Welcome to the introduction to computational cognitive modelling workshop!

Part 2: Introduction to artificial neural networks

Part 2: **Introduction to artificial neural networks**

Olivia Guest

Chris Brand

Nick Sexton

Nicole Cruz De Echeverria Loebell

What is a neural network?

A mathematical model

- Inspired by the nervous system
- A set of *units*, connected by *weights*
- The network *runs* by passing *activations* from the *input* (to the *hidden*) to the *output* units

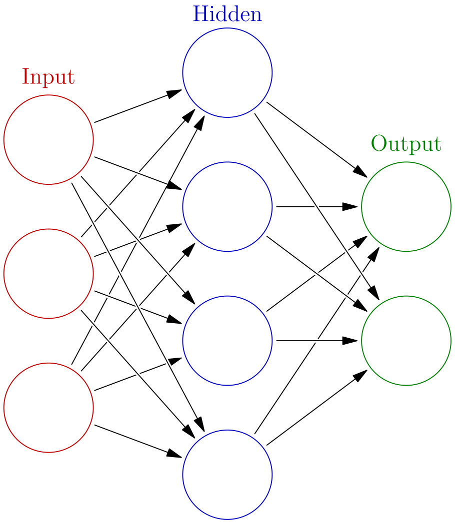


Figure: Glosser.ca / CC-BY-SA-3.0

Why use artificial neural networks for modelling?

Some aspects of their behaviour are like their namesake!

- Learn pretty much any input-output data
- Uncover rules on their own about data
- Generalise from what they have learnt
- Cope with noise and damage

How does an artificial neural network run?

By using maths, predictably!

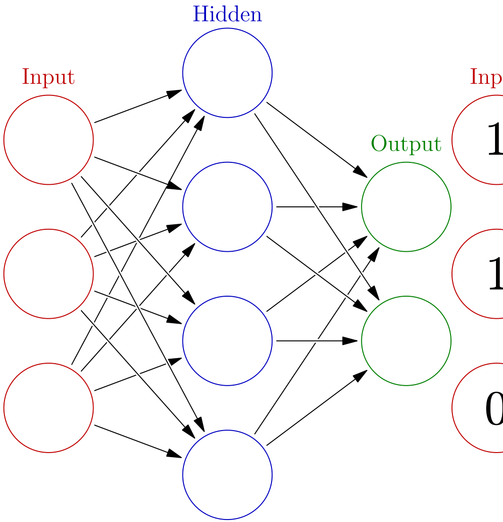
1. Input units are set to a *pattern*

2. Calculate hidden units' states:

$$
\begin{array}{rcll}
1 \times 0.5 & = & 0.5 & \\
1 \times 0.0 & = & 0.0 & \\
0 \times 0.8 & = & 0.0 & + \\
\hline
 & & 0.5 &
\end{array}
$$

3. Same for output units:

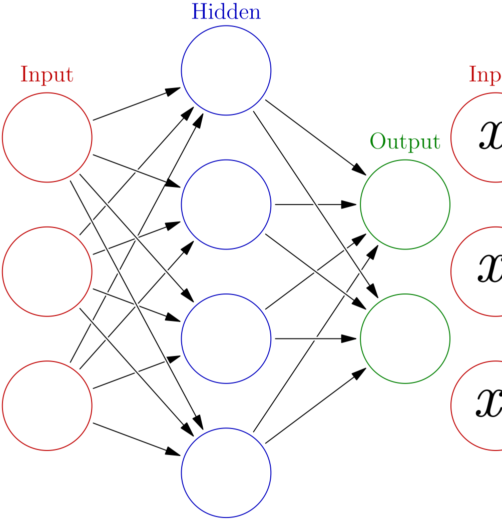
$$
\begin{array}{rcl}
0.5 \times 0.25 & = & 0.125 \\
0.3 \times 1.5 & = & 0.45 \\
1.6 \times -0.3 & = & -0.48
\end{array}
$$



How does an artificial neural network run?

By using maths, predictably!

- But programmers are *lazy*! General names save time
- input units: x_i
- hidden units: a_j
- output units: y_k
- connection weights: w_{ij}
- subscripts
  general: $ijklm...$
  specific: $12345...$



How does an artificial neural network run?

By using maths, predictably!

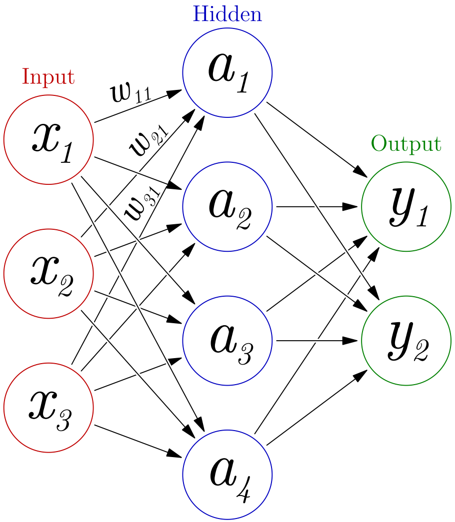
We use general names to write a general equation:

$$a_i =$$

$$a_i = \qquad x_j \times w_{ji}$$

$$a_i = \sum_{j=1}^{N} x_j \times w_{ji}$$

$$a_i = f\left(\sum_{j=1}^{N} x_j \times w_{ji}\right)$$



How does an artificial neural network run?

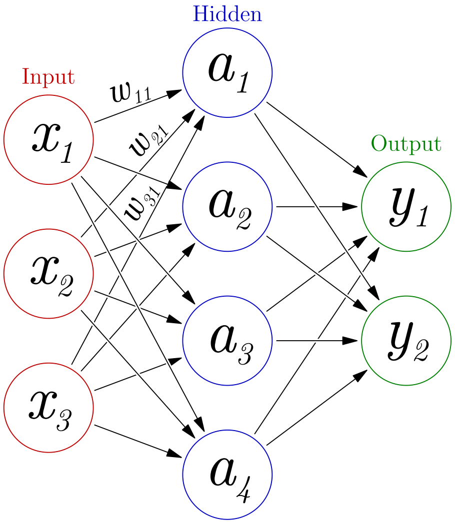
By using maths, predictably!

We use this equation by replacing *iterators* i and j:

$$a_i = f\left(\sum_{j=1}^{N} x_j \times w_{ji}\right)$$

$$a_{1} = f\left(\sum_{j=1}^{N} x_j \times w_{j1}\right)$$

$$a_{1} = f\left(\sum_{j=1}^{3} x_j \times w_{j1}\right)$$



How do networks learn?

Cunning!

- Many options: Hebbian learning, back-propagation of error, Boltzmann machine learning, self-organising map algorithm, etc.
- All learning algorithms work by changing the connection weights
- Learning can be divided into *supervised*, *unsupervised*, and *reinforcement*

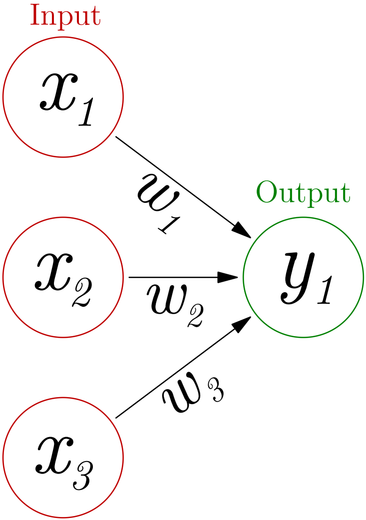
Hebbian learning

A very simple learning rule

"Cells that fire together, wire together"
— Carla Shatz

$$\Delta\, w_i = \eta \times x_i \times y_j$$

which means each weight is changed by a small in/decrement for every pattern



Hebbian learning

"Cells that fire together, wire together" — Carla Shatz

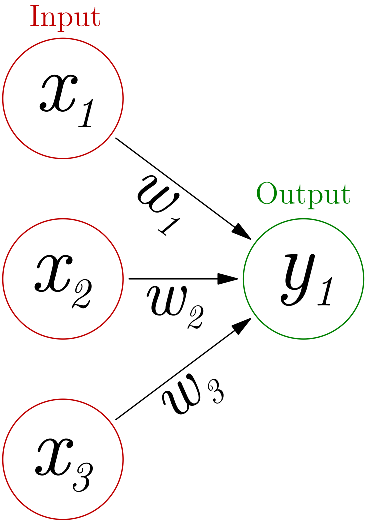
Hebb's rule is simple, but *very unstable*!

$$\Delta w_i = \eta \times x_i \times y_j$$

$$\Delta w_1 =$$
$$0.15\eta 0.5 \times x_i x_1 1.0 \times y_j y_1 0.3$$

$$\textbf{new } w_1 = \textbf{old } w_1 + \Delta w_1$$
$$\textbf{new } w_1 = 0.0 + \Delta w_1$$
$$\textbf{new } w_1 = 0.0 + 0.15$$
$$\textbf{new } w_1 = 0.15 \quad w_1 = 0.15$$

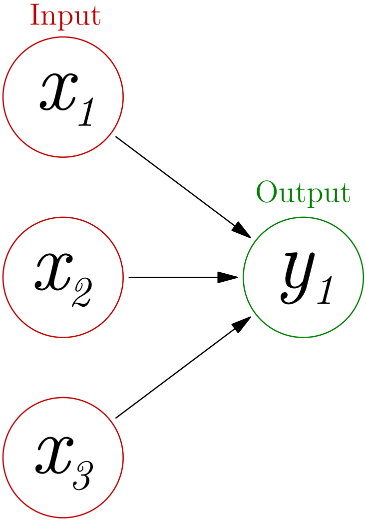
$w_1 = 0.15 +$ something positive $\quad w_1 = 0.15 +$ something positive +



The perceptron

A simple classifier

- Created in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt
- Linear classifier
- Simplest form of feedforward network



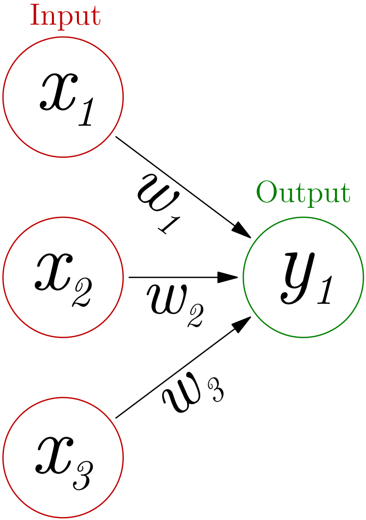
How does the perceptron learn?

Maths again!

1. Initialise weights
2. Run network using:

$$y_j = f\left(\sum_{1}^{N} w_i \times x_i\right)$$

same as always!



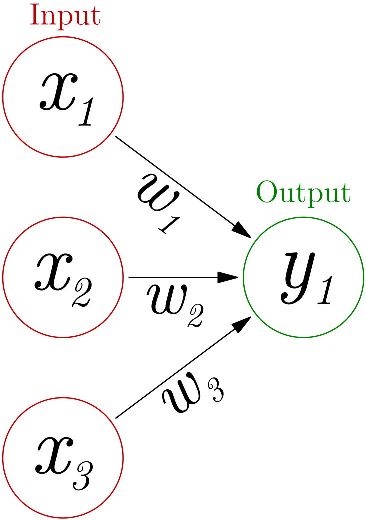
How does the perceptron learn?

Maths again!

1. Initialise weights
2. Run network
3. Update weights using:

$$\Delta w_i = \eta \ (d_j - y_j) \ \ y_j \ \times x_i$$

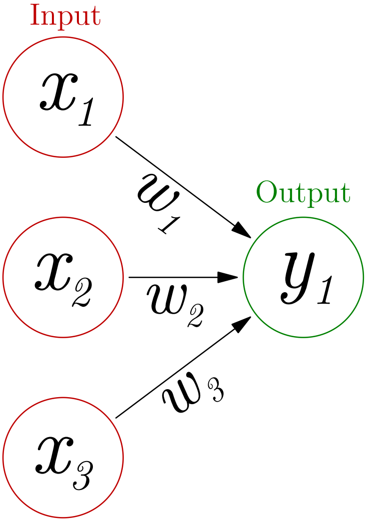
where d is what we want y to be given x, and η is the learning rate.



How does the perceptron learn?

Maths again!

1. Initialise weights
2. Run network
3. Update weights
4. Repeat 2 and 3
5. When do we stop?



The end

Time to program a perceptron!

Oh, and join our mailing list so we can send you stuff:
https://groups.google.com/d/forum/introcompcog