Perceptrons and Neural Networks

Manuela Veloso

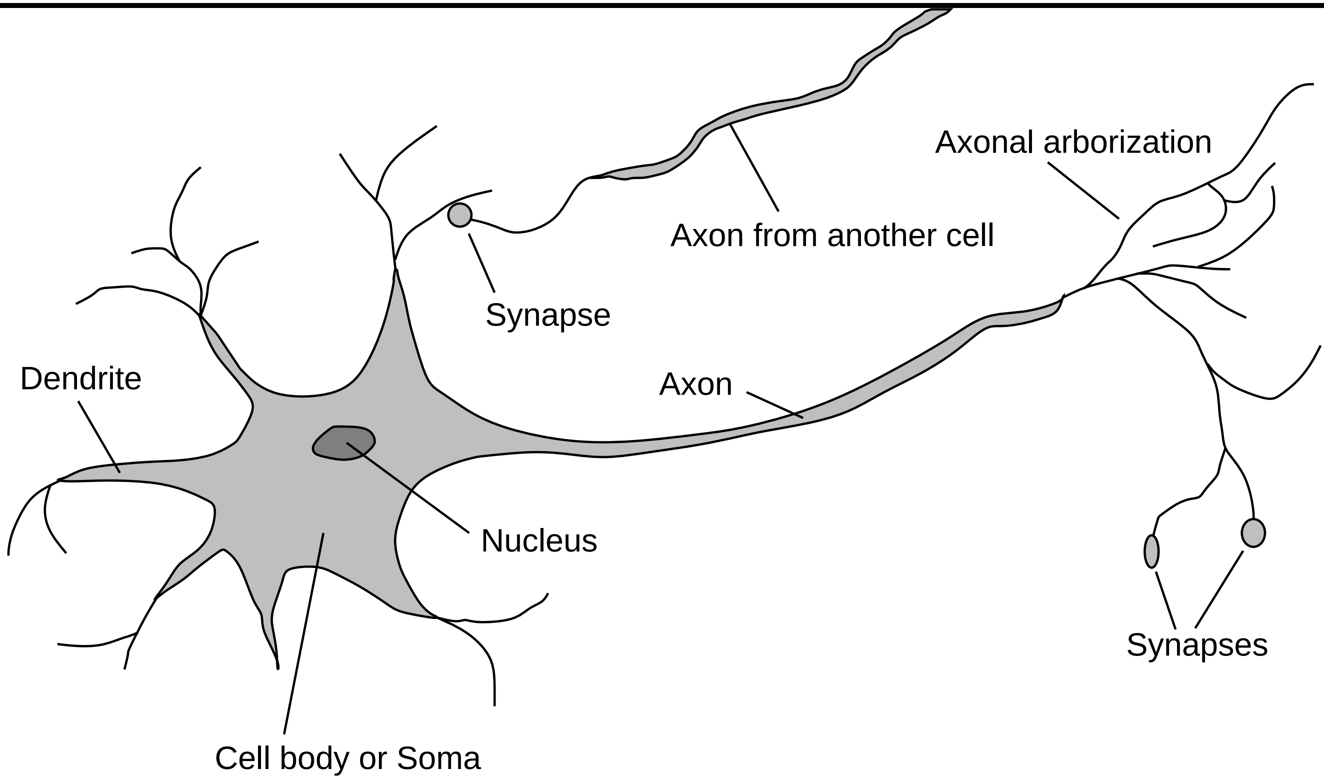
15-381 - Fall 2001

Motivation

- Beginnings of AI – chess, theorem-proving,... tasks thought to require "intelligence."
- Perception (language and vision) and common sense reasoning not thought to be difficult to have a machine do it.
- The human brain as a model of how to build intelligent machines.
- Brain-like machanisms - since McCulloch early 40s.
- Connectionism – building upon the architectures of the brain.

- Massively parallel simple neuron-like processing elements.
- "Representation" – weighted connections between the elements.
- Learning of representation – change of weights.
- Common sense – extremely well organized gigantic memory of facts – indices are relevant, highly operational knowledge, access by content.
- Classification tasks.

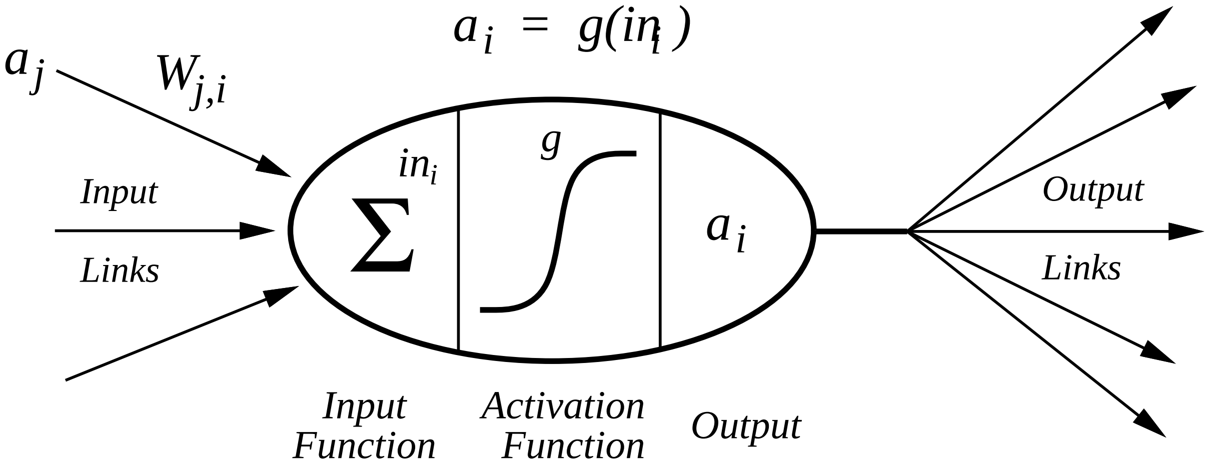
How the Brain Works



Memory

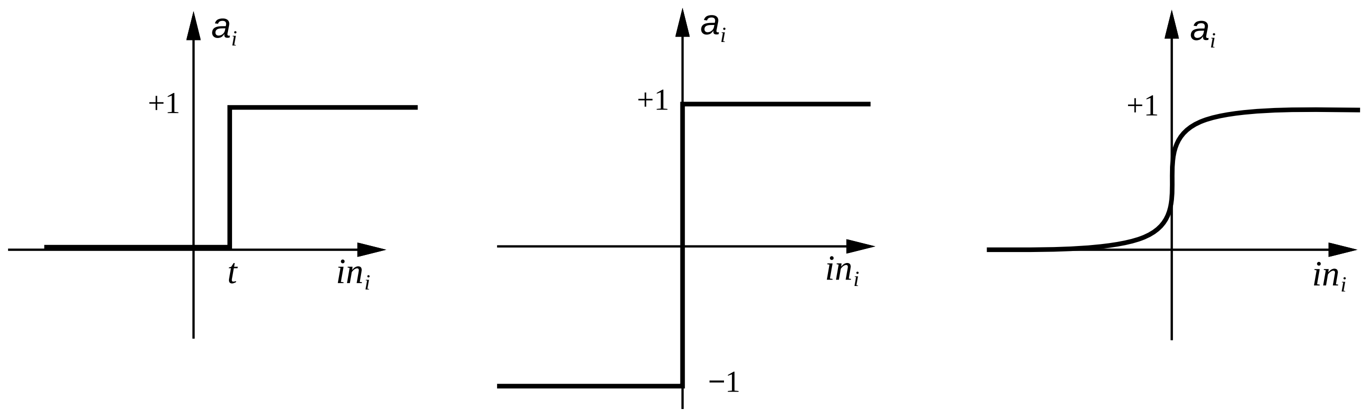
- 10^{11} neurons, 10^{14} connections

Main Processing Unit



$$a_i = g(\sum_j W_{j,i} a_j)$$

Different Threshold Functions



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

Learning Networks

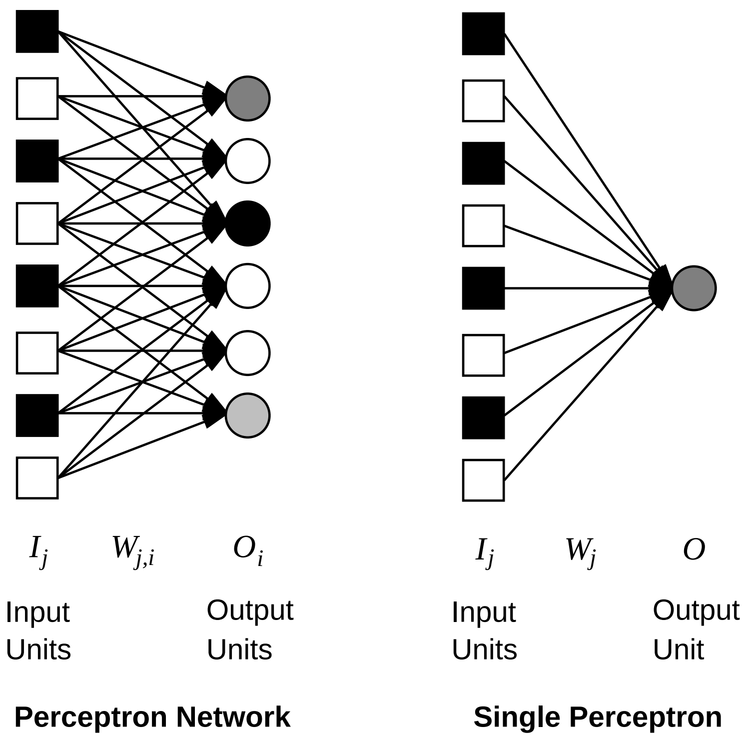
- How to acquire the right values for the connections to have the right knowledge in a network?

- Answer – learning: show the patterns, let the network converge the values of the connections for which those patterns correspond to stable states according to parallel relaxation.

- Neural networks that can learn: perceptrons, backpropagation networks, Boltzaman machines, recurrent networks, ...

Perceptrons

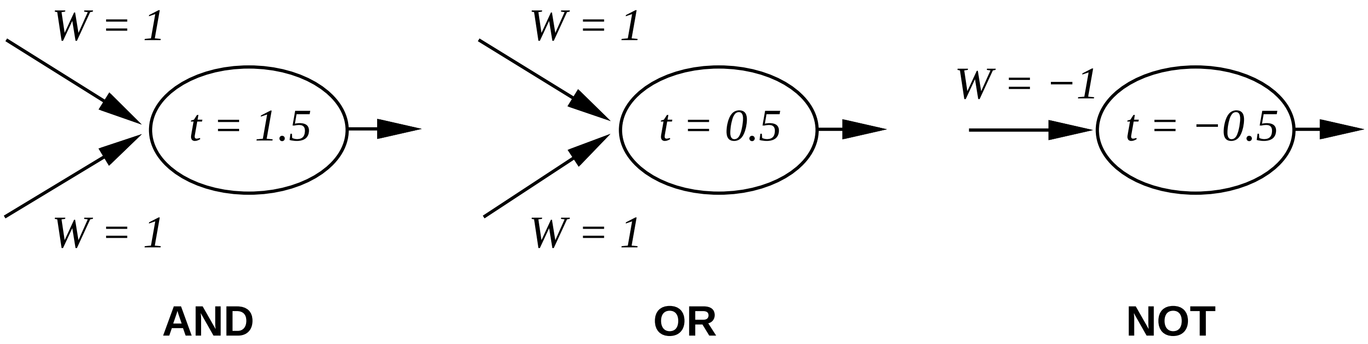
- Introduced in the late 50s – Minsky and Papert.
- Perceptron convergence theorem Rosenblatt 1962: Perceptron will learn to classify any linearly separable set of inputs.

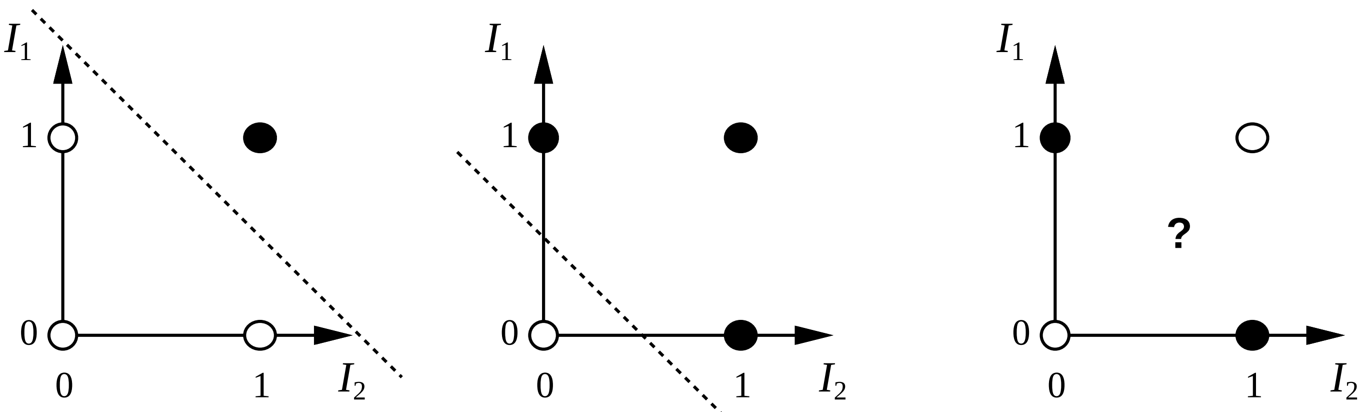


What Can a Perceptron Represent?

- and?
- or?
- not?
- xor?

Boolean Functions and Perceptrons





(a) I_1 **and** I_2 **(b)** I_1 **or** I_2 **(c)** I_1 **xor** I_2

Learning in Perceptrons

Rosenblatt 1960

Let y be the correct output, and $f(x)$ the output function of the network.

- Error: E $= y - f(x)$
- Update weights: $W_j \leftarrow W_j + \alpha x_j E$

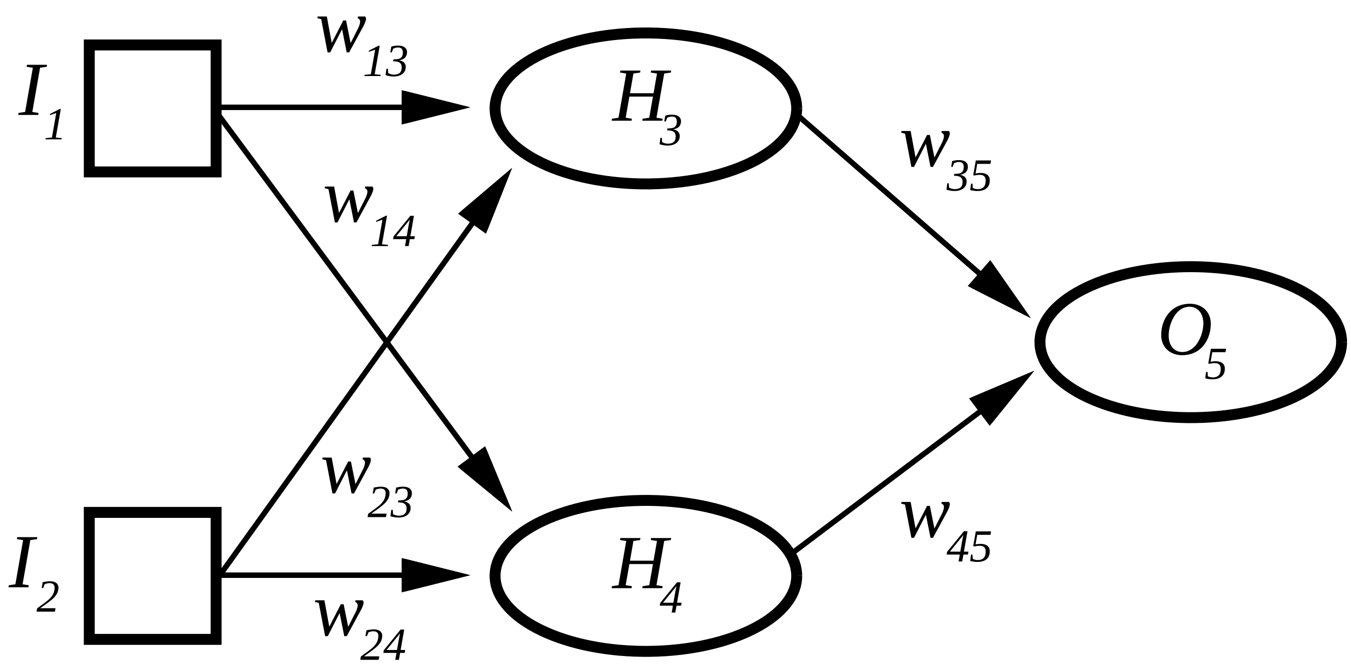
Discussion - Perceptrons

- Classifies a linearly separable set of inputs.
- Too restrictive – Anything else?
- Multi-layer perceptrons – found as a "solution" to represent nonlinearly separable functions – 1950s.
- Many local minima – Perceptron convergence theorem does not apply.
- 1950s - Intuitive Conjecture was: There is no learning algorithm for multi-layer perceptrons.
- Research in neural networks stopped until the 70s.

Backpropagation networks

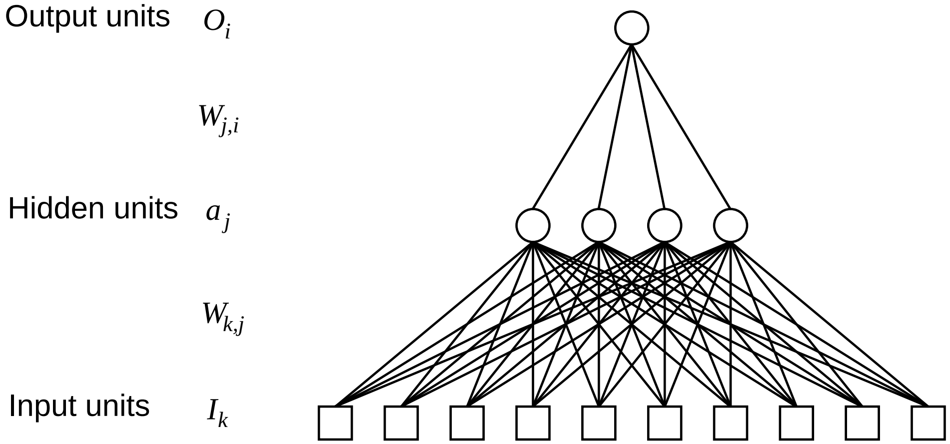
- Multi-layer perceptron.
- Goal again: Self-organizing neural networks – convergence to a stable structure.
- Weights change proportional to output errors.
- Gradient descent and chaining.
- After some training no more improvement.
- When to stop training?

Two-Layered Two-Unit Networks



xor?

Two-Layered Networks



$$O_i = g(\sum_j W_{j,i} a_j)$$

$$a_j = g(\sum_k W_{k,j} I_k)$$

Learning

- If g is differentiable, then we can take the derivative of the error with respect to each weight using the chain rule: $\frac{d}{dx}f(g(x)) = f'(g(x))g'(x)$.
- $Err_i = T_i - O_i$
- $W_{j,i} \leftarrow W_{j,i} + \alpha \times a_j \times Err_i \times g'(in_i)$
- If $\Delta_i = Err_i g'(in_i)$ then $W_{j,i} \leftarrow W_{j,i} + \alpha \times a_j \times \Delta_i$
- Error backpropagation - each hidden unit is responsible for some part of the error.
- $\Delta_j = g'(in_j) \sum_i W_{j,i} \Delta_i$
- $W_{k,j} \leftarrow W_{k,j} + \alpha \times I_k \times \Delta_j$

More machines

- Boltzaman machines - simulated annealing to make it “jump” out of local minima.
- High “temperatures” units have random behavior.
- Low “temperatures” - Hopfiel networks.
- Reinforcement learning - reward
- Unsupervised learning - output units “fight” for control of input – competitive learning.

Hopfield networks

- Hopfield – 1982 – a theory of memory.
- A network of processing elements – units – connected by weighted, symmetric connections.
- Weights are positive or negative.
- Elements are on or off, active or inactive.

```
procedure parallel relaxation
  while not-stable network
    pick a random unit
      let energy be the sum of the connections
        to all active neighbors
    if energy is positive
      then turn on the unit - unit becomes active
      else turn off the unit - unit becomes inactive
```

- Network is *stable* when no more units can change their state.

- Parallel relaxation is search.
- Possibly many local minima.

Discussion - Hopfield networks

- For a particular set of values of the connections, the network may have only a finite number of stable configurations.

- Network stores patterns. Values of the connections and topology of the network are in direct correspondence to the stable configurations – patterns.

- Weights of connections represent the “knowledge” encoded in a network.

- Partial pattern or slightly wrong pattern – Hopfield network converges to the closest stable pattern.

- Partial pattern – content-addressable memory.
- From a random initial configuration goes to closest stable state - local minimum.