Introduction to Computational Neuroscience

Teach First Easter school

Stephen J Eglen

2018-04-11

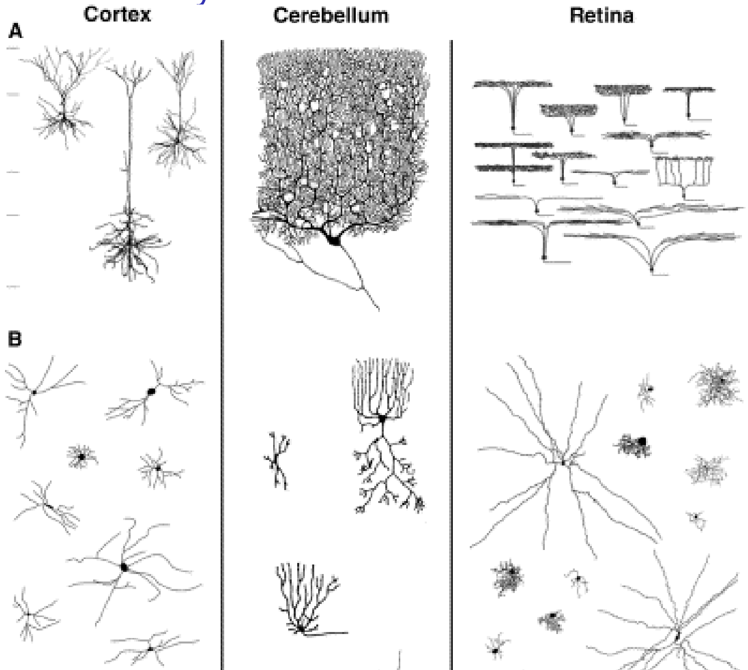
Aim of lecture

- Introduce you to Computational Neuroscience
- Specifically: Perceptron learning
- Please ask questions.

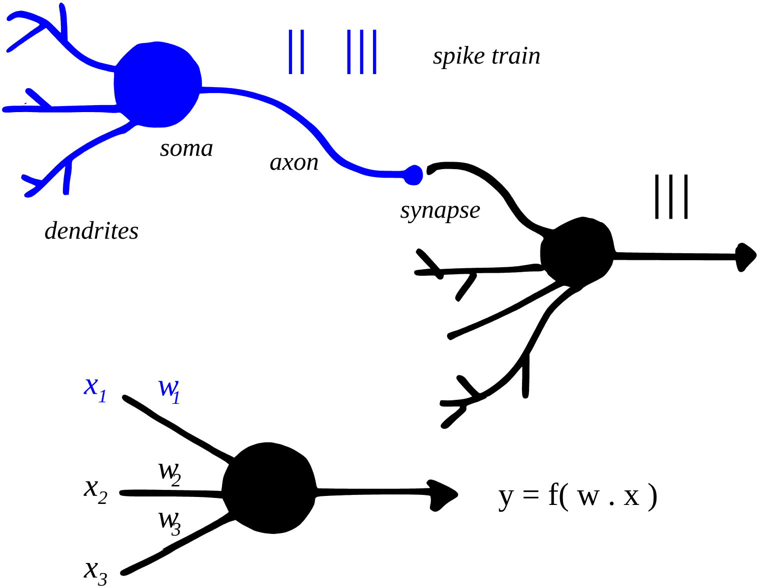
Slides available at: `http://bit.ly/eglen-tf18`

Understanding the brain

Neuronal diversity



Neurophysiology 101



$y = f(w \cdot x)$

Neurons as threshold devices

We say that a neuron fire if its inputs reach threshold.

e.g. total input for 2-d inputs: $z = w_1 x_1 + w_2 x_2$.

One firing rule:

$$y = \begin{cases} 1 & \text{if } z \geq \theta \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

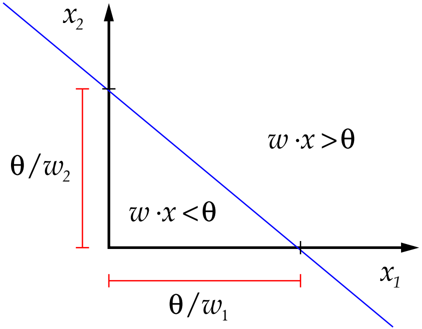
Perceptrons

Whether a perceptrons output is 0 or 1 depends on whether $\sum_{i=1}^{N} w_i x_i$ is less or greater than θ.

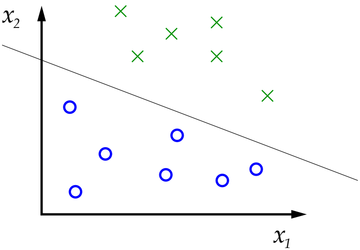
The equation $\sum_{i=1}^{N} w_i x_i = \theta$ defines a hyperplane in N-dimensional space.

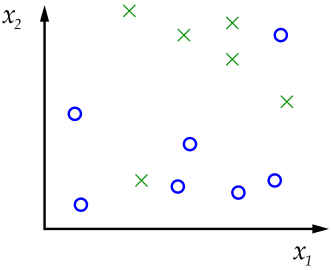
This hyperplane cuts the space in two.

$w_1 x_1 + w_2 x_2 = \theta \Rightarrow x_2 = \left(\frac{-w_1}{w_2}\right) x_1 + \frac{\theta}{w_2}$



Linearly separable problems





Learning involves adjusting the values of w and θ so that the decision plane can correctly divide the two classes.

What can they do?

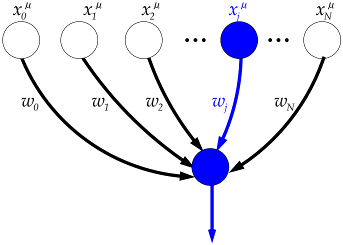
Generate simple building blocks, like logic gates.

1. OR
2. AND

Systematic “training” of weights

Intuitive and mathematical approach …

Intuitive explanation



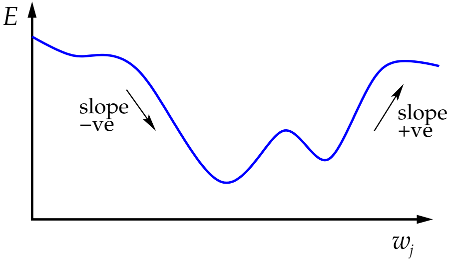
$y^\mu = \text{step}\left(\sum_{i=0}^{N} w_i x_i^\mu\right)$

Consider w_j's contribution to $\sum_i w_i x_i^\mu$ in different cases:

1. $y^\mu = t^\mu$ Perceptron has classified input μ correctly – change nothing
2. $x_j^\mu = 0$ Changing w_j will not affect the $\sum_i w_i x_i^\mu$ – change nothing
3. $x_j^\mu \neq 0, y^\mu < t^\mu$ The sum $\sum_i w_i x_i^\mu$ is too low – so increase it
4. $x_j^\mu \neq 0, y^\mu > t^\mu$ The sum $\sum_i w_i x_i^\mu$ is too high – so decrease it

The local rule: $\boxed{\Delta w_j = \epsilon \left(t^\mu - y^\mu\right) x_j^\mu}$

Perceptron learning rule



$$y = f(\mathbf{w} \cdot \mathbf{x}) \qquad \text{e.g.} \quad f(z) = 1/(1 + \exp(-z)), \quad f(z) = z$$

$$E = \frac{1}{2}(t - y)^2 \qquad t \text{ is target output}$$

$$\Delta w_j = -\epsilon \frac{\partial E}{\partial w_j} = -\epsilon \frac{\partial E}{\partial y} \frac{\partial y}{\partial w_j} = \epsilon (t - y) f'(\mathbf{w} \cdot \mathbf{x}) x_j$$

This is the method of gradient descent.

Training procedure.

0. Initialise the weight vector **w** to some random vector (or even the zero vector).
1. Select a pattern μ from the training inputs, $\mathbf{x}^\mu$ along with the target output, t^μ.
2. Present the input to the network to calculate the actual output, y^μ.
3. If the output matches the training signal, do nothing to the weight vector, else update the weights using the perceptron rule.
4. Repeat steps 1 to 3 until *convergence*.

Iterative algorithms / 1

- This training procedure is an *iterative algorithm*, similar in spirit to the Newton-Raphson method for root finding:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2}$$

- where starting from some initial guess x_0 you hopefully get closer and closer to the root, i.e. $f(x_n) = 0$.
- This method will not always find the root. How about our perceptron learning technique?

Demonstration

(Demonstration in matlab or similar.)

Perceptron convergence theorem (PCT)

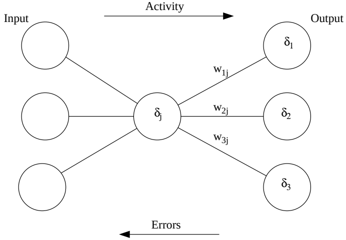
PCT states that if the two classes are linearly separable, the perceptron learning rule will find suitable weights in finite time.
We start from an initial weight vector $\mathbf{w}(0) = \mathbf{0}$ and show that the procedure must terminate if there is an ideal weight vector that separates the two classes.
(To do in class, time pending).

Limitations of perceptrons

Inability to solve “simple” problems like XOR.
Signalled the end of studying these in the 1960s.

Multilayer perceptrons: backpropagation algorithm

(Werbos PhD thesis, 1974)
Solving the **credit assignment problem** to generate "deltas" for hidden units.



But how to decide number of layers? Convergence?
Interest waned in 1990s, but now computers/GPUS are much faster . . .

Deep learning

Deep learning is everywhere. [New Science, April 2017]

`https://www.newscientist.com/article/2126738-google-uses-neural-networks-to-translate-without-trans`

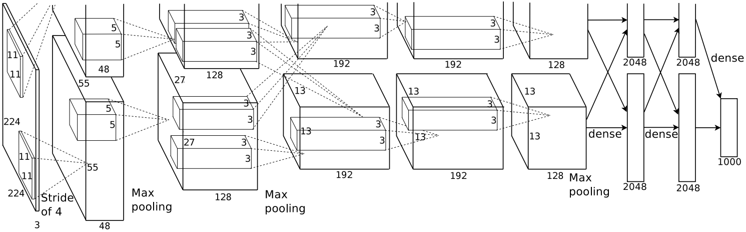
DAILY NEWS 4 April 2017

Google uses neural networks to translate without transcribing

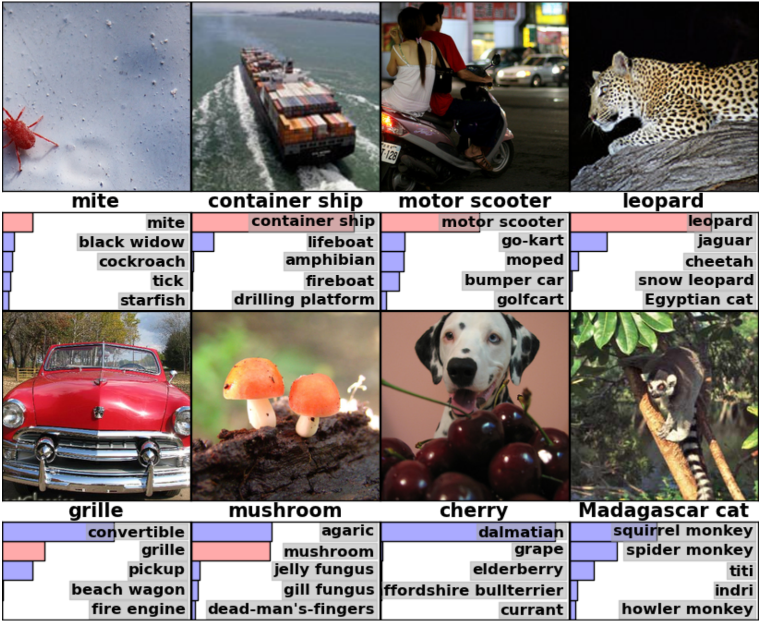
Image classification

2012 competition. 1,000 examples of 1,000 classes. Top-1 (37.5%) and top-5 (17.0%) error rate exceed previous state of the art. Won 2012 competition with top-5 test error rate of 15%; second had error rate of 26%.

Imagenet architecture



Imagenet performance



Reinforcement learning

Basic idea: limited feedback from a critic to say how well you did.
Two problems with reinforcement learning:

1. No specific signal on what the desired outputs should be. "It is as if each person in the United States tried to decide whether he or she had done a useful day's work by observing the gross national product [GNP] on a day by day basis" (Hinton, p220).
2. Reinforcement signal can occur long time after action. (e.g. one bad move in a game of chess, causing eventual loss of game.) "If … a person wants to how their behavior affects the GNP, they need to know whether to correlate today's GNP with what they did yesterday or with what they did five years ago." (Hinton, p220).

Hinton (1989) Connectionist learning systems, *Artificial Intelligence*, 40:185-234.

Classic control problem

Pole balancing problem.

https://www.youtube.com/watch?v=Lt-KLtkDlh8

Playing Go

Silver et al (2016) Nature 529:484–489

`http://doi.org/10.1038/nature16961`

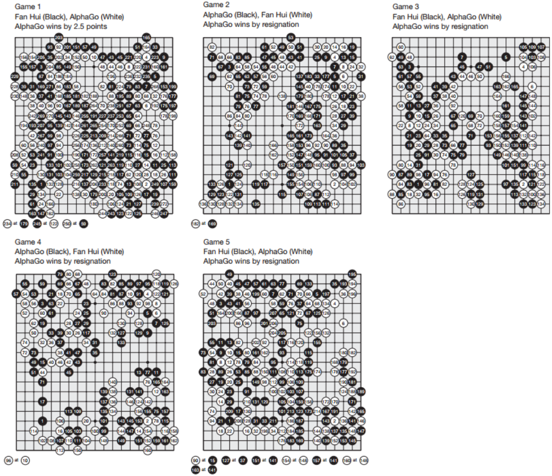
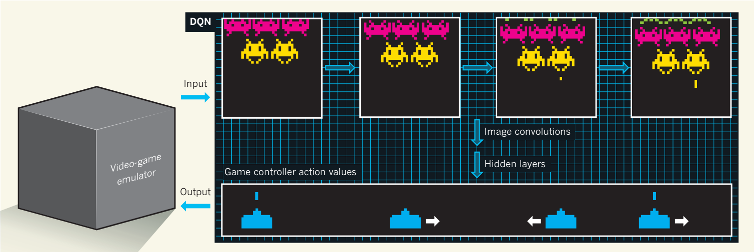


Figure 6 | Games from the match between AlphaGo and the European champion, Fan Hui. Moves are shown in a numbered sequence corresponding to the order in which they were played. Repeated moves on the same intersection are shown in pairs below the board. The first move number in each pair indicates when the repeat move was played, at an intersection identified by the second move number (see Supplementary Information).

Playing Atari video games

Mnih et al (2015) Human-level control through deep reinforcement learning. Nature 518:529–533.



System played better than professional human on 49 Atari 2600 games.
`https://www.youtube.com/watch?v=V1eYniJ0Rnk`

Summary

1. Maths/computing can be applied to interesting areas in the life sciences.
2. Interdisciplinary sciences are the future.
3. Careers in machine learning/data science/artificial intellgience are BIG.