TRÍ TUỆ NHÂN TẠO

Bài 13: Mạng thần kinh nhân tạo (1)

Nội dung

1. Một chút về quá trình phát triển của ANN
2. Perceptron
3. Huấn luyện một perceptron
   - Thuật toán huấn luyện Hebb
   - Thuật toán huấn luyện LMS
4. Sức mạnh của một perception

Phần 1

Một chút về quá trình phát triển của ANN

Sự phát triển của ANN

- 1943: Warren McCulloch & Walter Pitts công bố các nghiên cứu lý thuyết về hoạt động của mạng thần kinh tại ĐH Chicago
- 1949: Hebb xuất bản quyển "The Organization of Behavior"
- 1954: Minsky làm luận án tiến sĩ về một kiến trúc mạng thần kinh
- 1959: Rosenblatt xây dựng kiến trúc perceptron tại ĐH Cornell (ảnh bên)

Sự phát triển của ANN

- 1960: Widrow và Hoff giới thiệu thuật toán LMS
- 1961: Minsky viết quyển "Steps Toward Artificial Intelligent"
- 1969: Minsky và Papert viết một công trình đánh giá thấp khả năng của ANN
- 1982: Hopfield xây dựng mạng có nhớ, một dạng RNN
- 1982: Kohonen giới thiệu mạng tự tổ chức (SOM)
- 1986: Rumelhart, Hinton & Williams giới thiệu giải thuật lan truyền ngược lỗi
- 1989: Yann LeCun (Bell Labs) giới thiệu kiến trúc mạng CNN sơ khai

Sự phát triển của ANN

- 1997: Jürgen Schmidhuber giới thiệu LSTM (Long short-term Memory)
- 2006: Geoffrey Hinton giới thiệu thuật toán huấn luyện hiệu quả cho DBNs (deep belief networks)
- 2012: Nhóm của Geoffrey Hinton gây sốc ở cuộc thi ImageNet với một mạng CNN siêu lớn và chiến thắng cách biệt 10-15% so với đội đứng thứ 2
  - 500.000 nơ-ron, 60.000.000 tham số
  - Huấn luyện 1 tuần với 2 GPU NVidia GTX 580
- 2012: Google Brain sử dụng một CNN 1 tỉ tham số để tự học khái niệm “con mèo” từ 10 triệu ảnh không có nhãn

Sự phát triển của ANN

- 3/2016: Google DeepMind giới thiệu AlphaGo, sử dụng một thuật toán học sâu tăng cường để học chơi và thắng vô địch thế giới môn cờ Vây
- 12/2017: Google ra mắt AlphaZero, hệ thống tự học và chơi 3 game cờ hoàn toàn khác nhau (chess, go, shogi) ở trình độ mà chưa một phần mềm nào đạt được

- AI phát triển có nhanh hay không?

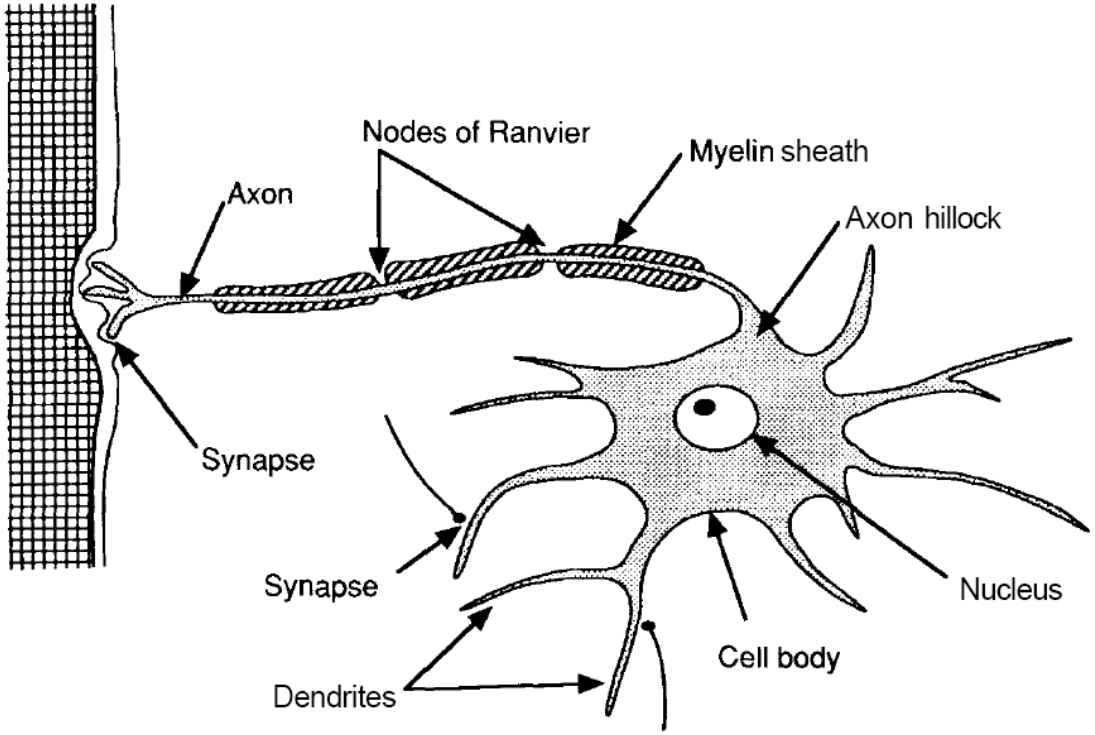
- KHÔNG, năm 1903 con người có chuyến bay đầu tiên (chỉ khoảng 37m), 66 năm sau con người đã đặt chân lên Mặt trăng. AI ra đời cách đây 62 năm.

Phần 2

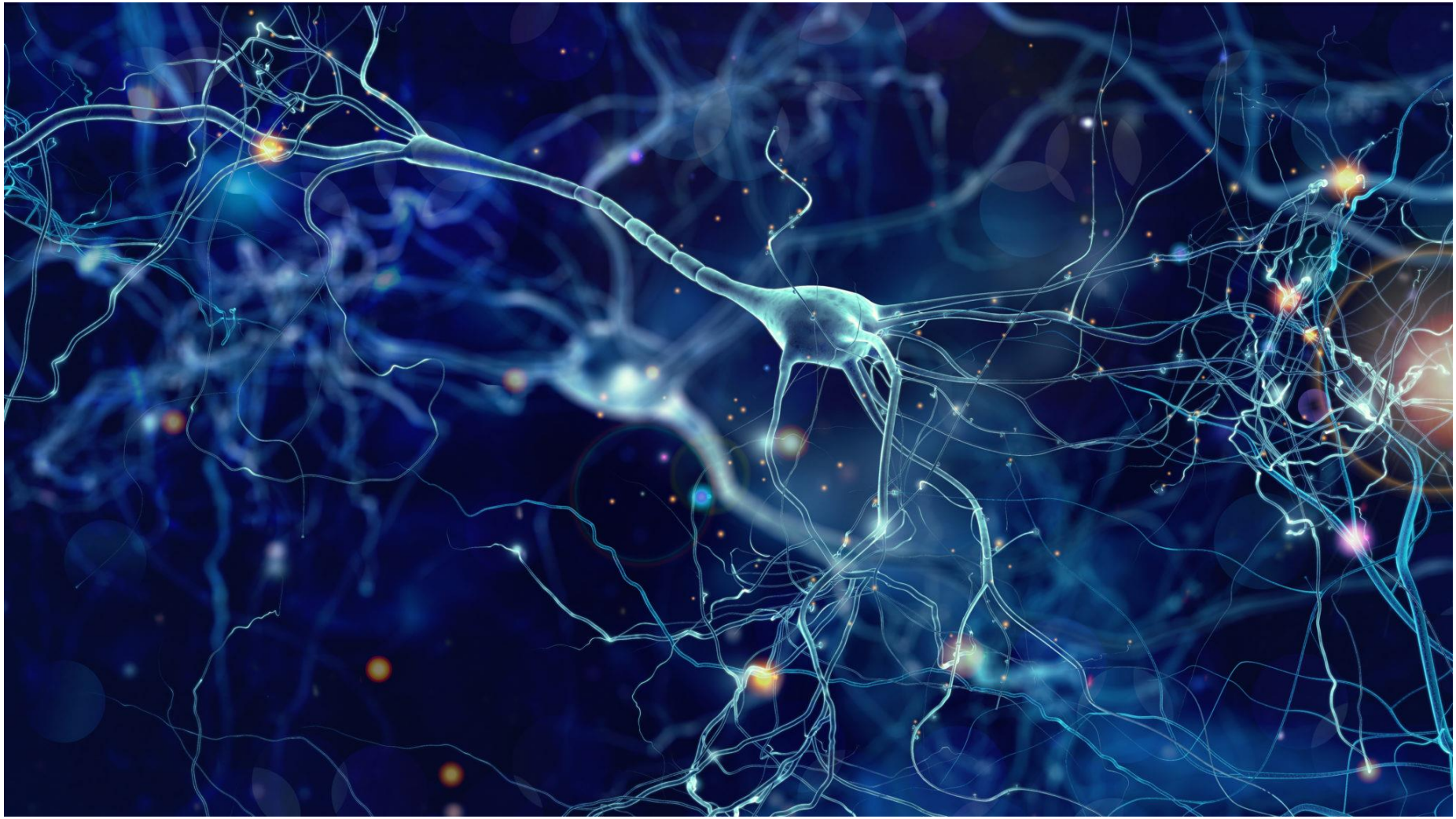
Perceptron

Tế bào thần kinh sinh học

- Soma: thân nơron, nhận hoặc phát xung động thần kinh
- Dendrites: dây thần kinh vào, đưa tín hiệu tới nơron
- Axon: đầu dây thần kinh ra, nối với dây thần kinh vào hoặc tới nhân tế bào của nơron khác thông qua khớp nối
- Synapse: khớp kích hoạt hoặc kích thích thông tin

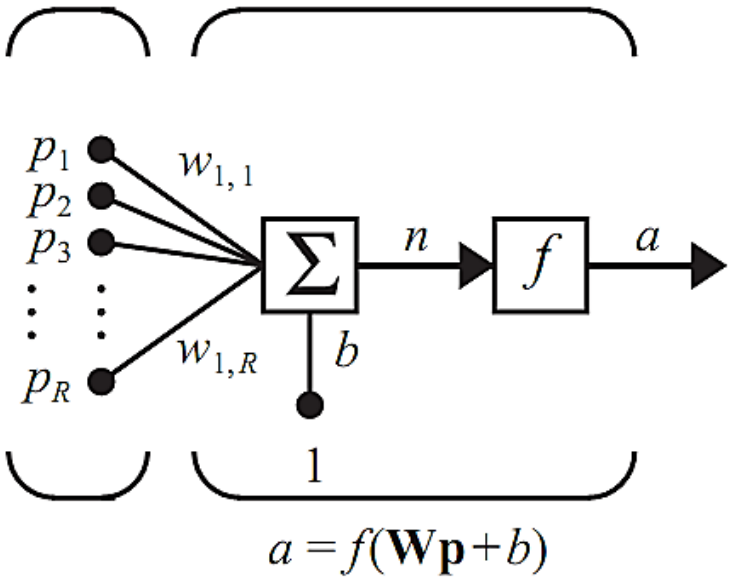


Tế bào thần kinh sinh học



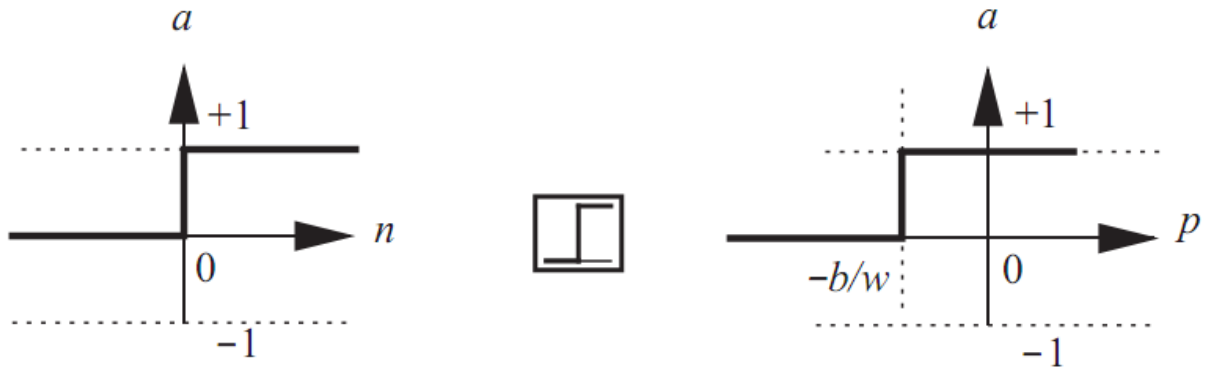
Perceptron

- Perceptron là mô phỏng của tế bào thần kinh sinh học
  - $[p_0, p_1, \ldots, p_r]$ là vector đầu vào với $p_0 = 1$
  - $[w_0, w_1, \ldots, w_r]$ là vector trọng số, $w_0 = b$
  - Bộ tổng: $n = p_0w_0+p_1w_1+\ldots+p_rw_r$
  - Đầu ra: a = hardlims(n) = (s>0) ? 1 : -1
  - Đích: y
- Hoạt động: nhận đầu vào, biến đổi thành đầu ra (hàm kích hoạt)
- Đây chỉ là một trong nhiều cách mô phỏng, các ý tưởng khác:
  - GRU
  - LSTM
  - …



Hàm kích hoạt

- Sử dụng để quyết định perception sẽ trả về kết quả như thế nào
  - Ví dụ như hàm hardlim: khi nào tổng các tín hiệu đầu vào dương thì hàm trả về 1, ngược lại trả về 0
- Thiết kế perception là thiết kế mở, cho phép ta có thể thay đổi hàm kích hoạt theo mục đích riêng, và có rất nhiều loại hàm kích hoạt khác nhau có thể sử dụng



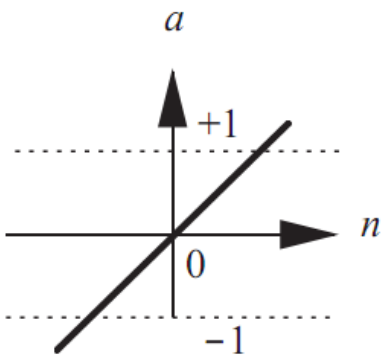
$a = hardlim(n)$

Hard Limit Transfer Function

$a = hardlim(wp + b)$

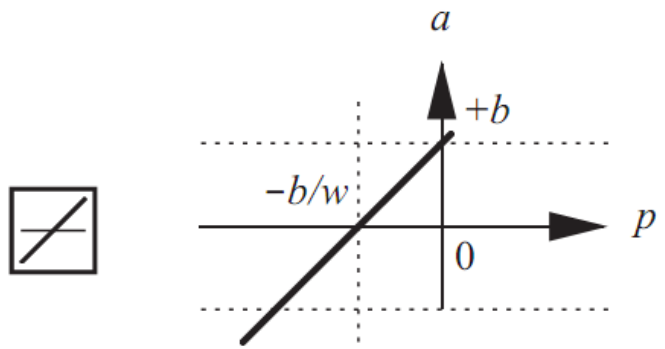
Single-Input *hardlim* Neuron

Hàm kích hoạt



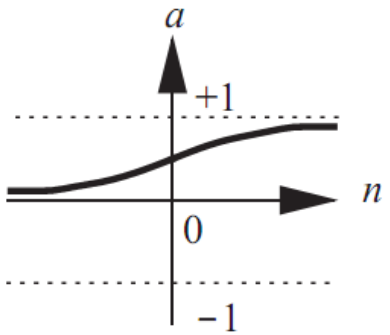
$a = purelin(n)$

Linear Transfer Function



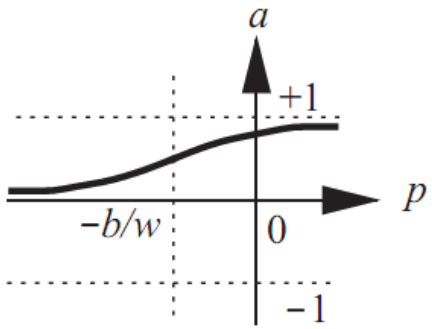
$a = purelin(wp + b)$

Single-Input *purelin* Neuron



$a = logsig(n)$

Log-Sigmoid Transfer Function



$a = logsig(wp + b)$

Single-Input *logsig* Neuron

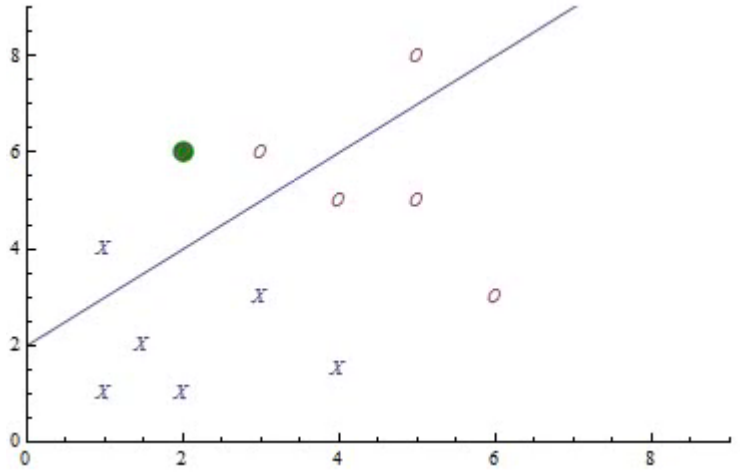
Phần 3

Huấn luyện một perceptron

Thuật toán huấn luyện Hebb

- Khởi tạo $[w_0, w_1, \ldots, w_n]$ ban đầu ngẫu nhiên
- Duyệt toàn bộ tập huấn luyện:
  - Nếu a = y (kết quả = đích): bỏ qua
  - Ngược lại: chỉnh w_i bởi $\Delta w_i = a * x_i$
- Lặp lại bước trên chừng nào còn có sai

- Đặc điểm:
  - Có tính minh họa
  - Hội tụ nếu có nghiệm
  - Không "tối ưu"



Thuật toán huấn luyện Hebb

Bước	w_0	w_1	w_2	x_1	x_2	f	y
1	-2	1	2	0.5	1.5	+1	+1
				-0.5	0.5	-1	-1
				0.5	0.5	-1	+1
2	-1	1.5	2.5				

Thuật toán huấn luyện LMS

- Khởi tạo [w_0, w_1,…, w_n] ban đầu bằng 0
- Duyệt toàn bộ tập huấn luyện:
  - Sai số e = y - f
  - Chỉnh w_i bởi $\Delta w_i = \eta * e * x_i$
- Lặp lại bước trên đến khi vector W không thay đổi

- Chú ý:
  - LMS chỉ tính tổng tín hiệu (f = s)
  - LMS hội tụ về siêu phẳng tối ưu ngăn cách hai tập mẫu (tổng bình phương sai số là nhỏ nhất)
  - Chọn giá trị η bằng bao nhiêu thì tốt? Nhỏ quá thì hội tụ chậm, lớn quá thì không ổn định

Phần 4

Sức mạnh của một perception

Sức mạnh của một perception

- Một perception mạnh hơn một cổng logic cơ bản
- Ví dụ một perception dùng hàm kích hoạt hardlim, 2 đầu vào, $w_1 = 1$, $w_2 = 1$
  - Chọn b = -1.5 ta được cổng AND
  - Chọn b = -0.5 ta được cổng OR