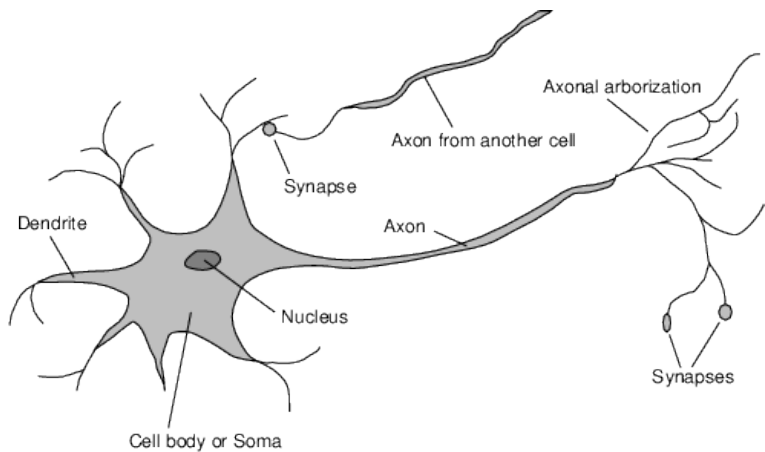


# Neural Networks

# Neural Networks

- Do ponto de vista computacional: métodos para representar funções usando redes de elementos aritméticos simples, e aprender tais representações através de exemplos.
- Do ponto de vista biológico: Modelo matemático para a operação do cérebro.
- **Neurônios:** Elementos aritméticos simples.
- **Redes Neurais:** conj de neurônios interligados.

# Neural Networks



# Neural Networks

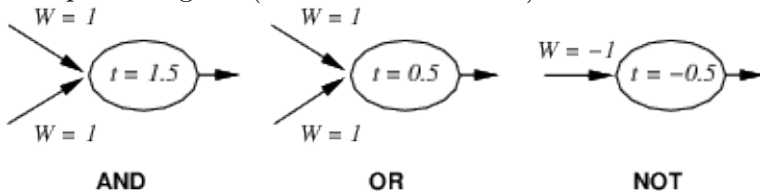
	Computer	Human Brain
Computational units	1 CPU, $10^5$ gates	$10^{11}$ neurons
Storage units	$10^9$ bits RAM, $10^{10}$ bits disk	$10^{11}$ neurons, $10^{14}$ synapses
Cycle time	$10^{-8}$ sec	$10^{-3}$ sec
Bandwidth	$10^9$ bits/sec	$10^{14}$ bits/sec
Neuron updates/sec	$10^5$	$10^{14}$

# Neural Networks

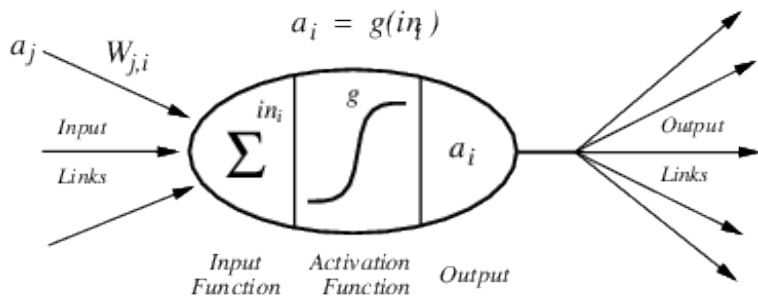
- Args a favor de redes neuronais:
  - ▶ Esperança de que um dispositivo possa ser construído de forma a combinar o paralelismo inerente do cérebro com trocas rápidas de contexto.
  - ▶ Redes neuronais podem prover um modelo de paralelismo massivo em contraste com paralelização de algoritmos tradicionais.
  - ▶ **Degradação gradual:** se algo errado acontecer, o desempenho decresce gradualmente.
  - ▶ Projetadas para serem treinadas utilizando algoritmos de aprendizagem indutiva.

# Neural Networks

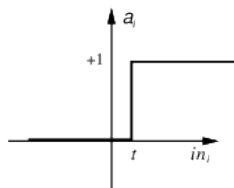
- Para construir uma rede: número de unidades, tipos de unidades, formato das conexões.
- Próximo passo: inicializar os pesos da rede, e treinar os pesos usando um algoritmo de aprendizagem aplicado a um conj de treinamento para a determinada tarefa implementada pela rede.
- A operação de unidades individuais pode ser comparada com portas lógicas (McCulloch and Pitts).



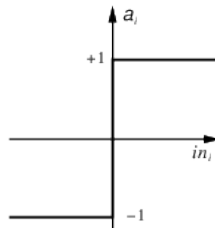
# Neural Networks



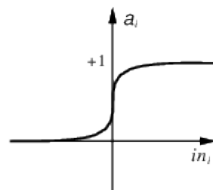
# Neural Networks



(a) Step function



(b) Sign function



(c) Sigmoid function



## Estruturas de Redes Neurais

- **feed-forward**: não há ciclos, grafo direcionado (DAG), links unidirecionais.
- **recorrente**: links podem formar topologias arbitrárias.
- Normalmente, redes feed-forward organizadas em camadas. Não há links entre nós da mesma camada, e links de uma camada para outra são unidirecionais.
- Computação pode prosseguir uniformemente da entrada para a saída.
- Em redes feed-forward também não temos memória (estados internos), visto que a informação não volta para o nó.

# Exemplos de Redes Neurais Recorrentes

- **Hopfield:** conexões bidirecionais entre os nós com pesos simétricos.
- todos os nós podem ser entradas ou saídas.
- função de ativação  $g$  produz -1 ou +1.
- Não encontra ótimos globais.

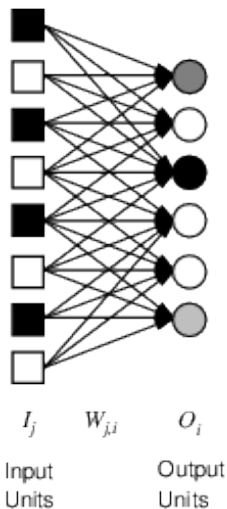
# Exemplos de Redes Neurais Recorrentes

- **Máquinas de Boltzmann:** também usa pesos simétricos, mas inclui unidades (escondidas) que não são entrada nem saída.
- Utilizam função de ativação estocástica, onde a prob da saída ser 1 é função dos pesos da entrada.
- Análogas ao algoritmo de “simulated annealing”, pois procura pela configuração que melhor se aproxima do conj de treinamento (tenta encontrar ótimos globais).
- Pode-se encontrar uma estrutura de rede que seja ótima através de algoritmos genéticos ou através de algoritmos do tipo “hill-climbing”.

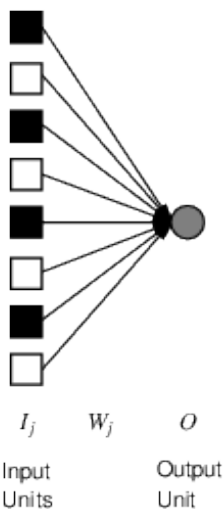
# Perceptrons

- Exemplos mais simples de redes feed-forward com apenas 1 camada.
- O que perceptrons conseguem representar?
- Função de maioria, por exemplo, onde a saída é 1 se mais da metade das  $n$  entradas forem iguais a 1.
- $O = Step_0(\sum_j W_j I_j) = Step_0(\mathbf{WI})$
- $W_j = 1$  e threshold  $t = n/2$ .
- Mesma função de maioria representada com árvores de decisão,  $O(2^n)$  nós. Com perceptrons, 1 unidade de saída e  $n$  pesos são suficientes para representar compactamente esta função.

# Perceptrons



**Perceptron Network**



**Single Perceptron**

## Redes de Hopfield: algoritmo

```
for i = 1 to n do
     $v_i = v_i^0$ ;
endfor
repeat
    for i = 1 to n do
         $v_i^- = v_i$ ;
         $v_i = \text{step}(\sum_{j=1}^n w_{ij}v_j + e_i - \Theta_i)$ 
    until  $v_i = v_i^-$  for all  $n_i \in N$ 
```

## Hopfield: exemplo 1 - Memória Associativa

- Rede vai estar carregada inicialmente com um endereço de memória
- Resultado esperado: que a rede convirja para aquele endereço
- Cada neurônio representado como um bit do endereço de memória
- matriz de pesos calculada por:  $(2b_i - 1)(2b_j - 1)$ , com  $b_i$  e  $b_j$  correspondendo aos bits  $i$  e  $j$  do endereço de memória
- matriz de pesos para o endereço 1001 (palavra de 4 bits)

bit	0	1	2	3
0	*	-1	-1	1
1	-1	*	1	-1
2	-1	1	*	-1
3	1	-1	-1	*

# Hopfield: exemplo 1 - Memória Associativa

- Resultados dependem da entrada e do limiar  $\theta$  (estado inicial=1001)

Entrada	Saída	
	$\theta = 0$	$\theta = -1$
0000	1001	1111
1001	1001	1001
1101	1001	1111
1010	1001	1001
1111	0110	1111
0110	0110	0110
0101	0110	1111
0111	0110	0110



# Hopfield: exemplo 1 - Memória Associativa

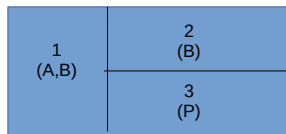
- Resultados dependem da entrada e do limiar  $\theta$  (estado inicial=01101001)

Entrada	Saída ( $\theta = 0$ )
00000000	01101001
00001100	01101001
11001100	01101001
11111111	10010110
10000001	10010110
01010101	10010110

## Hopfield: exemplo 2 - Coloração de mapas

- cada neurônio é representado por um par região-cor (1A significa, por exemplo, região A com cor amarela)
- pesos são atribuídos às ligações entre os neurônios, de forma que duas regiões adjacentes não possam ser pintadas com a mesma cor (por exemplo, peso -2)
- matriz de pesos para o mapa 1, com 3 regiões:

bit	1A	1B	2B	3P
0	*	-2	1	1
1	-2	*	0	1
2	-1	0	*	1
3	1	1	1	*



# Deep Learning using Neural Networks: Vision

