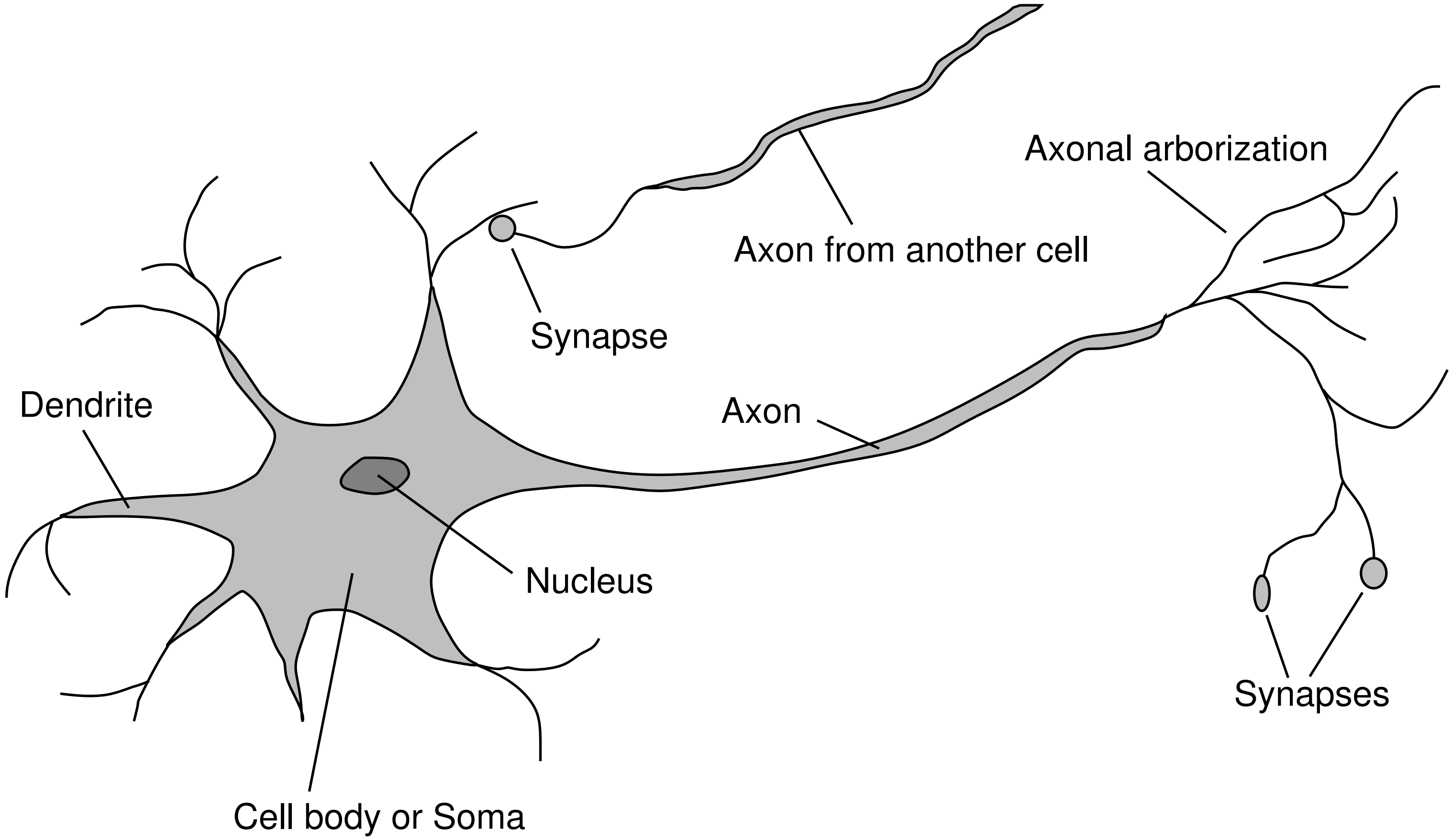
Cap. 19: Aprendendo em Redes Neuronais e Redes de Crenças

- 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 cerébro.
- **Neurônios**: Elementos aritméticos simples.
- **Redes Neuronais**: conj de neurônios interligados.



Axonal arborization
Axon from another cell
Synapse
Dendrite
Axon
Nucleus
Synapses
Cell body or Soma

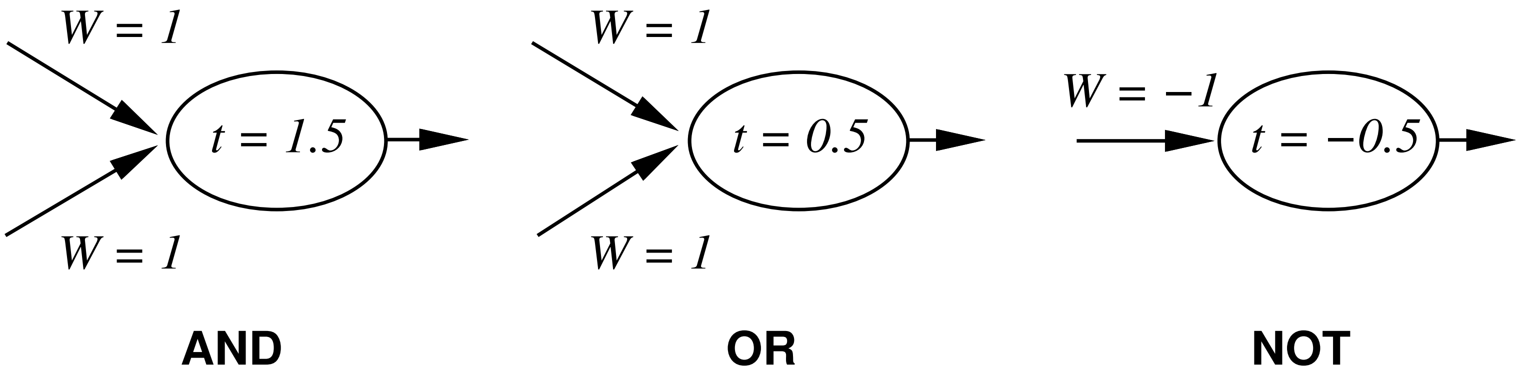
	Computer	Human Brain
Computational units	1 CPU, 10^5 gates	10^{11} neurons
Storage units	10^9 bits RAM, 10^{10} bits disk	10^{11} neurons, 10^{14} synapses
Cycle time	10^{-8} sec	10^{-3} sec
Bandwidth	10^9 bits/sec	10^{14} bits/sec
Neuron updates/sec	10^5	10^{14}

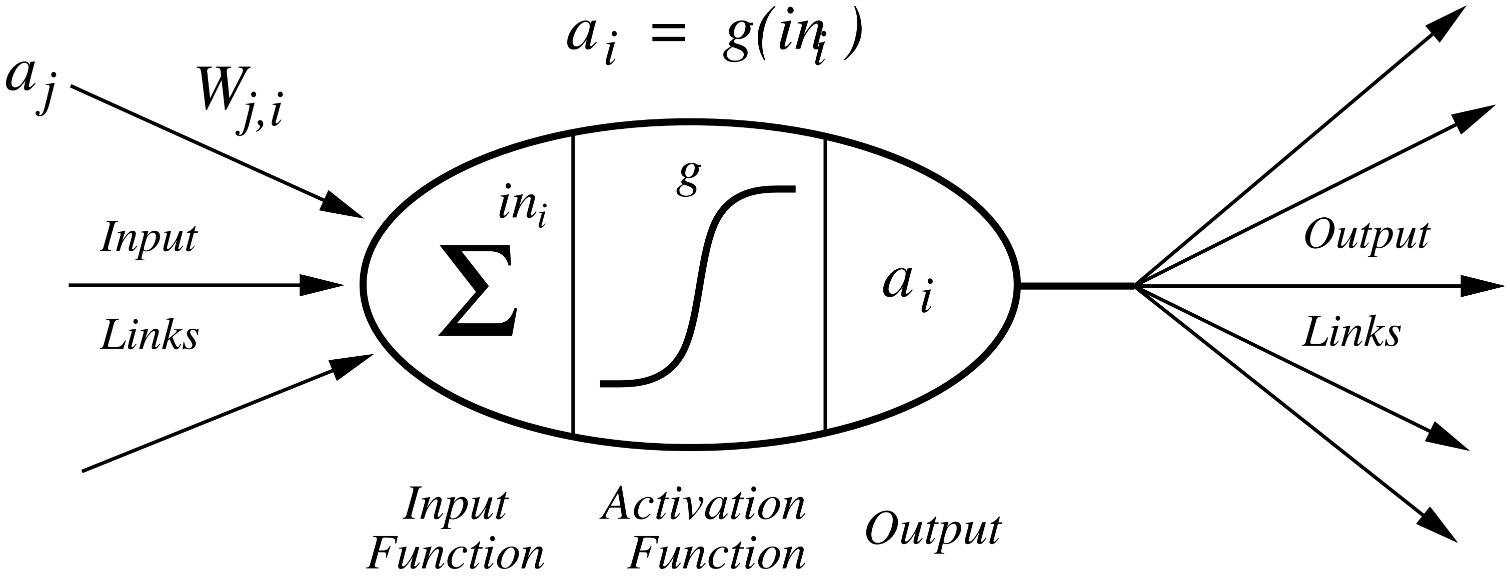
Cap. 19: Aprendendo em Redes Neuronais e Redes de Crenças

- 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 alguma coisa dá errada, o desempenho decresce gradualmente.
  - Projetadas para serem treinadas utilizando algoritmos de aprendizado indutivo.

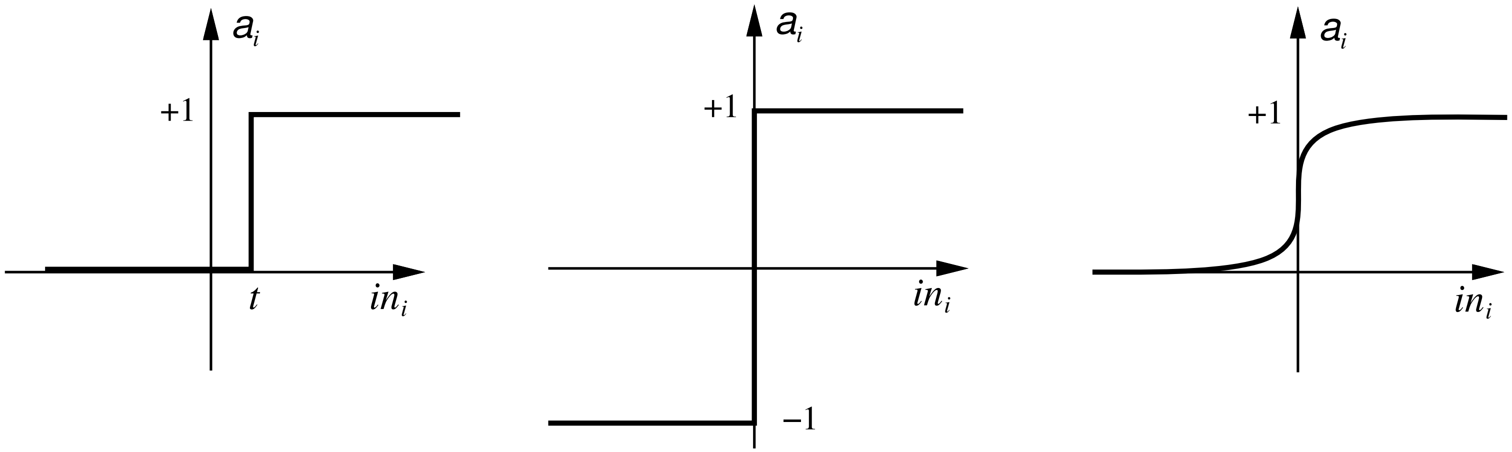
Redes Neuronais

- 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 aprendizado 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).


W = 1
W = 1
t = 1.5
AND
W = 1
W = 1
t = 0.5
OR
W = −1
t = −0.5
NOT



$a_i = g(in_i)$
a_j
$W_{j,i}$
Input
Links
in_i
Σ
g
a_i
Output
Links
Input Function
Activation Function
Output



(a) Step function **(b) Sign function** **(c) Sigmoid function**

Estruturas de Redes Neuronais

- **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 Neuronais Recorrentes

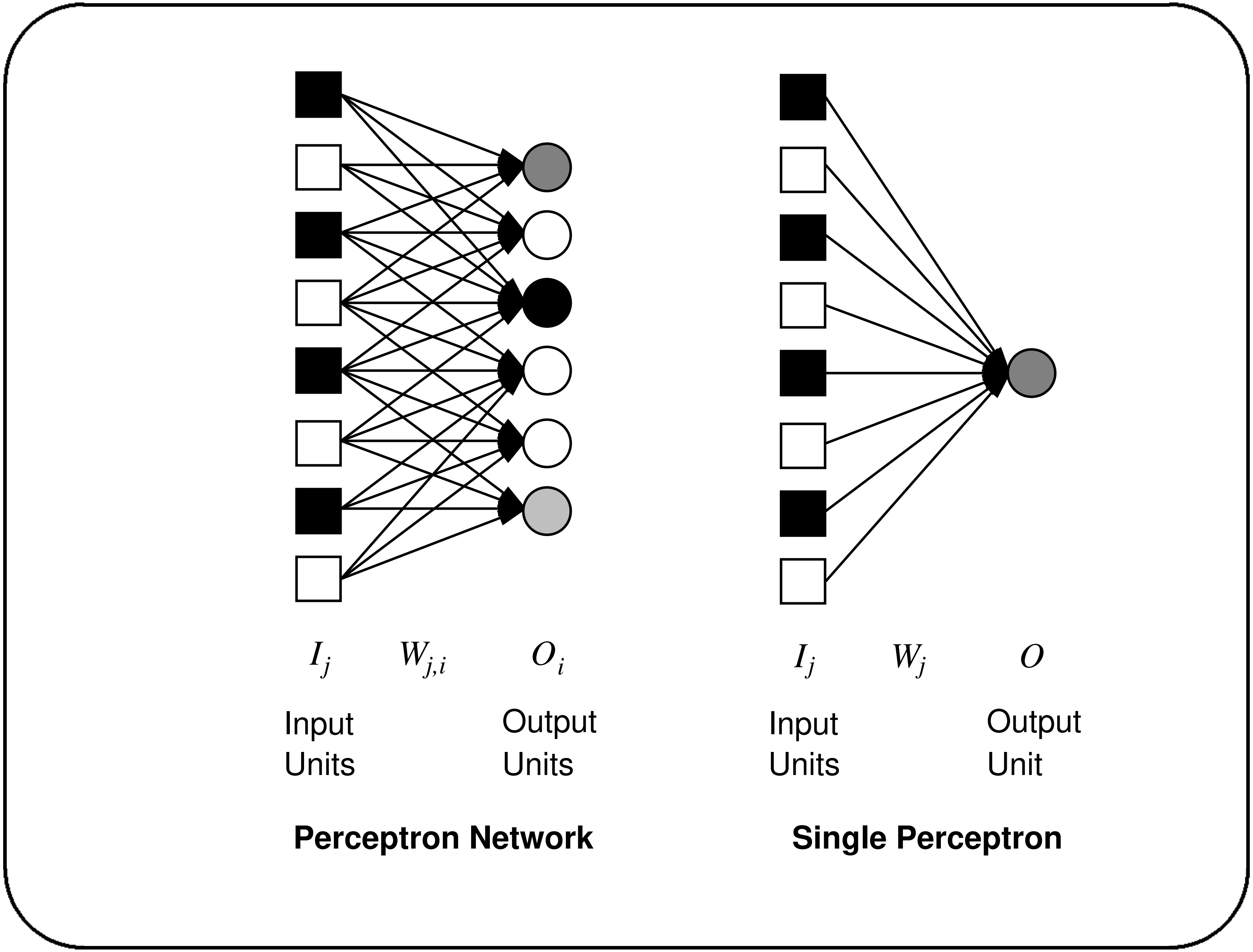
- **Hopfield**: conexões bidirecionais entre os nós com pesos simétricos.
- todos os nós podem ser entradas ou saídas.
- função de ativação g produz -1 ou +1.
- Não encontra ótimos globais.

Exemplos de Redes Neuronais 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.
- $O = Step_0(\sum_j W_j I_j) = Step_0(WI)$
- $W_j = 1$ 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.


I_j
$W_{j,i}$
O_i
Input Units
Output Units
Perceptron Network
I_j
W_j
O
Input Units
Output Unit
Single Perceptron