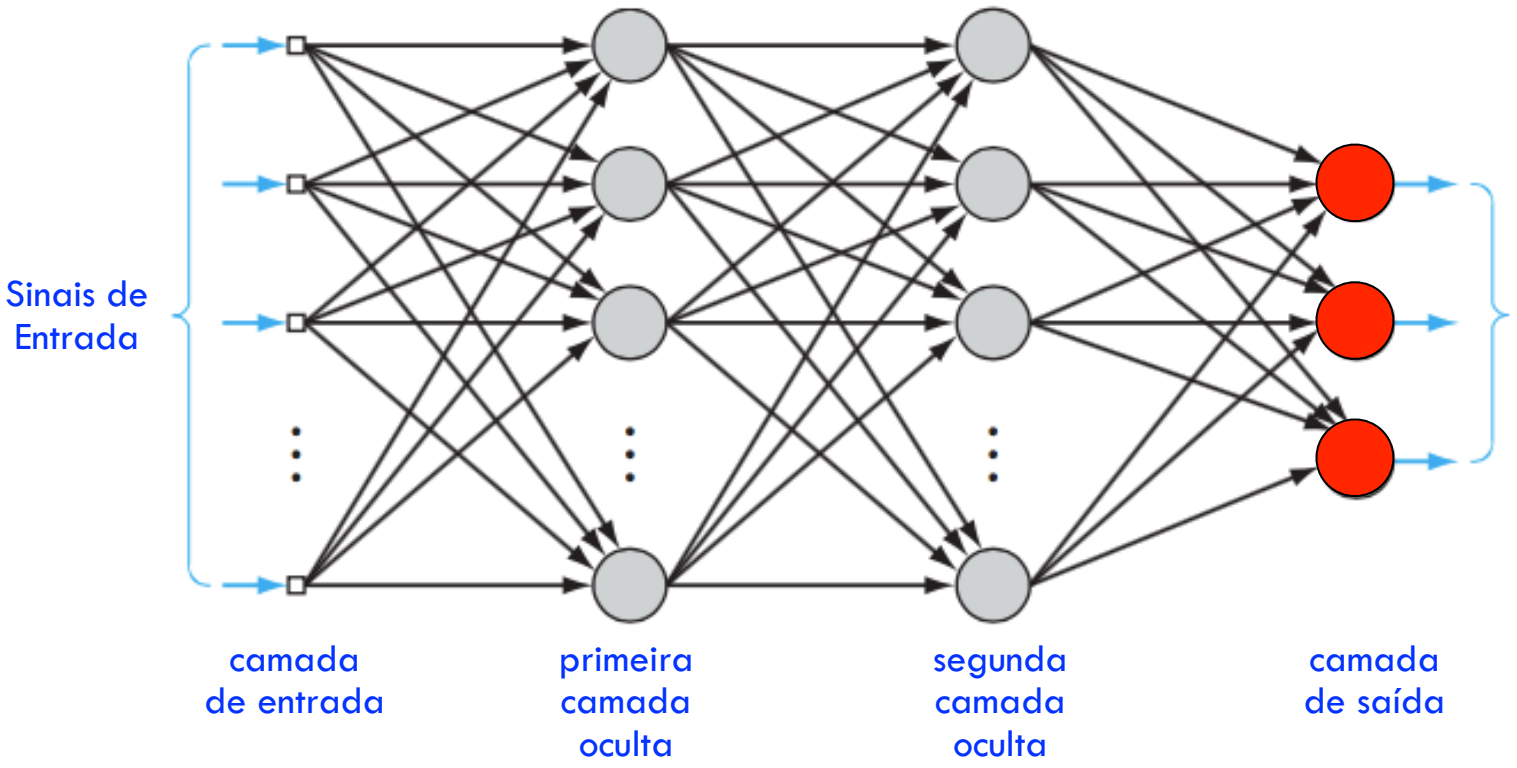
Introdução


Sinais de Entrada
camada de entrada
primeira camada oculta

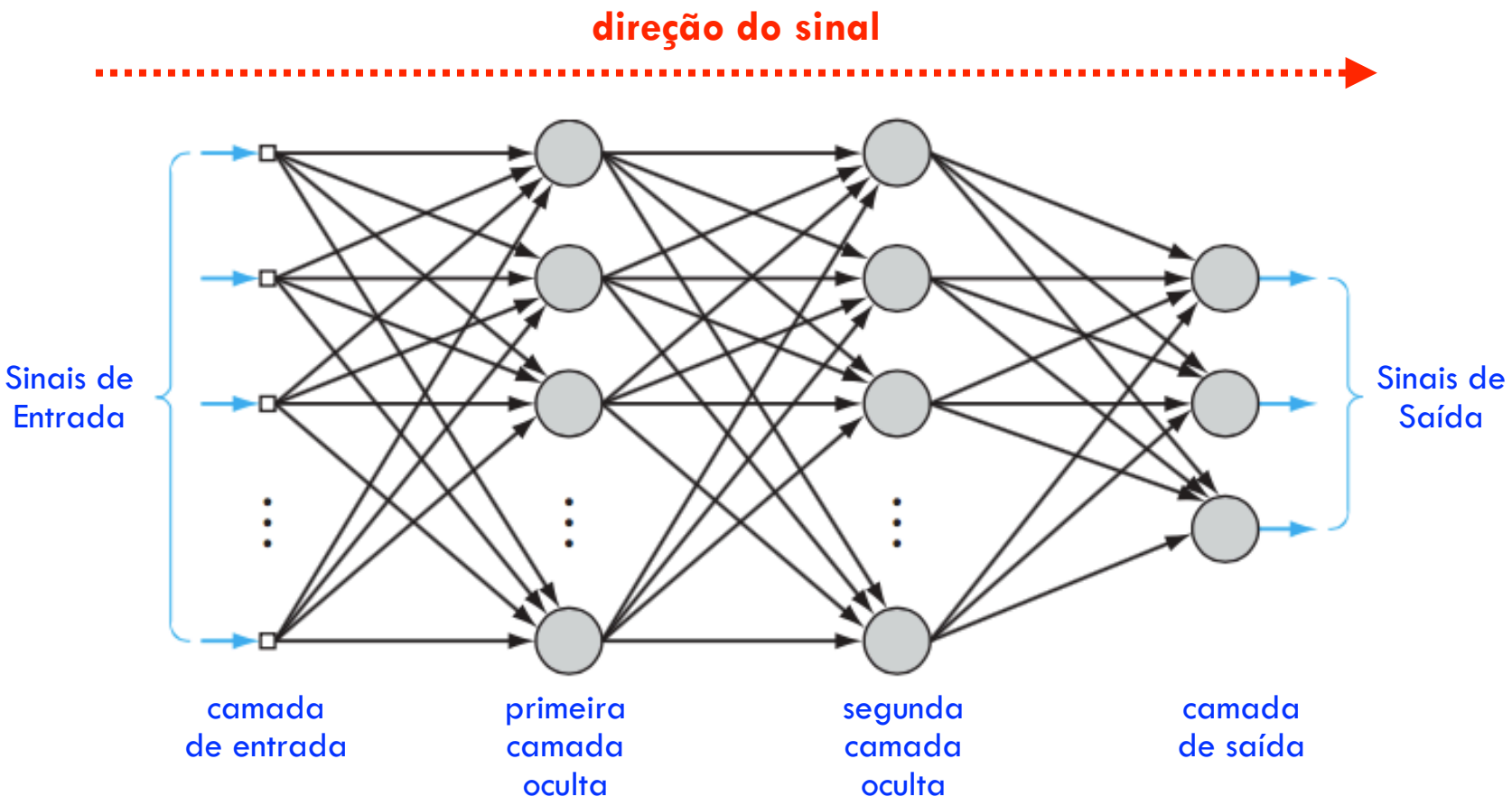
Introdução


Sinais de Entrada
camada de entrada
primeira camada oculta
segunda camada oculta

Introdução


Sinais de Entrada
camada de entrada
primeira camada oculta
segunda camada oculta
camada de saída

Introdução


direção do sinal
Sinais de Entrada
Sinais de Saída
camada de entrada
primeira camada oculta
segunda camada oculta
camada de saída

Introdução

- **Deficiências:**

Introdução

- **Deficiências:**

1. A **análise teórica** é **difícil.** Há muitas conexões e funções não-lineares

Introdução

- **Deficiências:**

1. A **análise teórica** é **difícil.** Há muitas conexões e funções não-lineares

2. Muitos neurônios: **difícil** de **visualizar** o processo de aprendizado

Introdução

- **Deficiências:**

1. A **análise teórica** é **difícil.** Há muitas conexões e funções não-lineares

2. Muitos neurônios: **difícil** de **visualizar** o processo de aprendizado

3. **Aprendizado** é **difícil**. Há um espaço muito maior de funções, e mais representações dos padrões de entrada

Roteiro

1. Introdução
2. **Multilayer Perceptron**
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Multilayer Perceptron

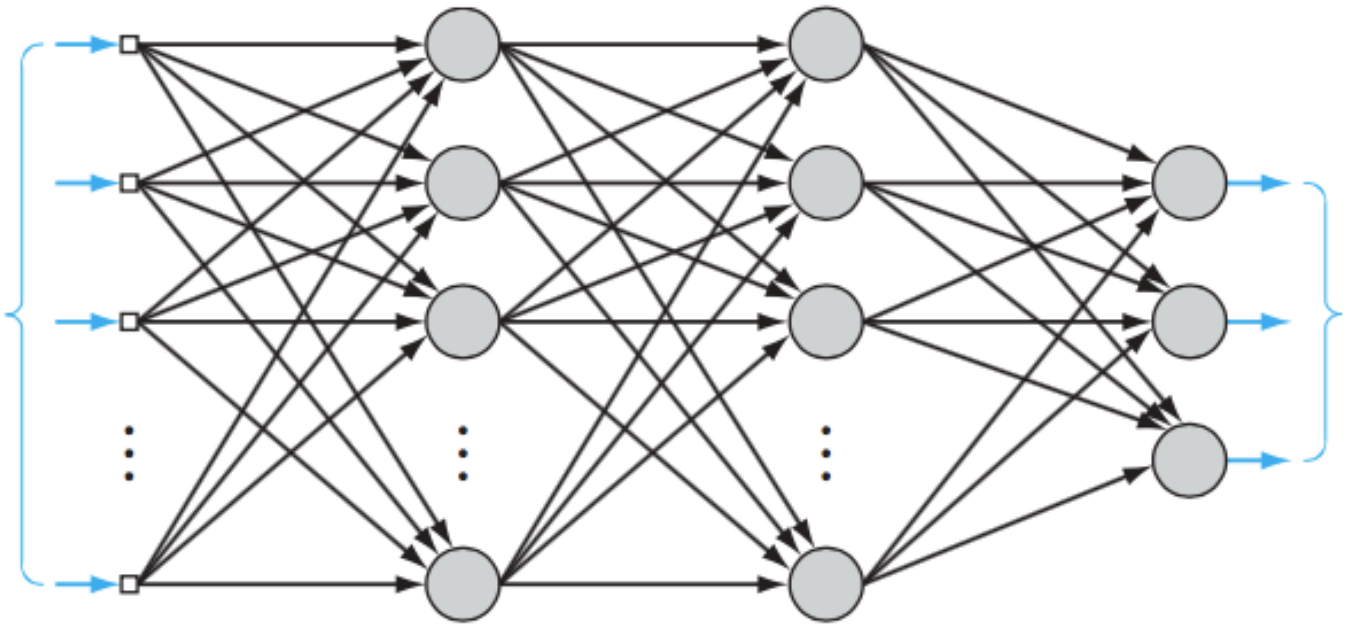
Como **aprender**?

Multilayer Perceptron

- ***Backpropagation***

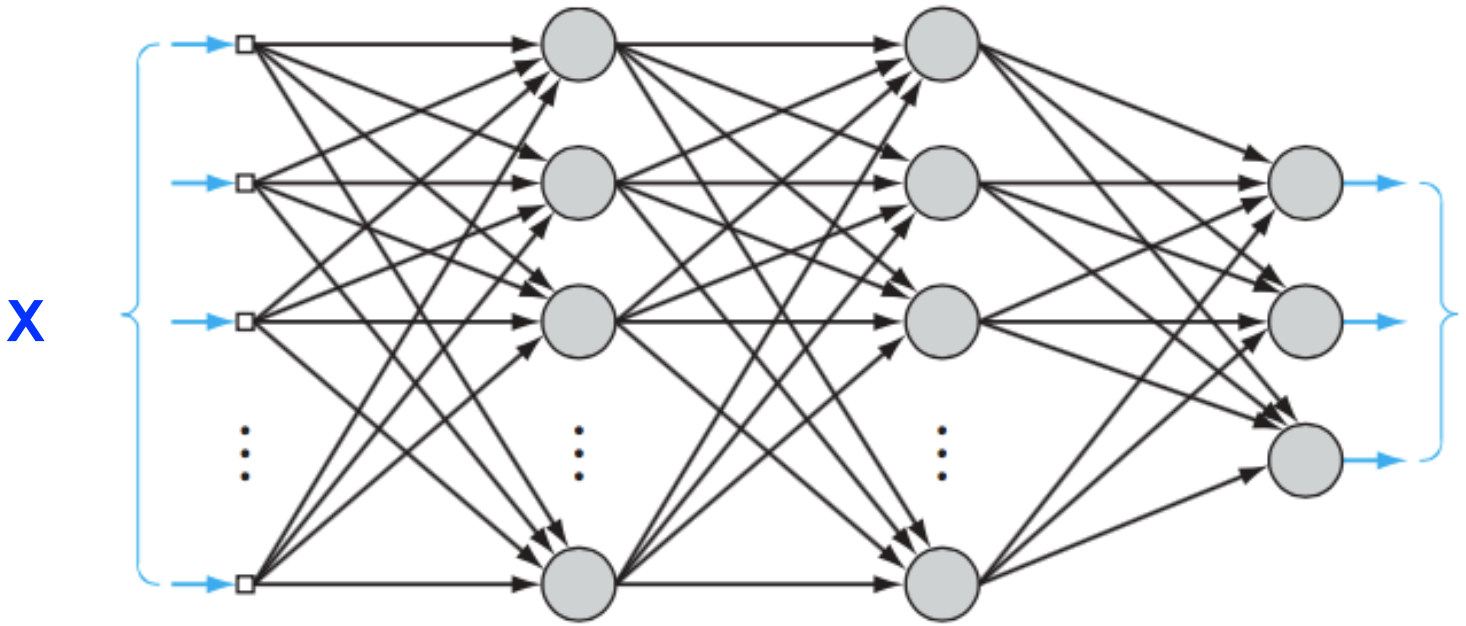
Multilayer Perceptron

- ***Backpropagation***



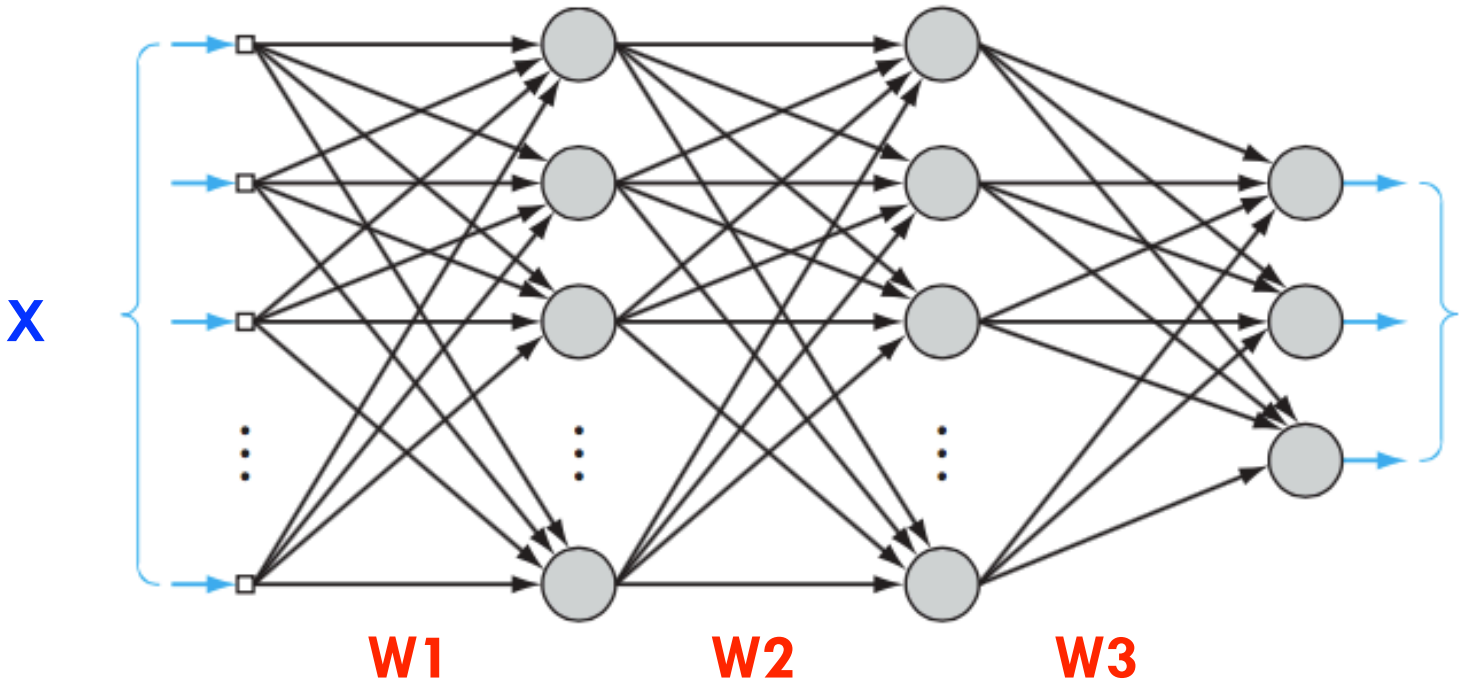
Multilayer Perceptron

- ***Backpropagation***



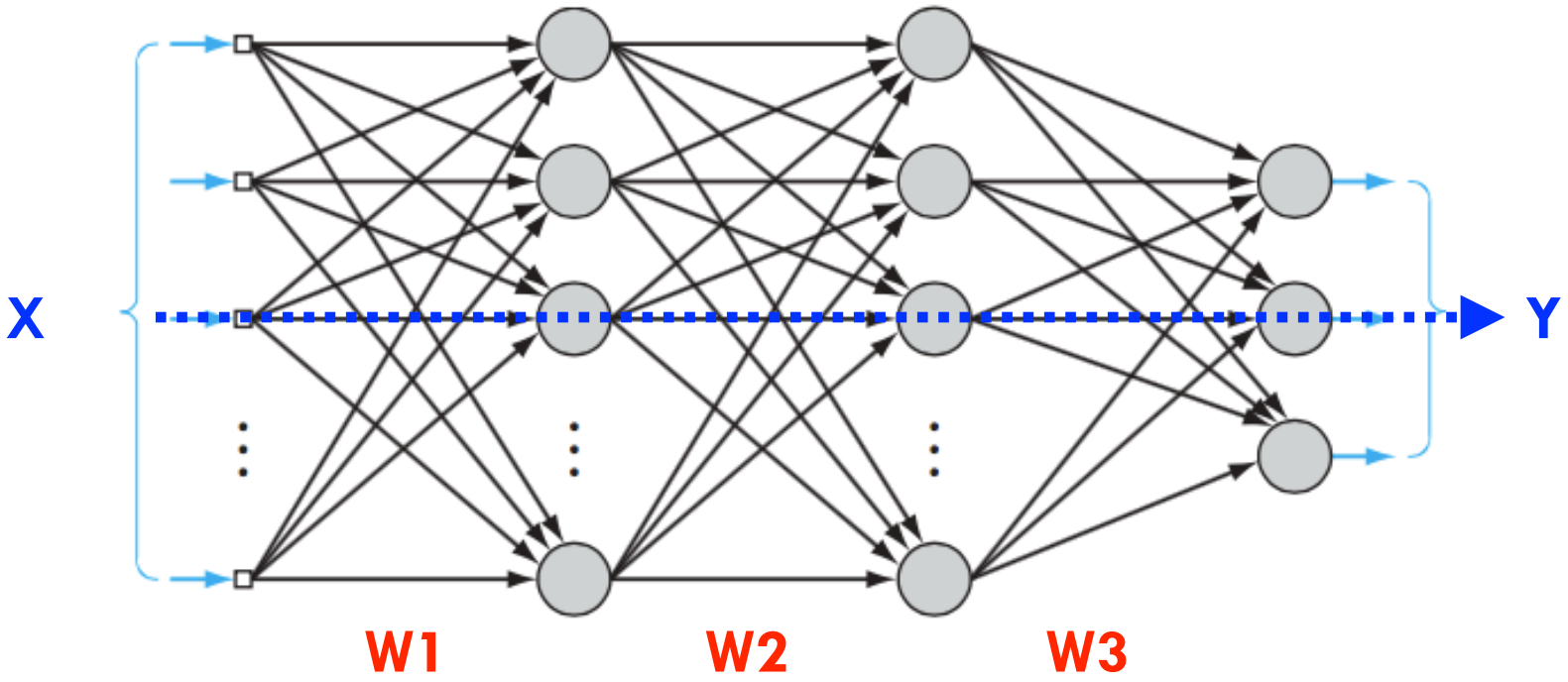
Multilayer Perceptron

- ***Backpropagation***



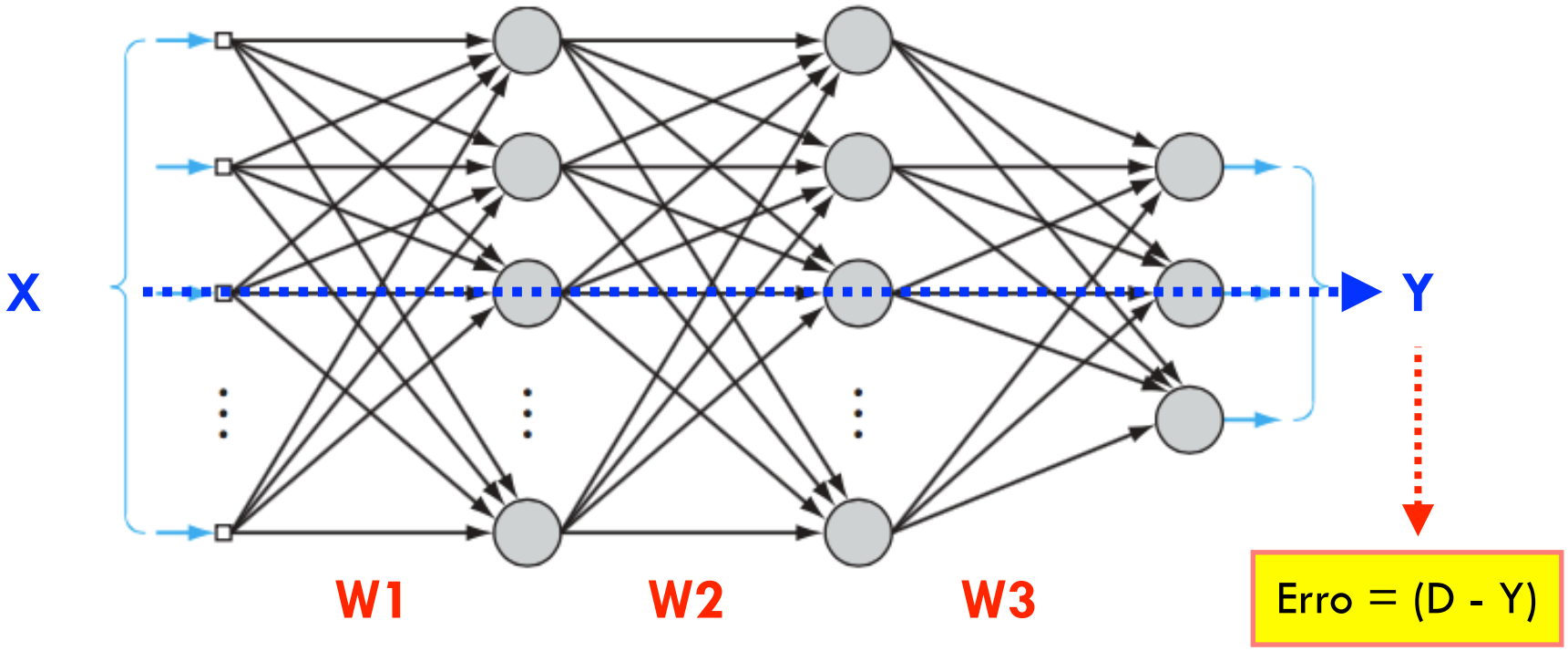
Multilayer Perceptron

- ***Backpropagation***



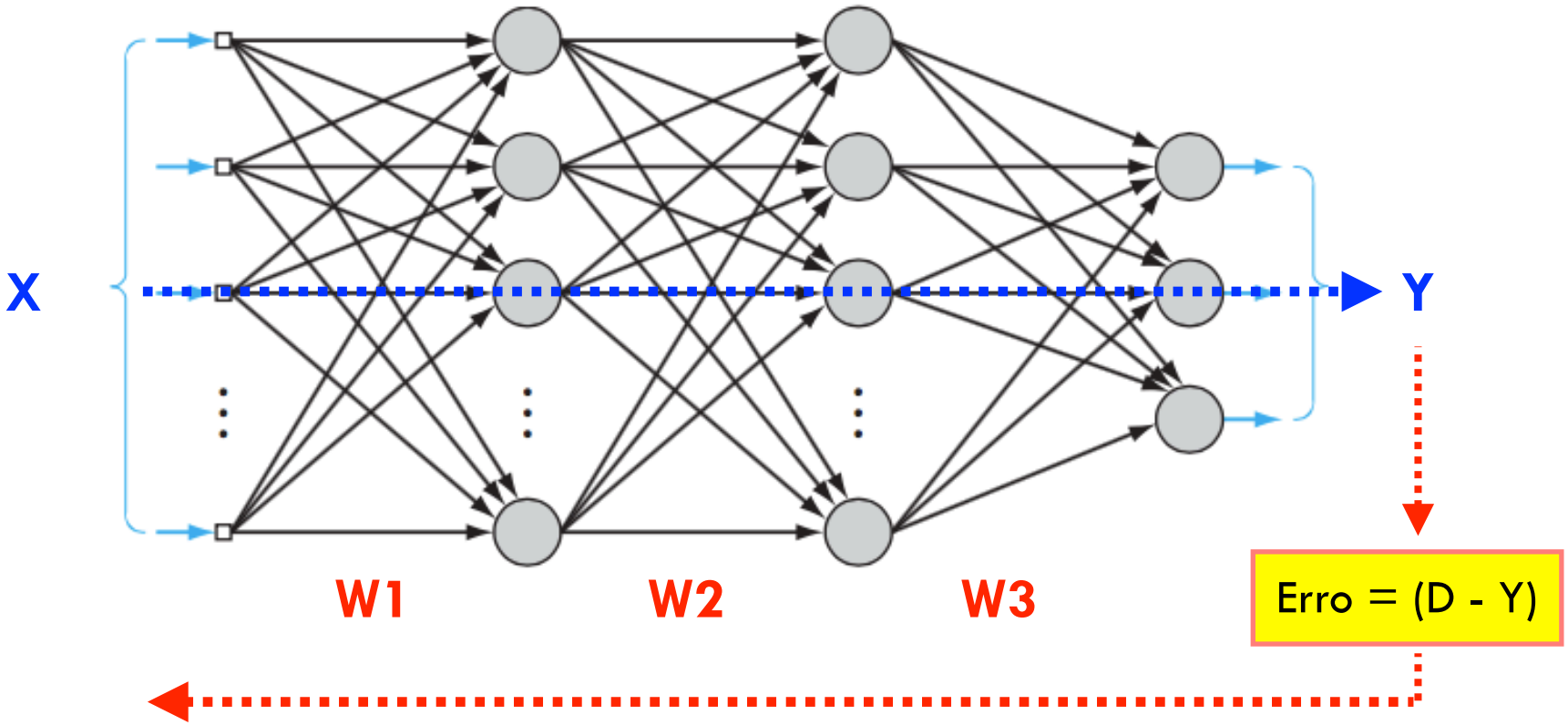
Multilayer Perceptron

- ***Backpropagation***



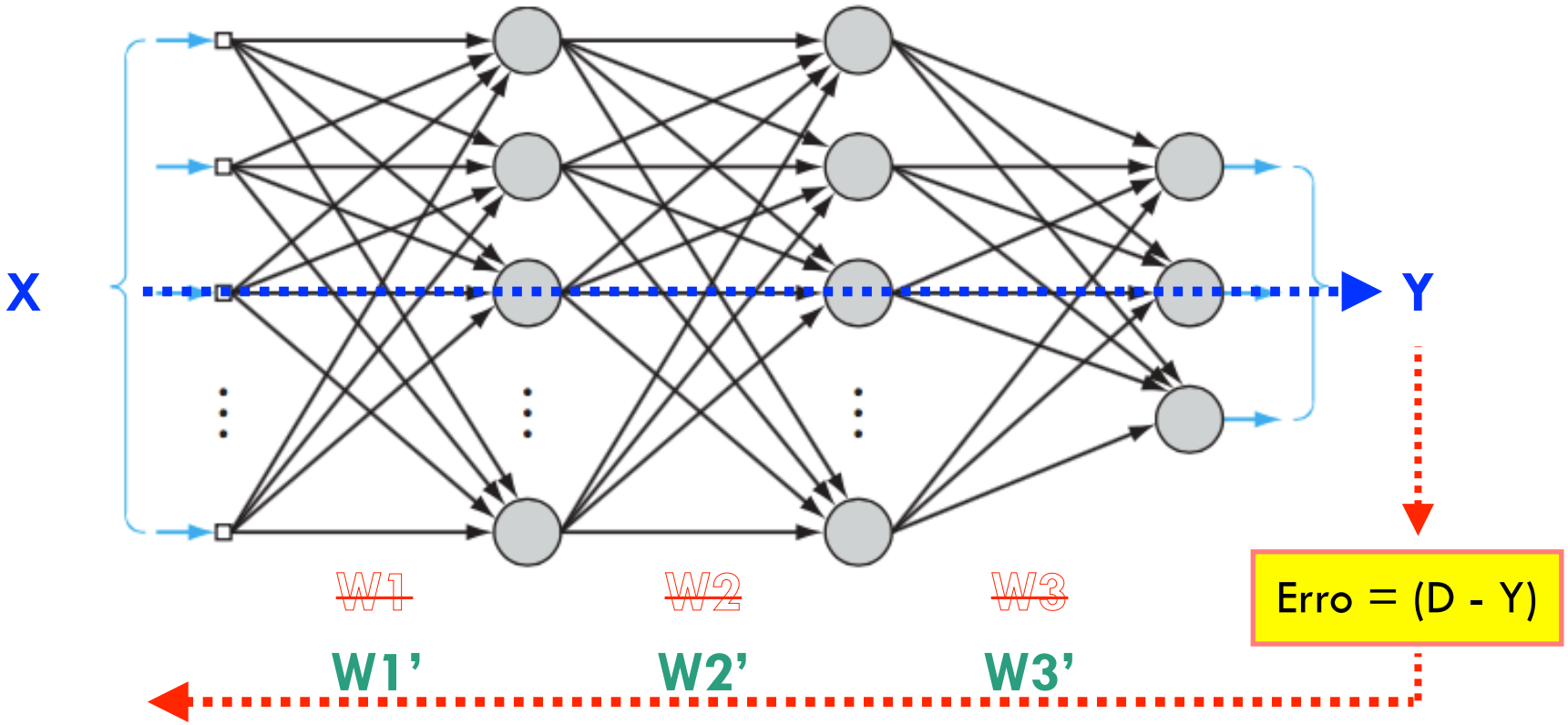
Multilayer Perceptron

- ***Backpropagation***



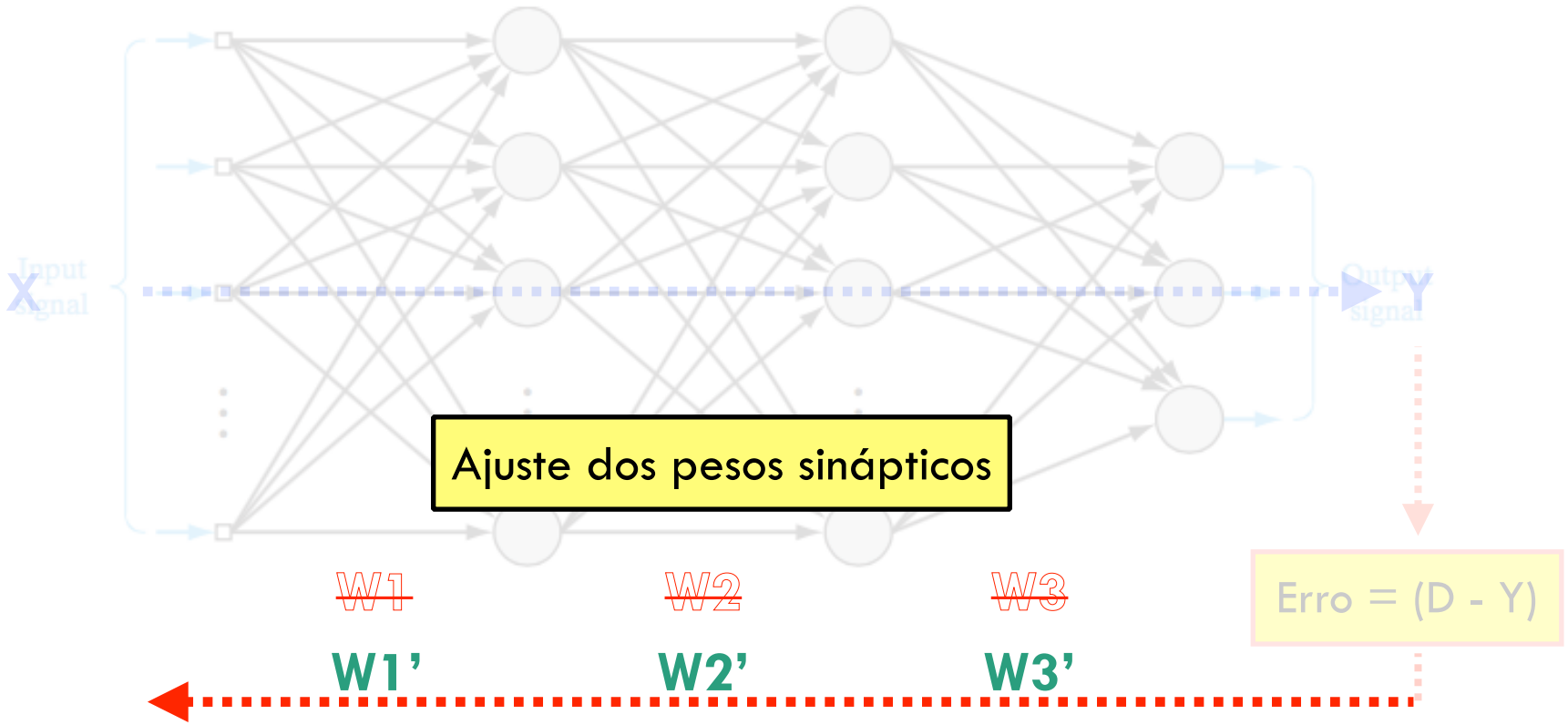
Multilayer Perceptron

- ***Backpropagation***



Multilayer Perceptron

- ***Backpropagation***



Multilayer Perceptron

- **2 Fases:**

Multilayer Perceptron

- **2 Fases:**
  - ***Forward***: propagação do sinal

    Entrada → Camadas Ocultas → Saída

    Estimativa do erro

Multilayer Perceptron

- **2 Fases:**

  - ***Forward***: propagação do sinal

    Entrada → Camadas Ocultas → Saída

    Estimativa do erro

  - ***Backward***: sinal de erro é retropropagado

    Entrada ← Camadas Ocultas ← Saída

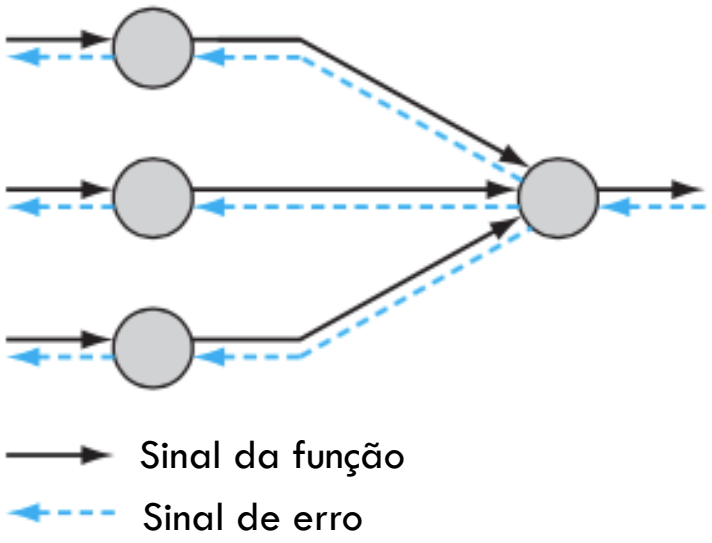
    Ajustes sinápticos

Multilayer Perceptron

- Neurônios da camada oculta, executam dois cálculos:

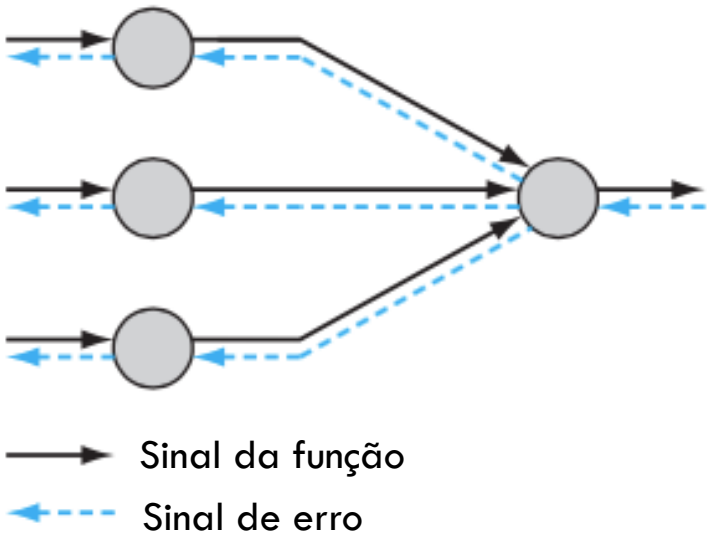
Multilayer Perceptron

- Neurônios da camada oculta, executam dois cálculos:



Multilayer Perceptron

- Neurônios da camada oculta, executam dois cálculos:

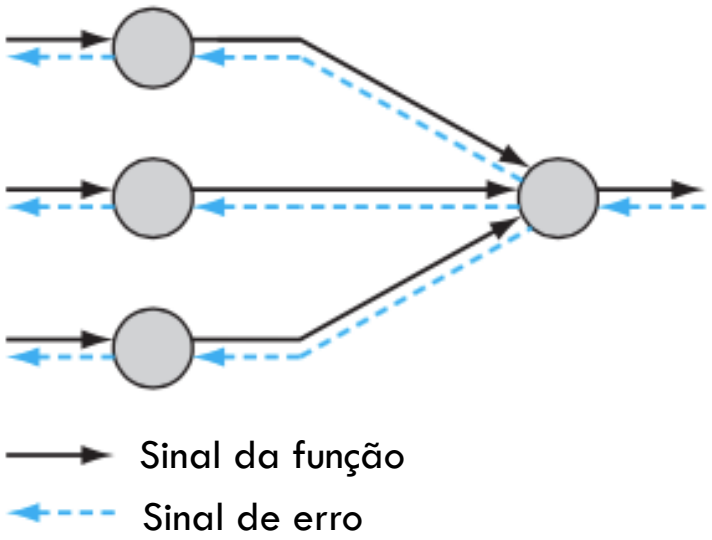


1

- sinal de função (y_{ij})
  - ϕ não-linear
  - ϕ contínua

Multilayer Perceptron

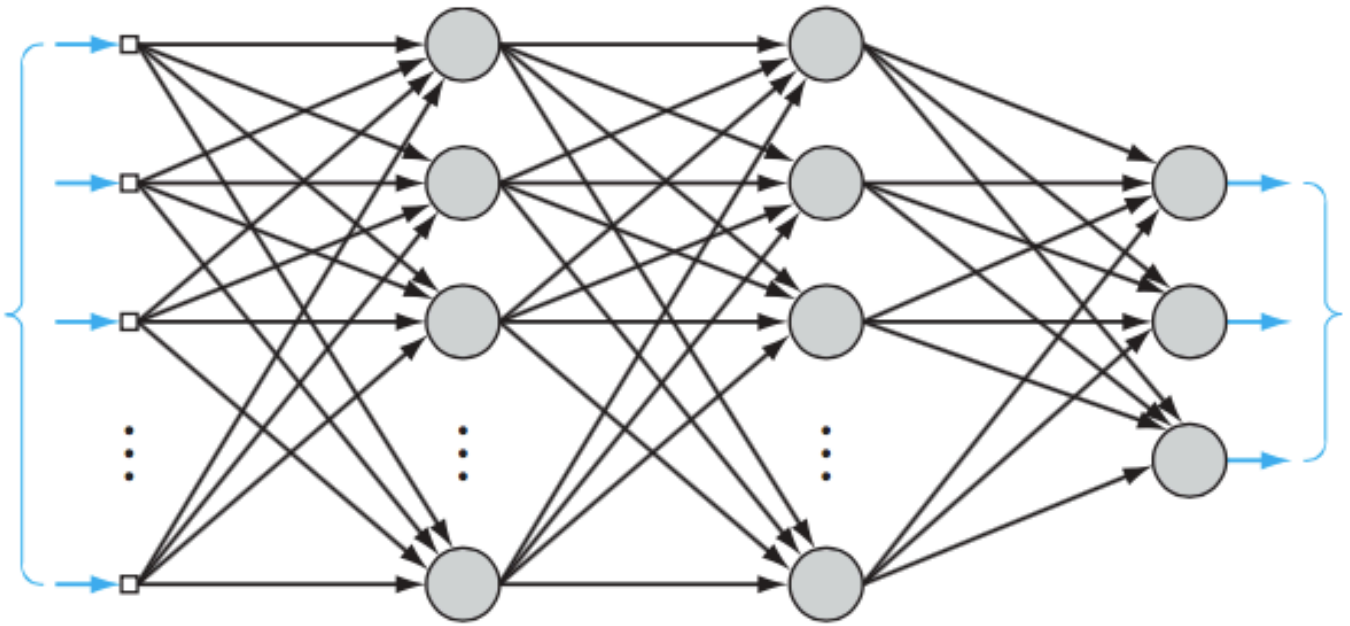
- Neurônios da camada oculta, executam dois cálculos:



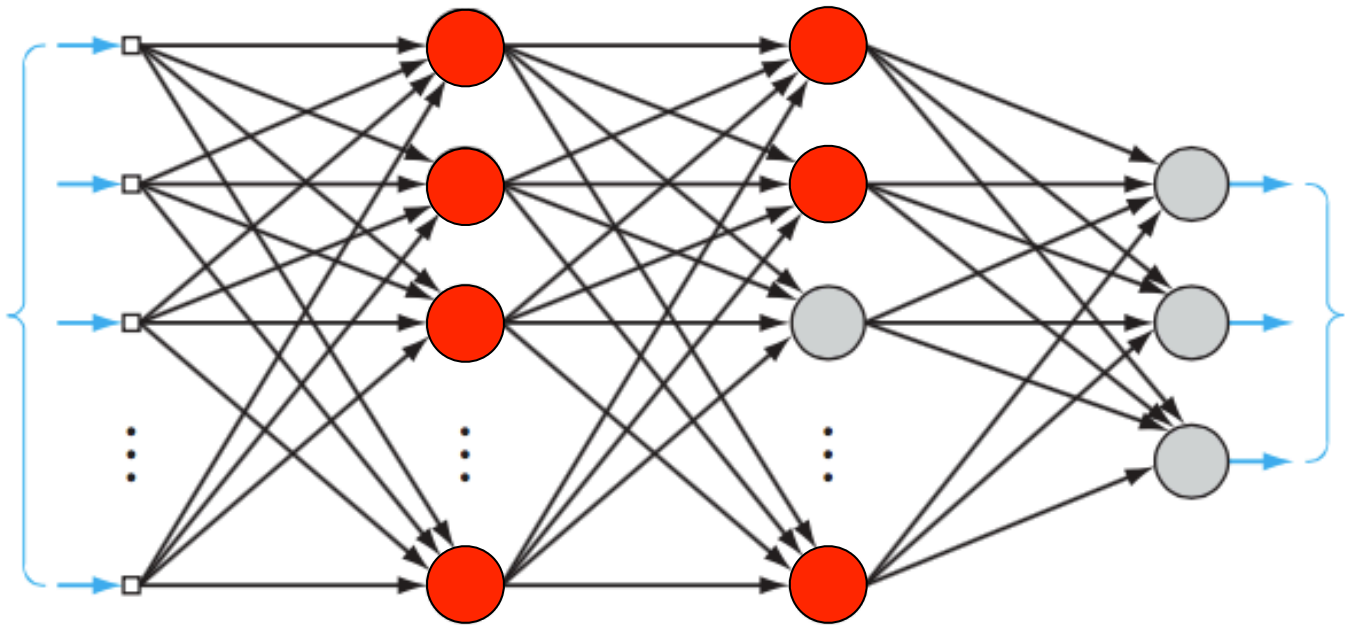
2

- estimativa do vetor gradiente (δ_{ij})
  - gradiente da superfície de erro
  - **retropropagação**

Multilayer Perceptron

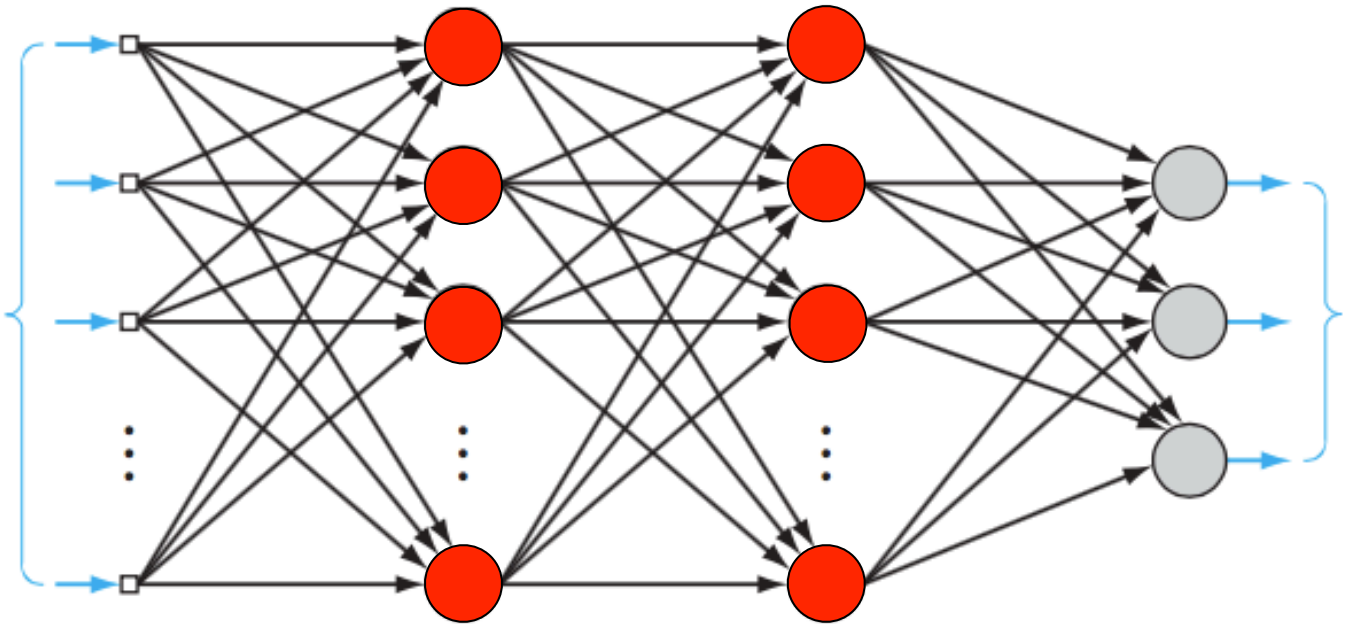


Multilayer Perceptron



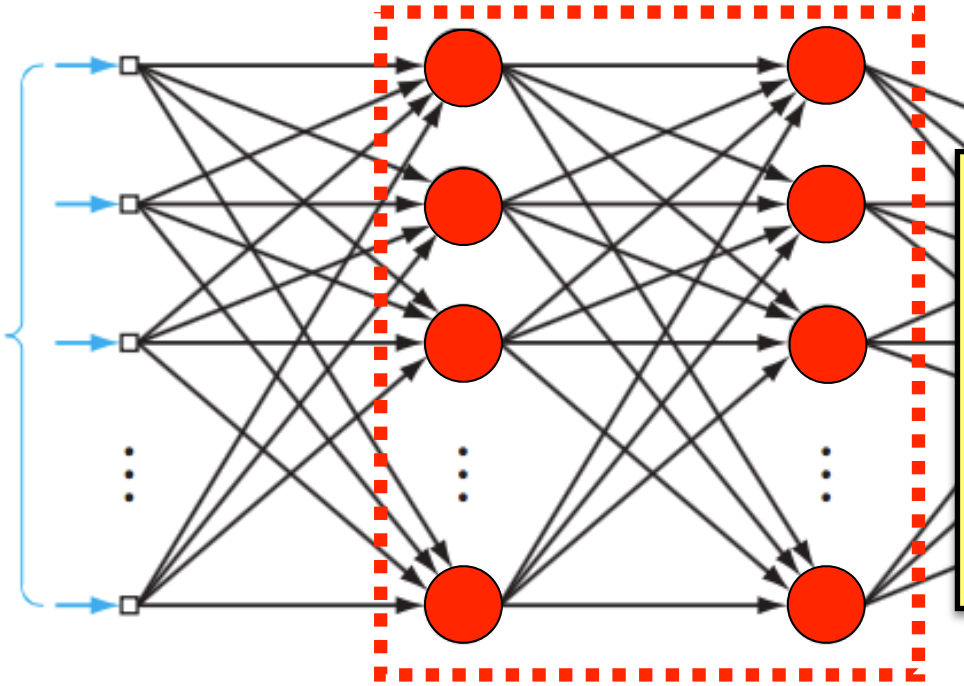
Multilayer Perceptron

- Neurônios das camadas ocultas



Multilayer Perceptron

- Neurônios das camadas ocultas



Função

- detectores de atributos
- transformação não-linear dos dados (ϕ)
- representação do problema em espaço diferente do original

Multilayer Perceptron

- **O que as camadas intermediárias formam?**

Multilayer Perceptron

- **O que as camadas intermediárias formam?**

1 camada → **linhas** retas no espaço de decisão

Multilayer Perceptron

- **O que as camadas intermediárias formam?**

1 camada → **linhas** retas no espaço de decisão
2 camada → combina as linhas da camada anterior em **regiões convexas**

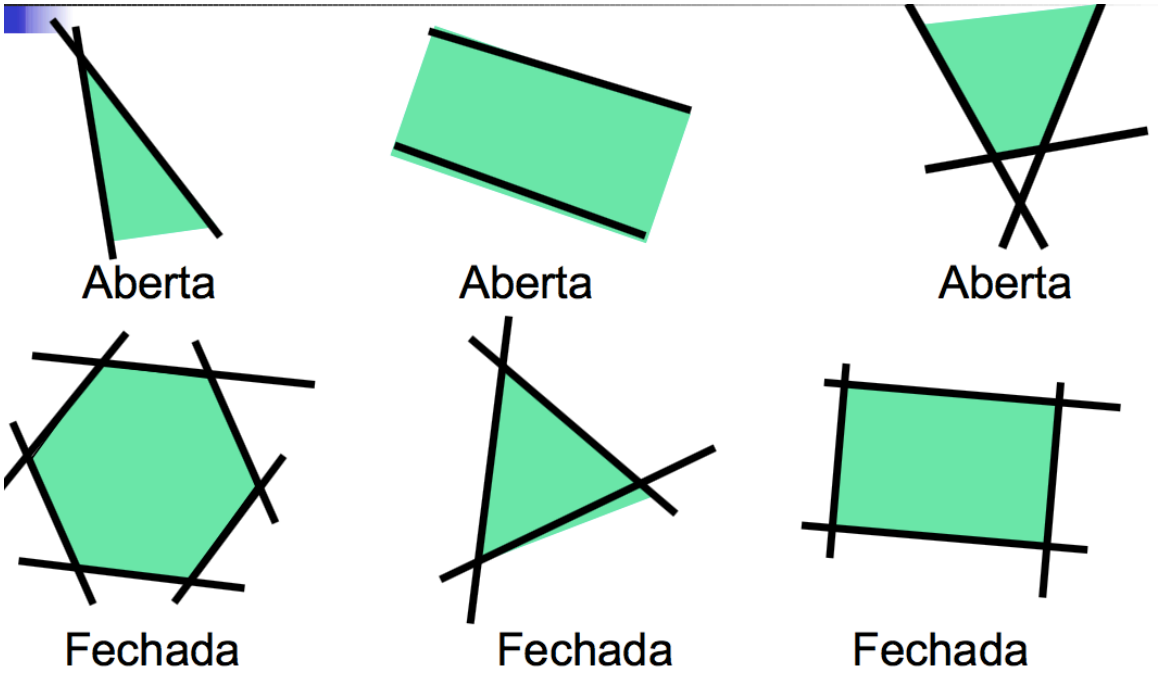
Multilayer Perceptron

- **O que as camadas intermediárias formam?**

1 camada → **linhas** retas no espaço de decisão
2 camada → combina as linhas da camada anterior em **regiões convexas**
3 camada → combina as regiões convexas produzindo **formatos abstratos**
…

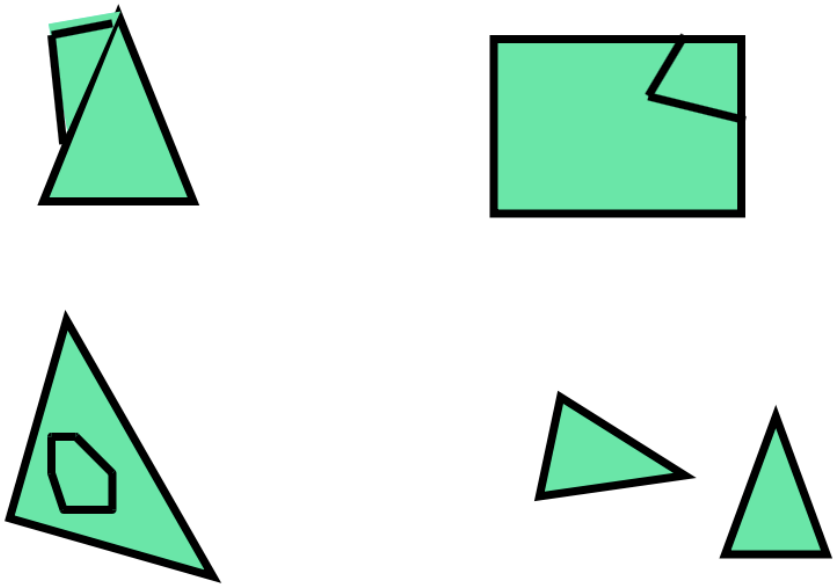
Regiões convexas

- **Combinações de hiperplanos**



Figuras convexas

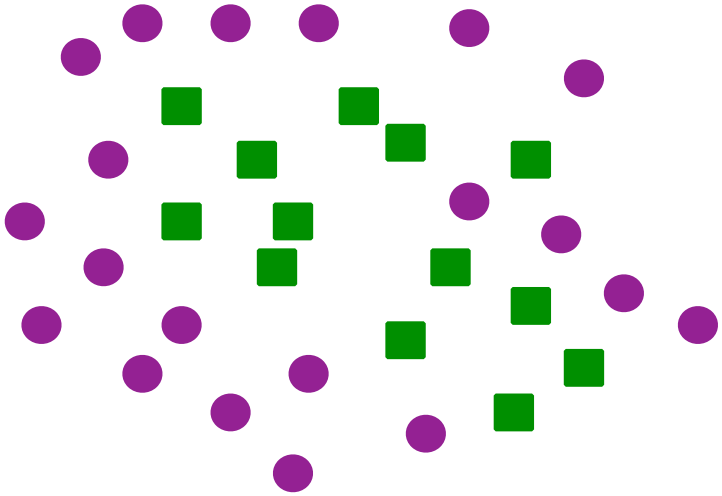
- **Combinações de regiões convexas**

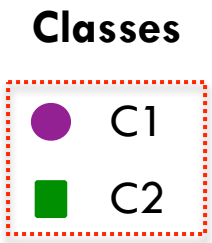


Roteiro

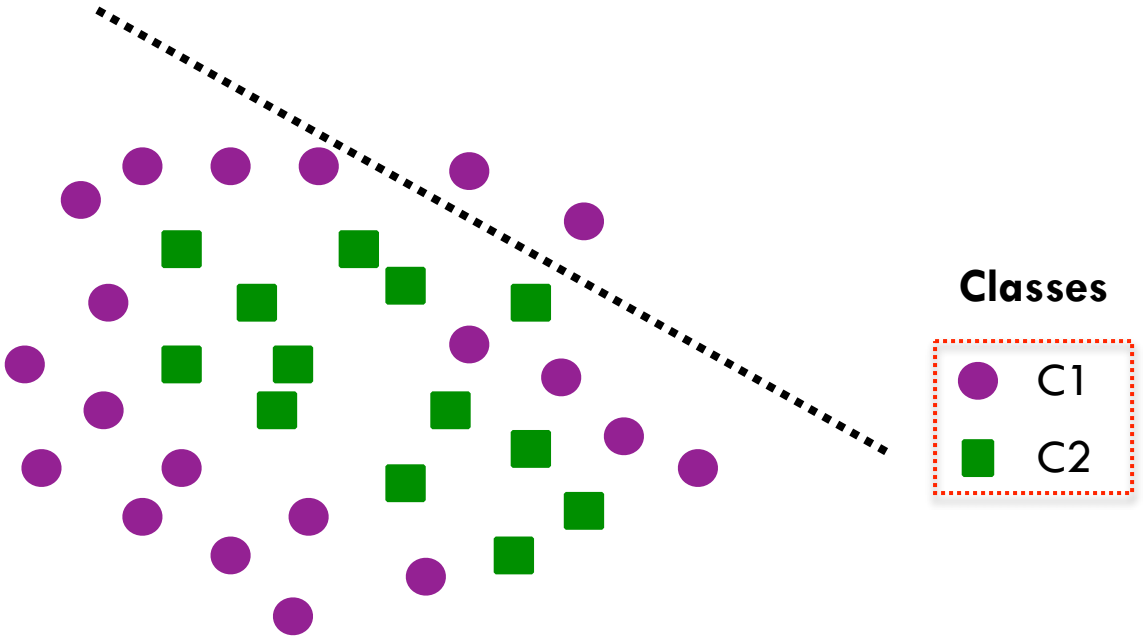
1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Exemplo

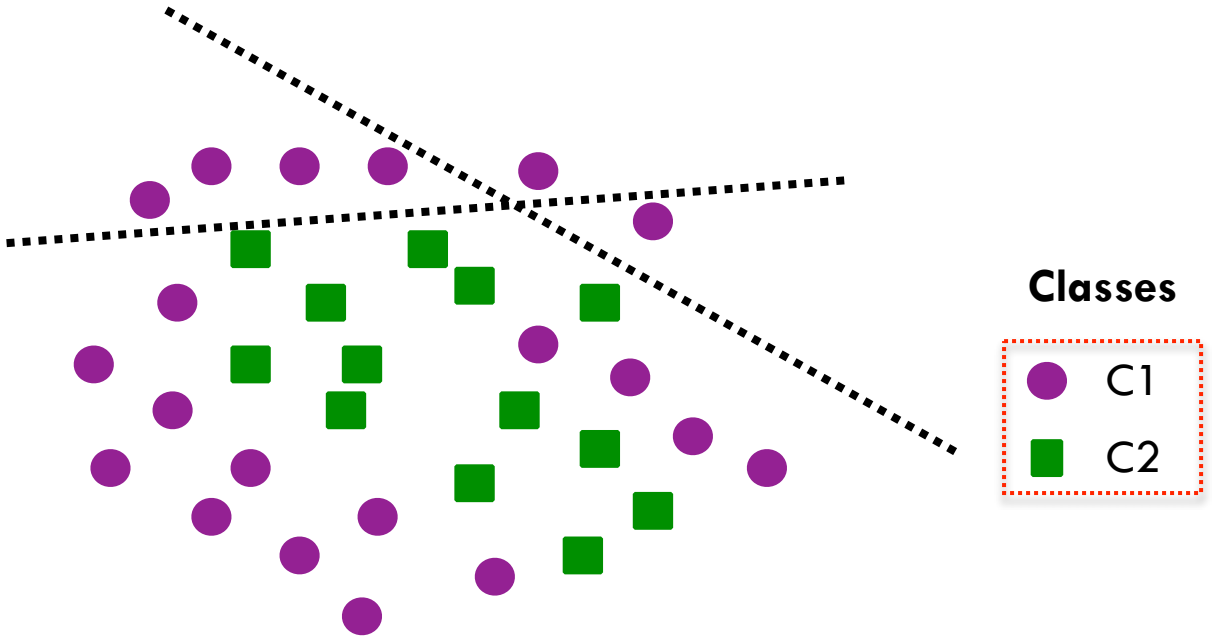




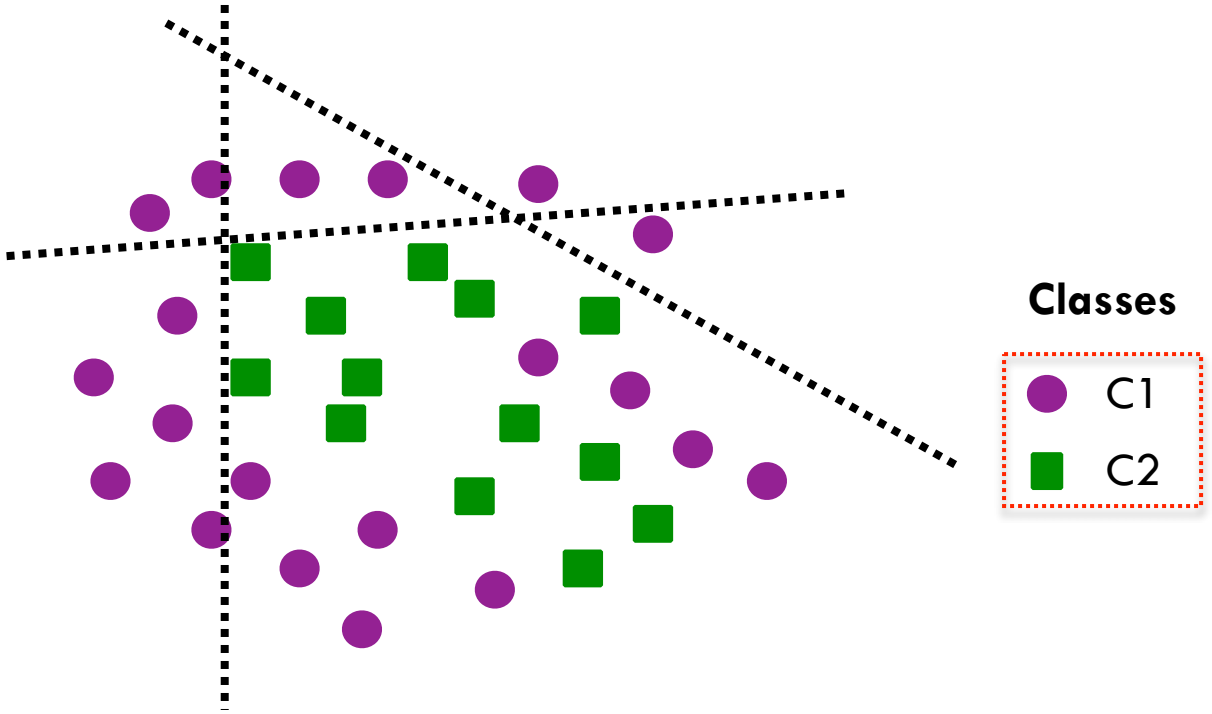
Exemplo



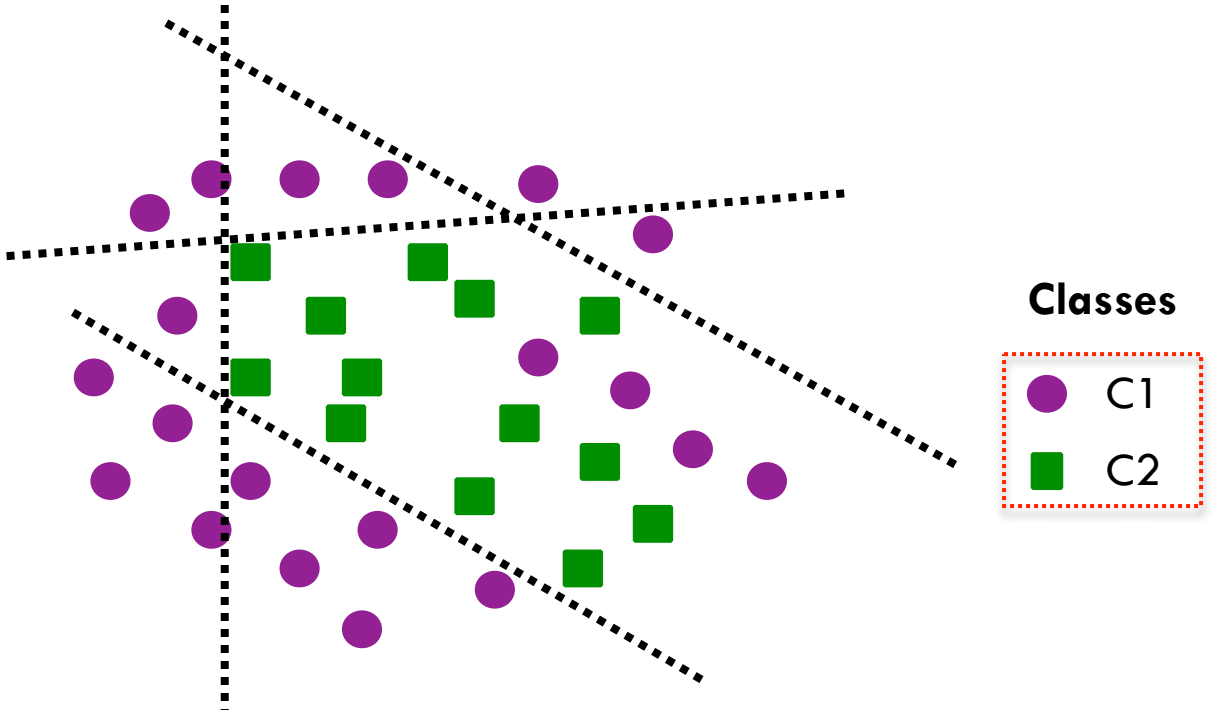
Exemplo



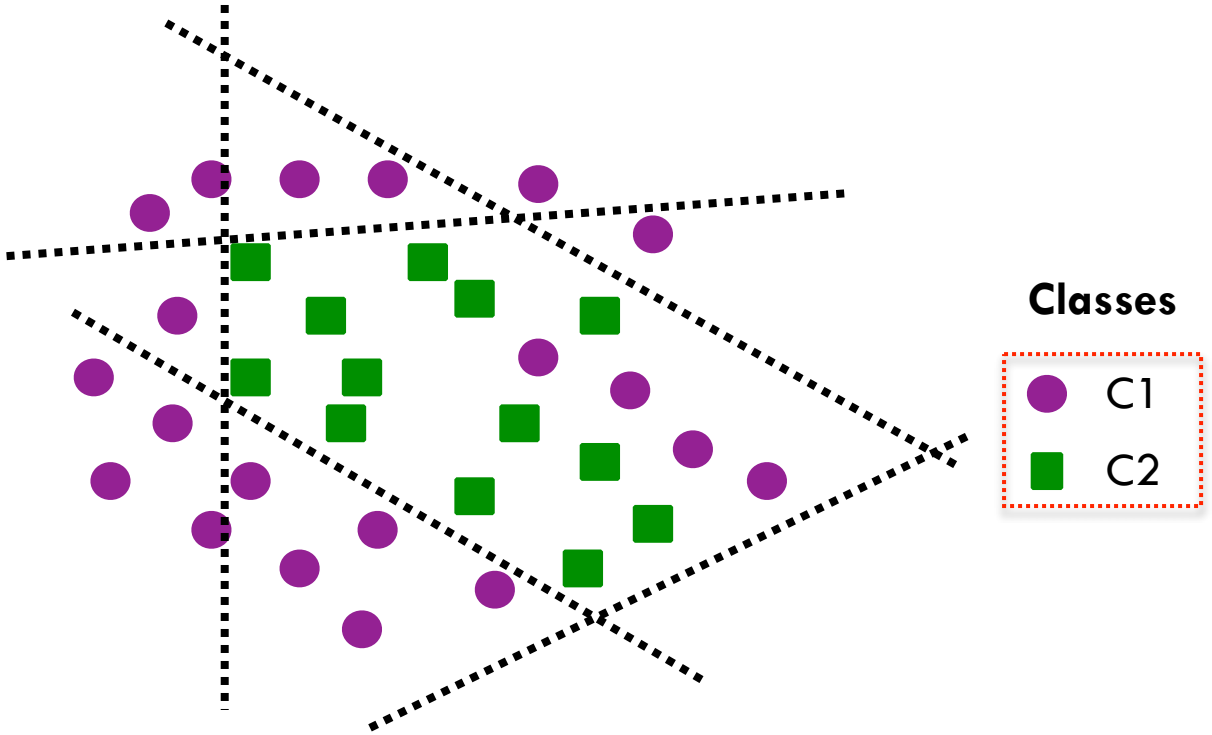
Exemplo



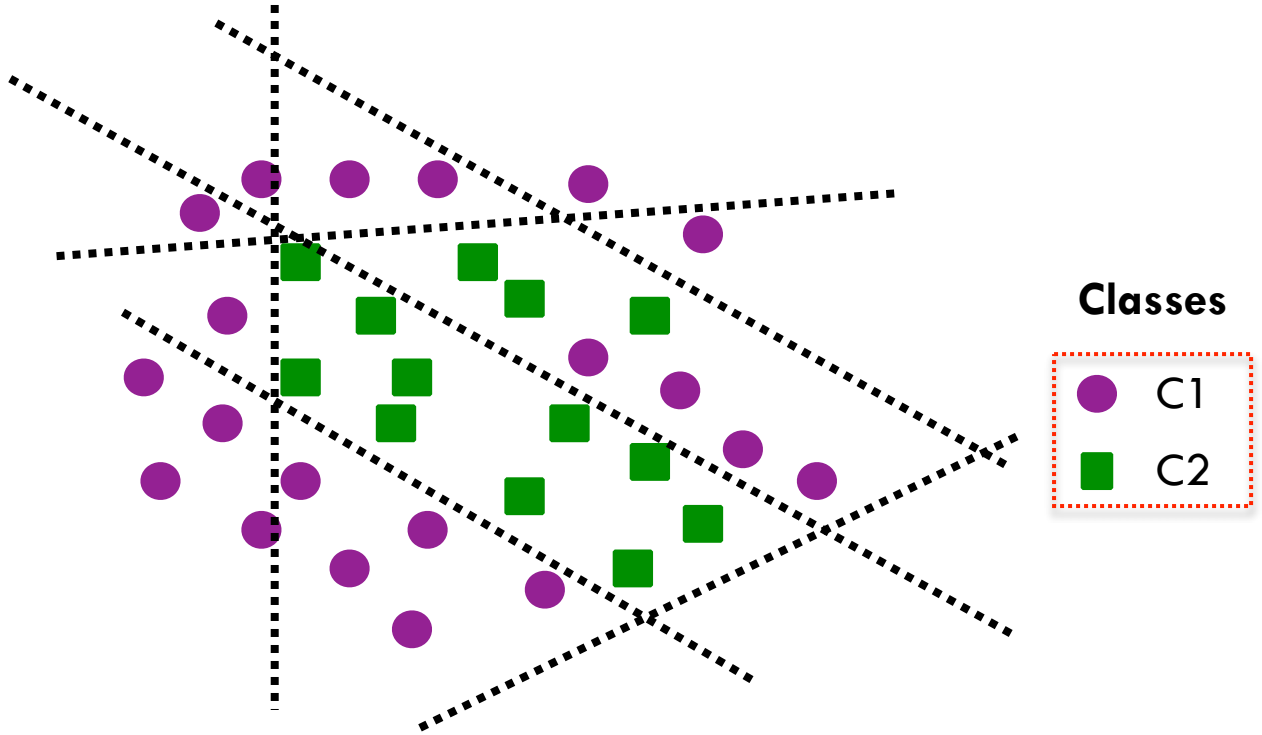
Exemplo



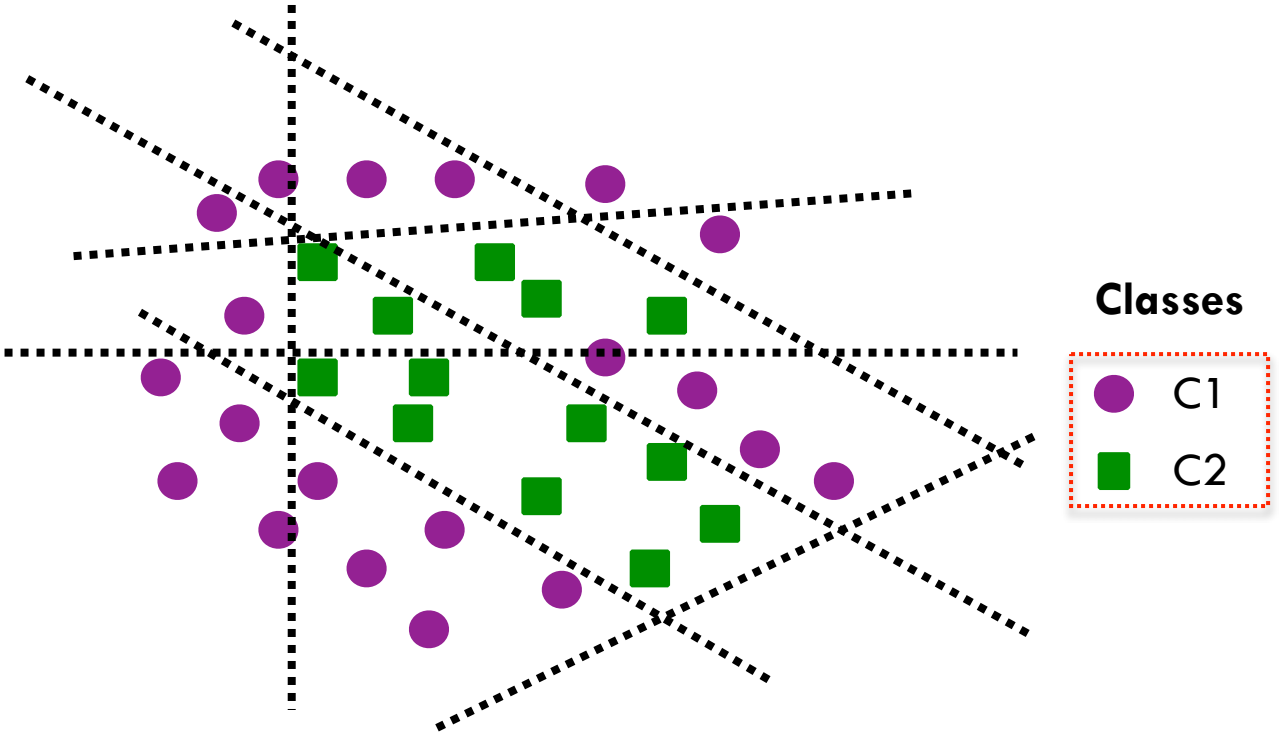
Exemplo



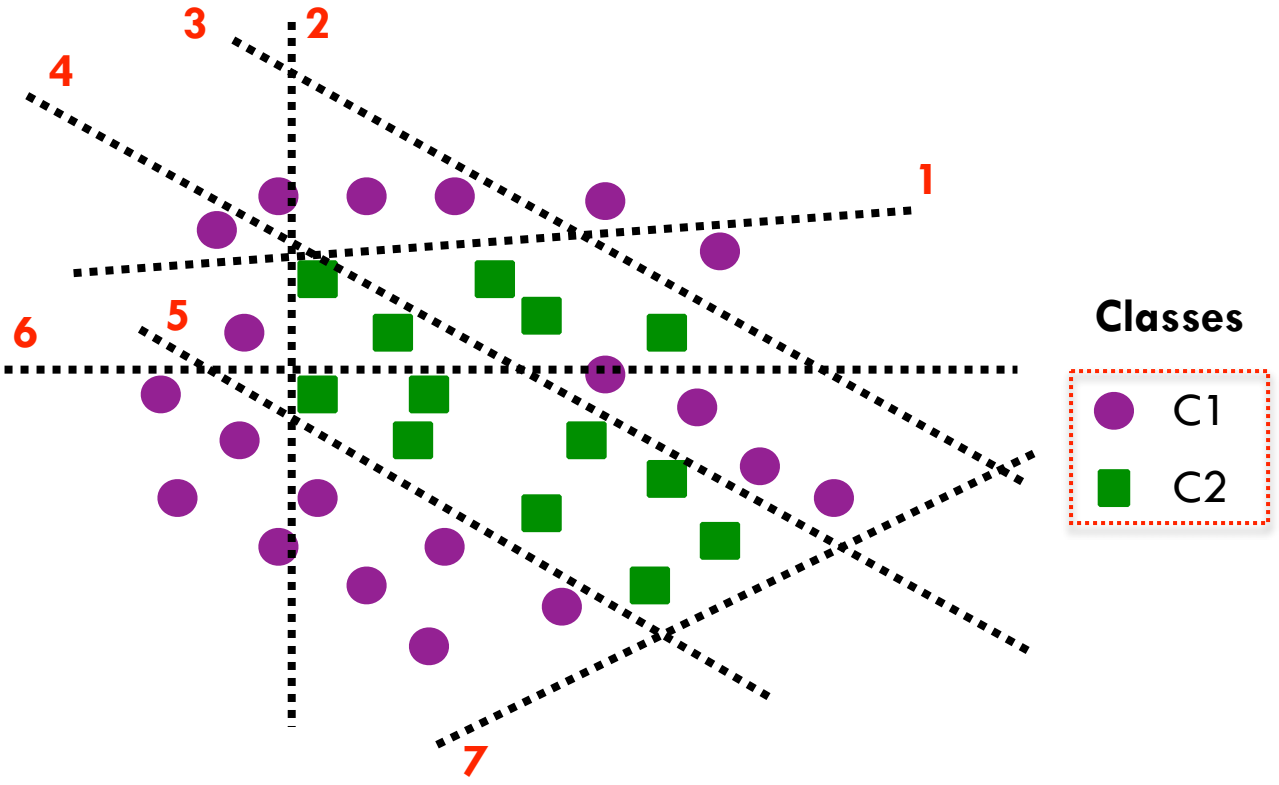
Exemplo



Exemplo



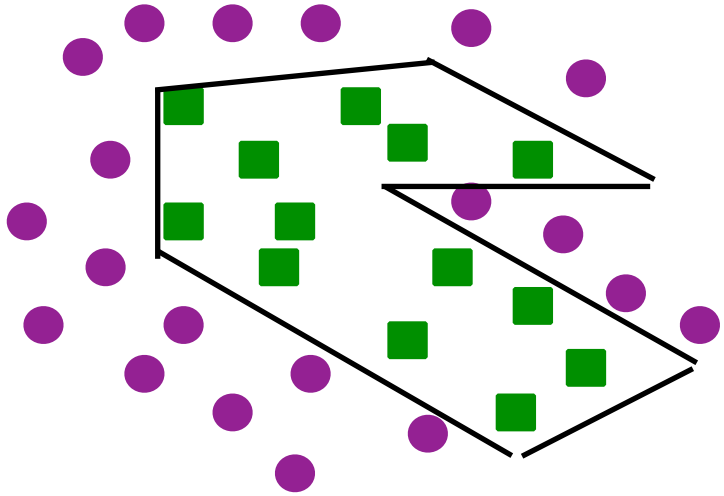
Exemplo

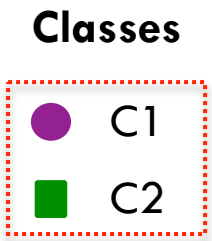


7 hiperplanos = 7 neurônios

Exemplo

No nosso exemplo hipotético, precisaríamos de uma região convexa com 7 hiperplanos para separar as classes corretamente.





7 hiperplanos = 1 região convexa

Roteiro

1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Formalização

- **MLP:**
  - τ = [x(n), d(n)] → exemplo de treinamento
  - $y_j(n)$ → sinal produzido na saída do neurônio j na camada de saída, estimulado por x(n), aplicado na camada de entrada

Formalização

- **MLP:**
  - τ = [x(n), d(n)] → exemplo de treinamento
  - $y_j(n)$ → sinal produzido na saída do neurônio j na camada de saída, estimulado por x(n), aplicado na camada de entrada
  - Sinal do erro produzido pelo neurônio j é:

$$e_j(n) = d_j(n) - y_j(n)$$

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

$$\varepsilon_j(n) = \frac{1}{2} e_j^2(n)$$

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

$$\varepsilon_j(n) = \frac{1}{2} e_j^2(n)$$

- Somando o erro de todos os neurônios da camada de saída:

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

$$\varepsilon_j(n) = \frac{1}{2} e_j^2(n)$$

- Somando o erro de todos os neurônios da camada de saída:

$$\varepsilon(n) = \sum_{j \in C} \varepsilon_j(n)$$

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

$$\varepsilon_j(n) = \frac{1}{2} e_j^2(n)$$

- Somando o erro de todos os neurônios da camada de saída:

$$\varepsilon(n) = \sum_{j \in C} \varepsilon_j(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)$$

Formalização

- O erro instantâneo produzido no neurônio j é dado por:

$$\varepsilon_j(n) = \frac{1}{2} e_j^2(n)$$

- Somando o erro de todos os neurônios da camada de saída:

$$\varepsilon(n) = \sum_{j \in C} \varepsilon_j(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)$$

C é o conjunto de neurônios de saída

Formalização

- Se houverem **N** exemplos de treinamento, o erro médio sobre todos os exemplos (risco empírico) é dado por:

Formalização

- Se houverem **N** exemplos de treinamento, o erro médio sobre todos os exemplos (risco empírico) é dado por:

$$\varepsilon_{avg}(N) =$$

Formalização

- Se houverem **N** exemplos de treinamento, o erro médio sobre todos os exemplos (risco empírico) é dado por:

$$\varepsilon_{avg}(N) = \frac{1}{N}\sum_{n=1}^{N}\varepsilon(n)$$

Formalização

- Se houverem **N** exemplos de treinamento, o erro médio sobre todos os exemplos (risco empírico) é dado por:

$$\varepsilon_{avg}(N) = \frac{1}{N}\sum_{n=1}^{N}\varepsilon(n) = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C} e_j^2(n)$$

Formalização

- Se houverem **N** exemplos de treinamento, o erro médio sobre todos os exemplos (risco empírico) é dado por:

$$\varepsilon_{avg}(N) = \frac{1}{N}\sum_{n=1}^{N}\varepsilon(n) = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C} e_j^2(n)$$

Erro quadrático médio da época considera todos os neurônios da camada de saída **C** e todos os exemplos do conjunto de treinamento (**N**)

Roteiro

1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Treinamento

Treinamento

1. **Treinamento Online**
2. **Treinamento em Batch**

Treinamento

1. **Treinamento Online**

Treinamento

1 **Treinamento Online**

- Ajuste dos pesos (W) ocorre após a apresentação de cada exemplo
- A busca no espaço multidimensional de pesos torna-se estocástica

Treinamento

1 **Treinamento Online**

- Ajuste dos pesos (W) ocorre após a apresentação de cada exemplo
- A busca no espaço multidimensional de pesos torna-se estocástica
- Menos suscetível a ficar preso em mínimos locais
- Quando há redundância, tira vantagem ao ajustar os pesos

Treinamento

1 **Treinamento Online**

- Ajuste dos pesos (W) ocorre após a apresentação de cada exemplo
- A busca no espaço multidimensional de pesos torna-se estocástica
- Menos suscetível a ficar preso em mínimos locais
- Quando há redundância, tira vantagem ao ajustar os pesos
- Detecta melhor pequenas mudanças nos dados de treinamento
- Simples de implementar / Bons resultados em problemas difíceis

Treinamento

2 **Treinamento em Batch**

Treinamento

2 **Treinamento em Batch**

- Apresenta todos os exemplos para a rede
- Só depois faz o ajuste de pesos. 1 ajuste para uma época completa

Treinamento

2 **Treinamento em Batch**

- Apresenta todos os exemplos para a rede
- Só depois faz o ajuste de pesos. 1 ajuste para uma época completa
- Estimativa mais precisa do vetor de gradientes
- Paralelização do processo de aprendizado

Treinamento

2 **Treinamento em Batch**

- Apresenta todos os exemplos para a rede
- Só depois faz o ajuste de pesos. 1 ajuste para uma época completa
- Estimativa mais precisa do vetor de gradientes
- Paralelização do processo de aprendizado
- Mais difícil de detectar mudanças pequenas nos dados
- Mais se há exemplos redundantes, não consegue identificar, pois ajusta os pesos para todos os exemplos de uma única vez

Treinamento

Em ambos os casos

- Vários experimentos, iniciando W com valores diferentes
- desempenho da rede é a média dos diferentes experimentos
- Análise da Curva de Aprendizado
- erro da época x época

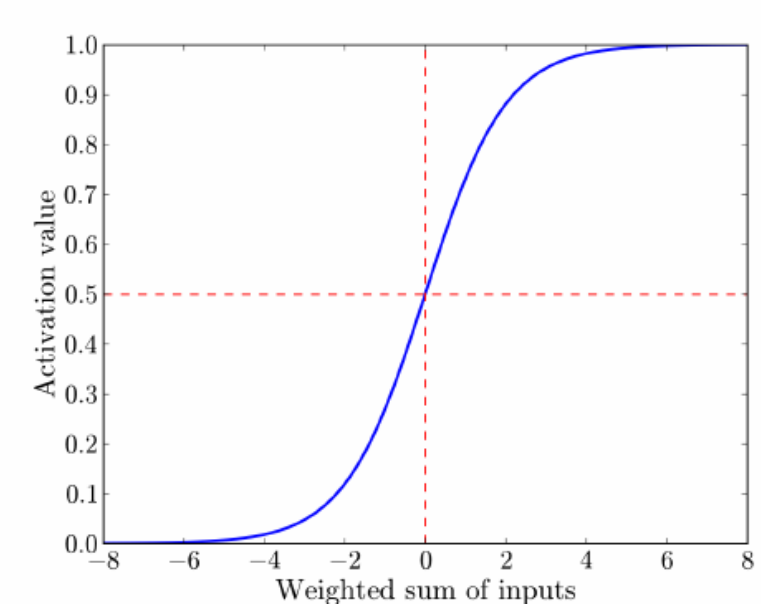
Funções de Ativação

- **ϕ deve ser diferenciável:**

Funções de Ativação

- **ϕ deve ser diferenciável:**
  - funções sigmoidal / logística

$$\varphi_j(v_j(n)) = \frac{1}{1 + exp(-av_j(n))}, \qquad a > 0$$

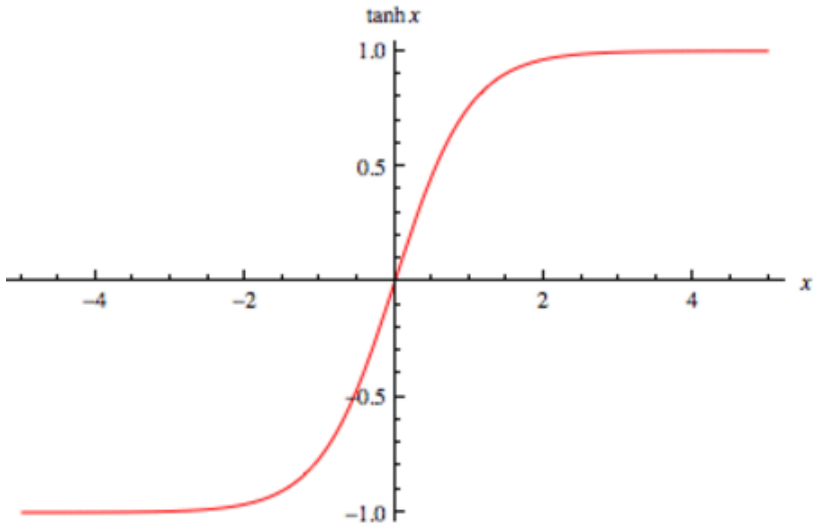


Funções de Ativação

- função tangente hiperbólica:

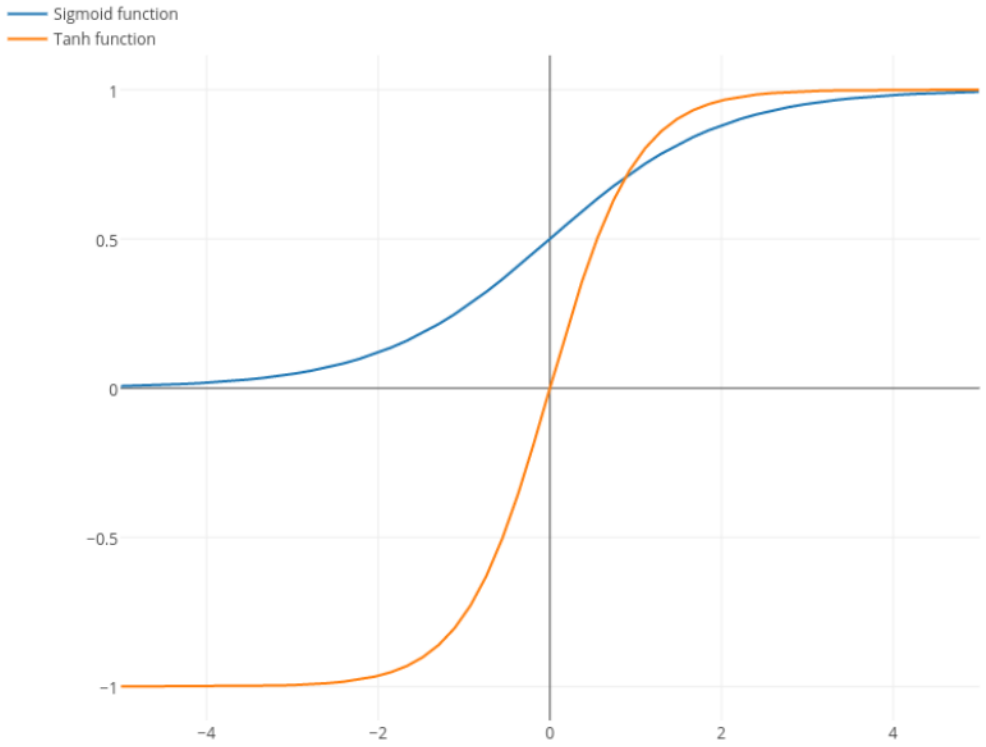
$$\varphi_j(v_j(n)) = a\tanh(bv_j(n))$$

  - a e b são constantes positivas
  - amplitude do sinal de saída: $-a \leq y_j \leq +a$



Funções de Ativação

- Comparativo entre as duas formas de funções de ativação
  - tanh x sigmoid



Resumindo ...

- MLP
  - Perceptron Multicamadas
  - neurônio J possui sinais de ativação e sinais de erro
  - função de ativação deve ser diferenciável
  - Treinamento leva em consideração todos os erros computados

Resumindo ...

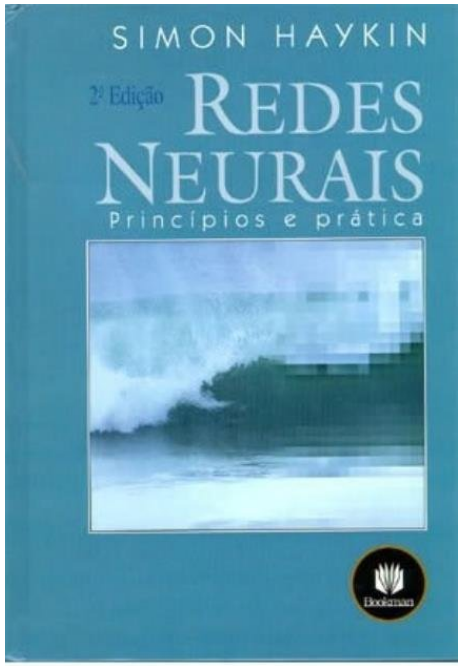
- MLP
  - Perceptron Multicamadas
  - neurônio J possui sinais de ativação e sinais de erro
  - função de ativação deve ser diferenciável
  - Treinamento leva em consideração todos os erros computados

- Próxima aula
  - *Backpropagation*
  - Exemplo(s)
  - Exercícios

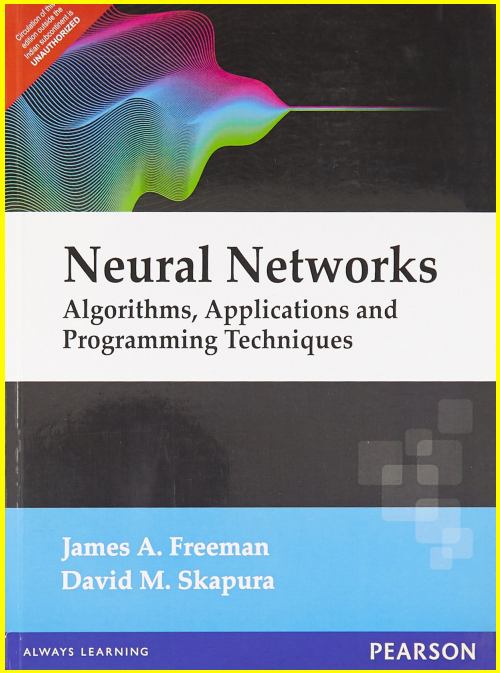
Roteiro

1. Introdução
2. Multilayer Perceptron
3. Exemplo
4. Formalização
5. Treinamento
6. Referências

Literatura Sugerida



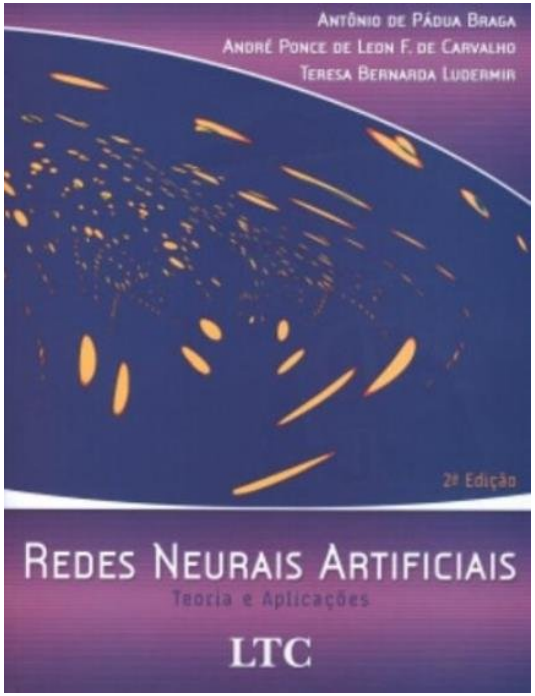
(Haykin, 1999)



(Freeman & Skapura, 1991)

Literatura Sugerida

[Faceli et al, 2011]



[Braga et al, 2007]

Perguntas?

Prof. Rafael G. Mantovani

rgmantovani@gmail.com