# Connectionism

http://compcogscisydney.org/psyc3211/

A/Prof Danielle Navarro
d.navarro@unsw.edu.au
compcogscisydney.org

# Structure of the "categorisation" lectures

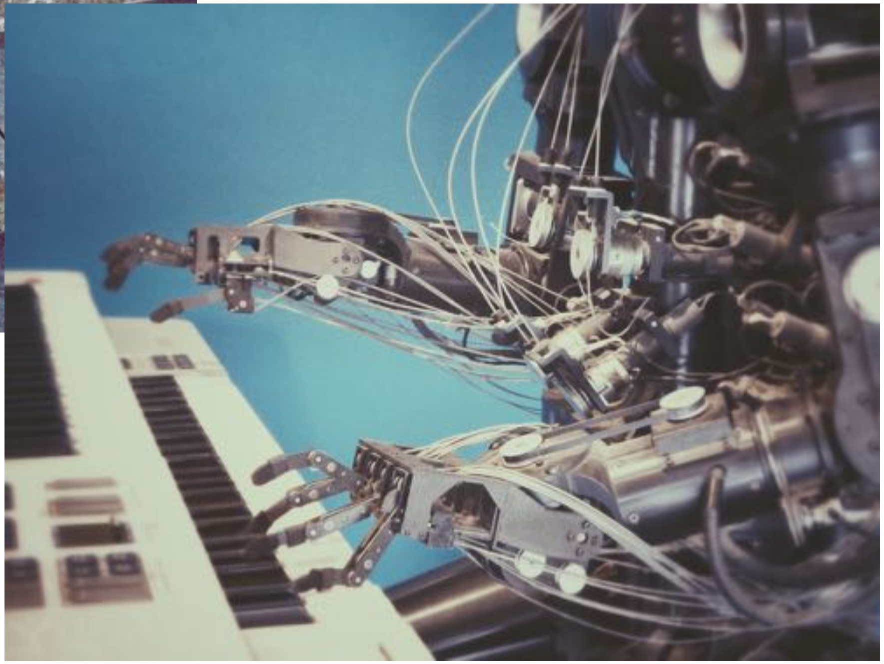
- L1: Connectionism
- L2: Statistical learning
- L3: Semantic networks
- L4: Wisdom of crowds
- L5: Cultural transmission
- L6: Summary

Yes there's some categorisation in these lectures, but not as much as you'd think. The name of this lecture series comes from the fact that I'm filling in for Brett Hayes and his lectures were more categorisation focused ☺

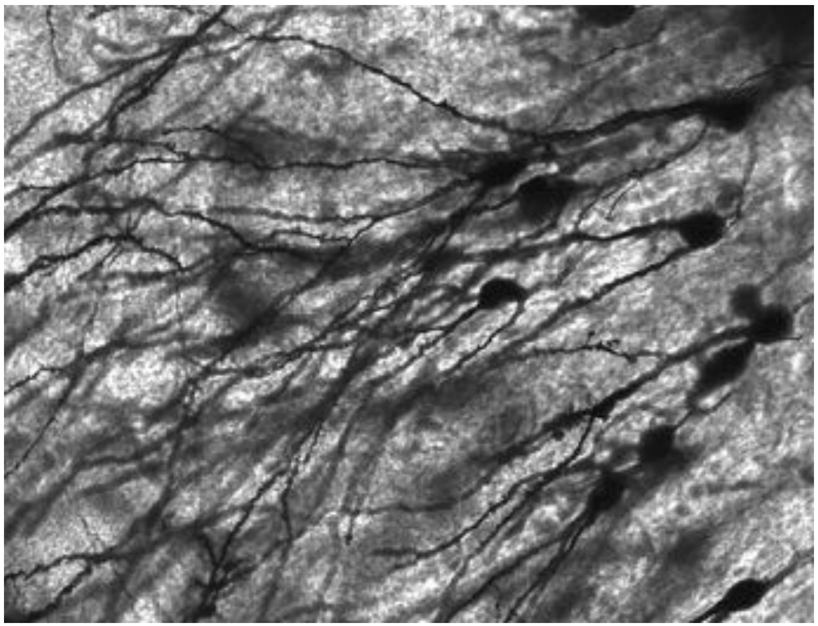
# Structure of the lecture

- Rescorla-Wagner rule
- Backpropagation of error rule
- From backprop to deep networks
- The state of the art? Atari games?

# How do people learn?



# Two perspectives (among many!)



This lecture!

Connectionism

- Neural networks
- Biologically inspired
- Learning from error
- Pattern recognition
- Flexible learning

# Two perspectives (among many!)

Next lecture!

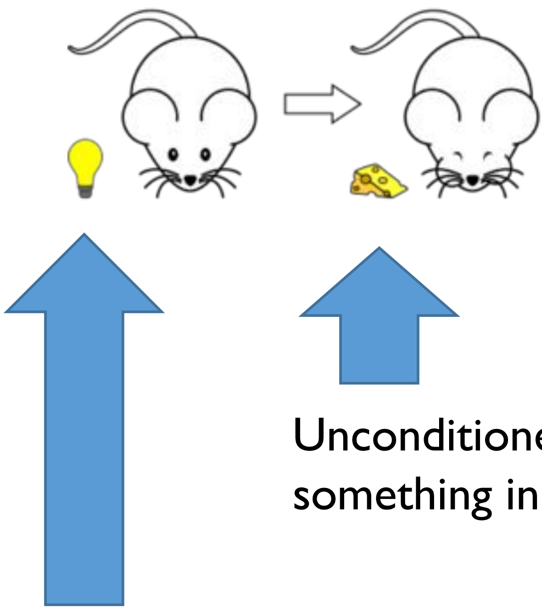
Probabilistic models

- Bayesian models
- Statistical learning
- Structured learning
- Rich generalisation

# The Rescorla-Wagner model: An “error driven” learning theory

# Associative learning

In the simplest design (forward conditioning) the CS is presented slightly before the US, so that it can serve as a signal that reward is coming

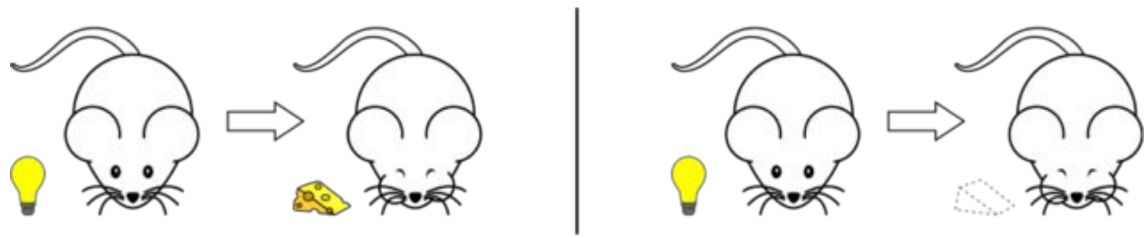


Unconditioned stimulus (US) – something inherently rewarding

Conditioned stimulus (CS) – something initially neutral

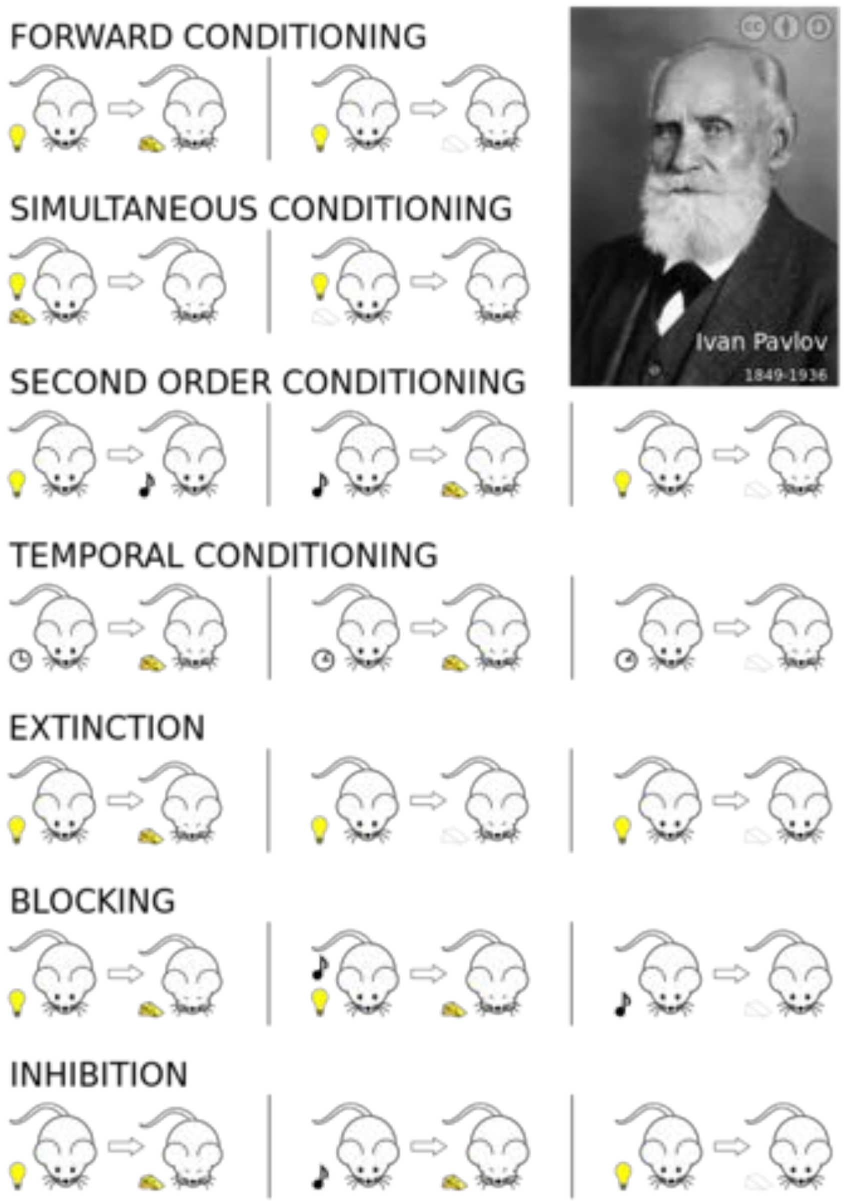
# Associative learning

(well, Pavlovian anyway)



After some number of presentations, the learner starts to respond to the CS in the same way they would respond to the US

They have a learned association between the CS and the US




There are many variations on this!

(Long list of empirical effects to account for)

# Rescorla-Wagner model

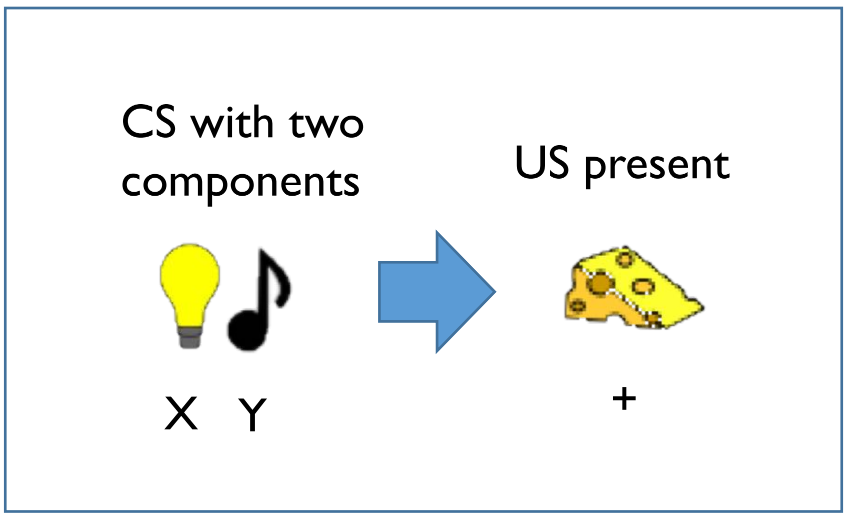
$$V_x \leftarrow V_x + \alpha\beta(\lambda - V_{xy})$$

One popular (though <u>incomplete</u>) account of associative learning is the Rescorla-Wagner model

But what does this strange inscription mean?????

# Rescorla-Wagner model

$$V_x \leftarrow V_x + \alpha\beta(\lambda - V_{xy})$$




An XY+ trial

Consider a design in which there are two features present (X and Y) and the learner needs to predict an outcome that might be present (+) or absent (-)

# Rescorla-Wagner model

$$V_x \leftarrow V_x + \alpha\beta(\lambda - V_{xy})$$

The old strength of association for stimulus X

The new strength of association for stimulus X after seeing XY+

The difference between the old and the new. By convention "differences" are denoted "delta", so we call this "delta-V", ΔV

# Rescorla-Wagner model

$$\Delta V_x = \alpha\beta(\lambda - V_{xy})$$

This delta-V describes "how much we learn about X from the current trial/event"

The "alpha" and "beta" terms here are parameters describing learning rates.

- alpha depends on the conditioned stimulus (X, the thing that is acquiring the association)

- beta depends on the unconditioned stimulus (the thing we're learning to predict)

This isn't an associative learning class so we'll group them together in a single "learning rate", gamma…

# Rescorla-Wagner model

$$\Delta V_x = \gamma(\underline{\lambda - V_{xy}})$$

gamma is a learning rate

This term here is the
"reward prediction error"

# Rescorla-Wagner model

$$\Delta V_x = \gamma(\lambda - V_{xy})$$

lambda is represents the "intrinsic" value of the outcome (unconditioned stimulus), sometimes referred to as the "reward", r

# Rescorla-Wagner model

$$\Delta V_x = \gamma(r - V_{xy})$$

reward

$V_{xy}$ is the "predicted reward": the amount of reward/punishment that the learner expects to receive upon seeing the compound stimulus XY

In the Rescorla-Wagner model, expectations are *additive*, which means that:

$$V_{xy} = V_x + V_y$$

(But not all learning models assume additivity)

# Error driven learning!

$$\Delta V_x = \gamma(r - V_{xy})$$

reward

expected reward

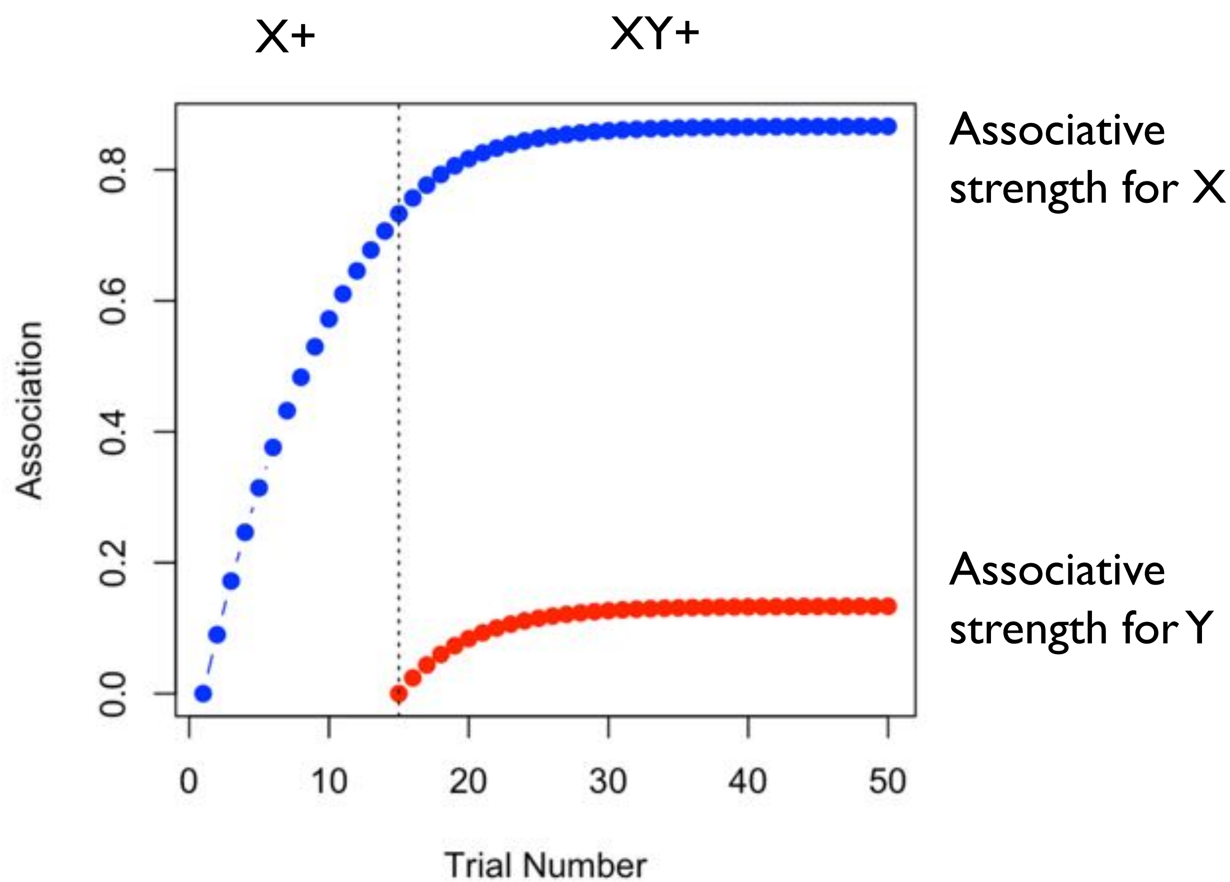
the difference between outcomes and expectations is the *prediction error,* and it is this error that "drives" learning

how much does the learner change their beliefs?

learning is gradual, and depends on a learning rate

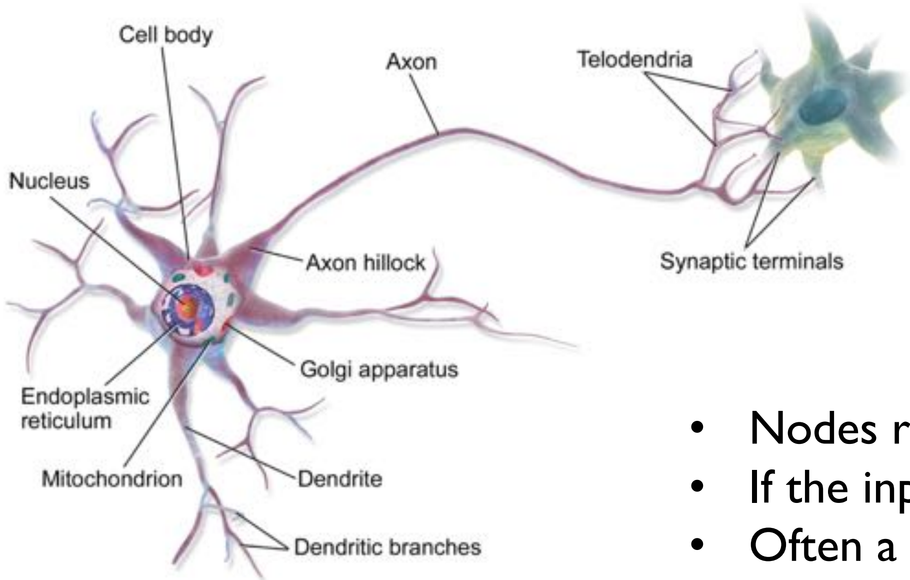
# Example: Blocking

http://compcogscisydney.org/psyr/programming.html



# Recasting the Rescorla-Wagner model as a simple neural network

# What is a neural network?




- Real neurons are complicated things
- "Neural networks" are abstractions

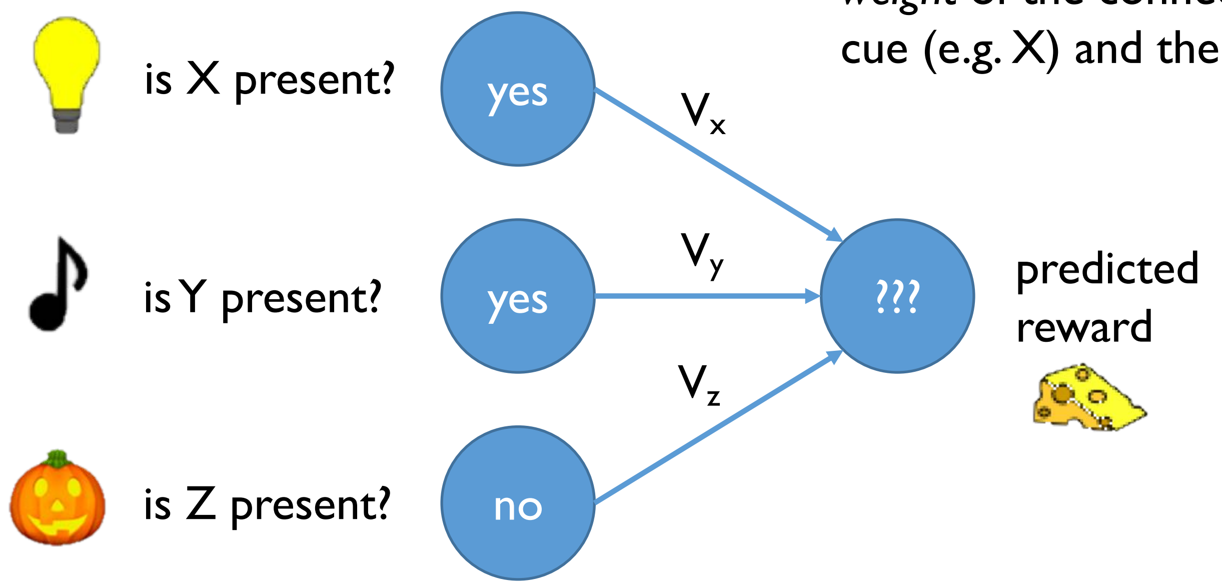
- Nodes receive input from other nodes
- If the input exceeds a threshold, the node "fires"
- Often a simplified "soft" threshold is used… instead of "firing" it just sends a transformed signal to the next node

- Learning occurs by adjusting the "strength" of the connections between nodes

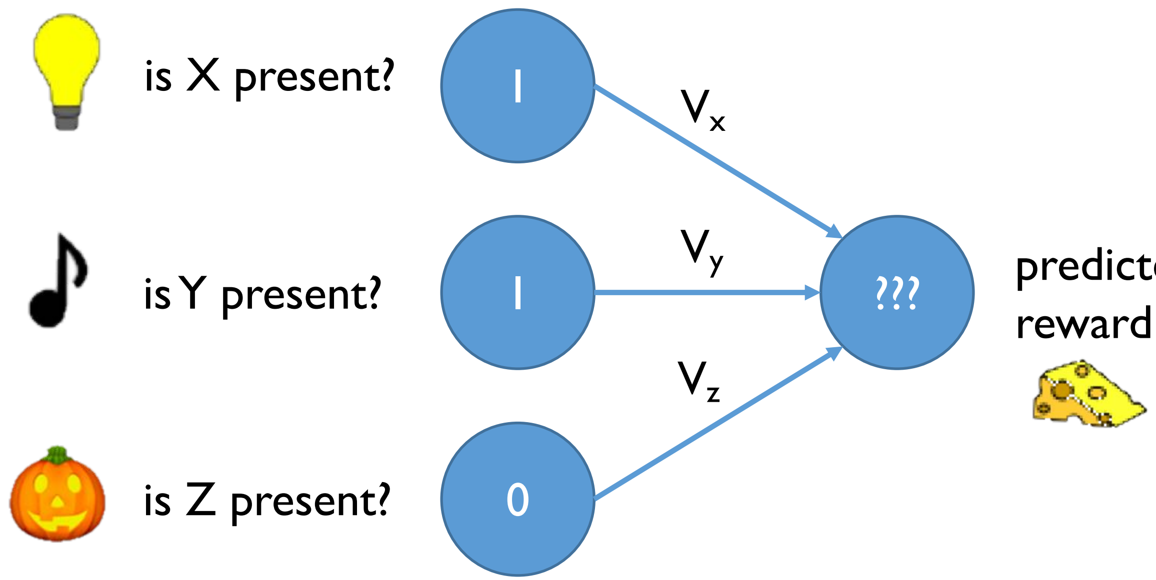
$$\Delta V_x = \gamma(r - V_{xy})$$

# A feed-forward network

the associative strength describes the *weight* of the connection between the cue (e.g. X) and the reward

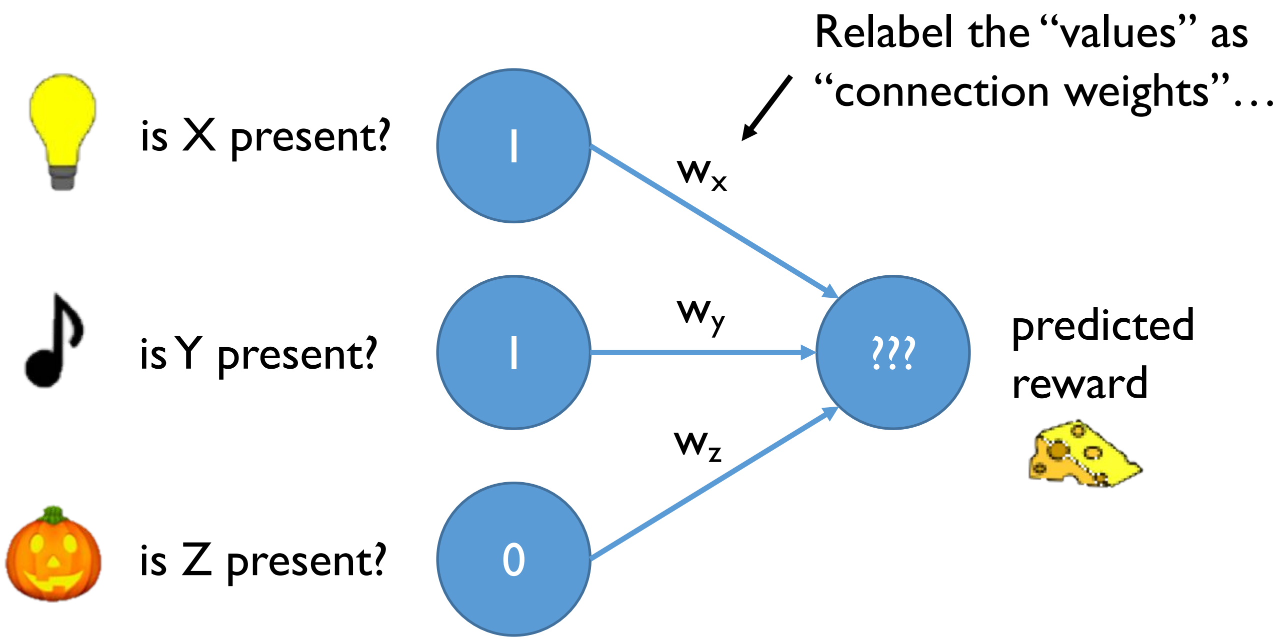


# A feed-forward network




Convert the human-friendly words "yes" and "no" into numerical <u>*activation levels*</u>

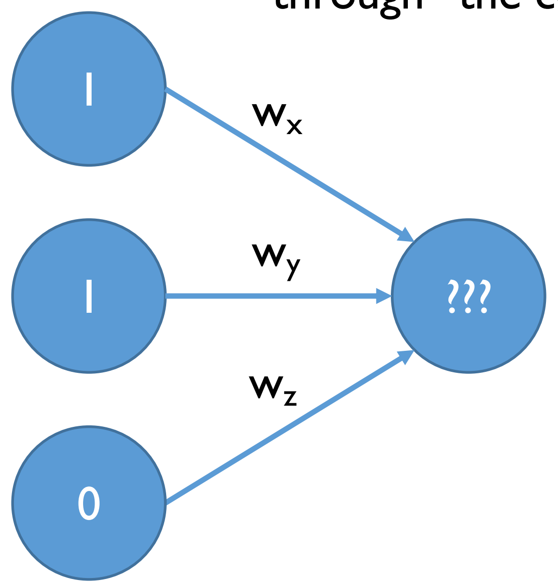
# A feed-forward network



# A feed-forward network

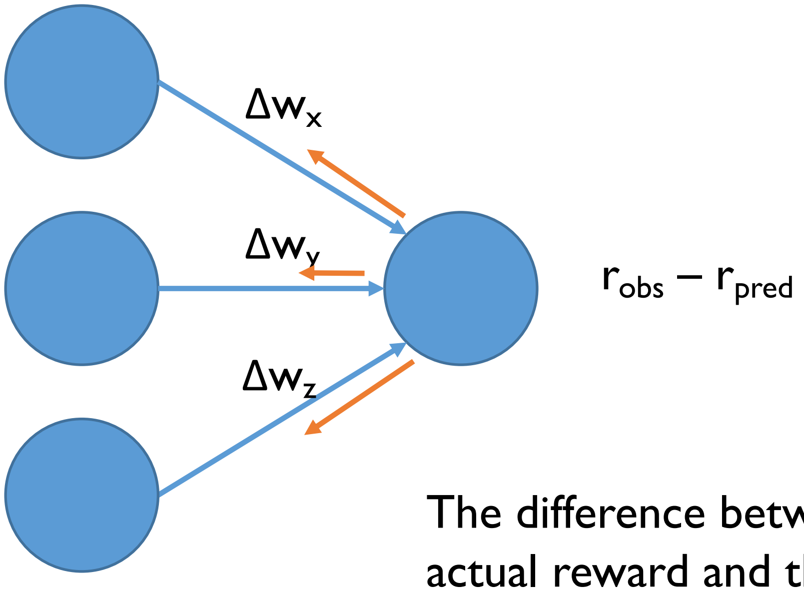
The input is processed by passing it “through” the connection/association weights

Collectively, the activation of these nodes describe an *input pattern*




The predicted reward produced when we pass the input through the connection weights is called the *output*

# Learning by back-propagating error




The difference between the actual reward and the output is used to adjust the connection weights using an error driven learning rule

# Using neural networks to solve categorization problems

# How to classify irises????

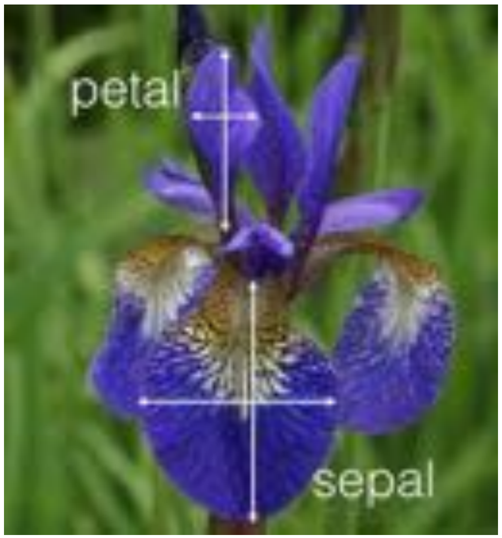
Iris setosa

Iris versicolour

Iris virginica

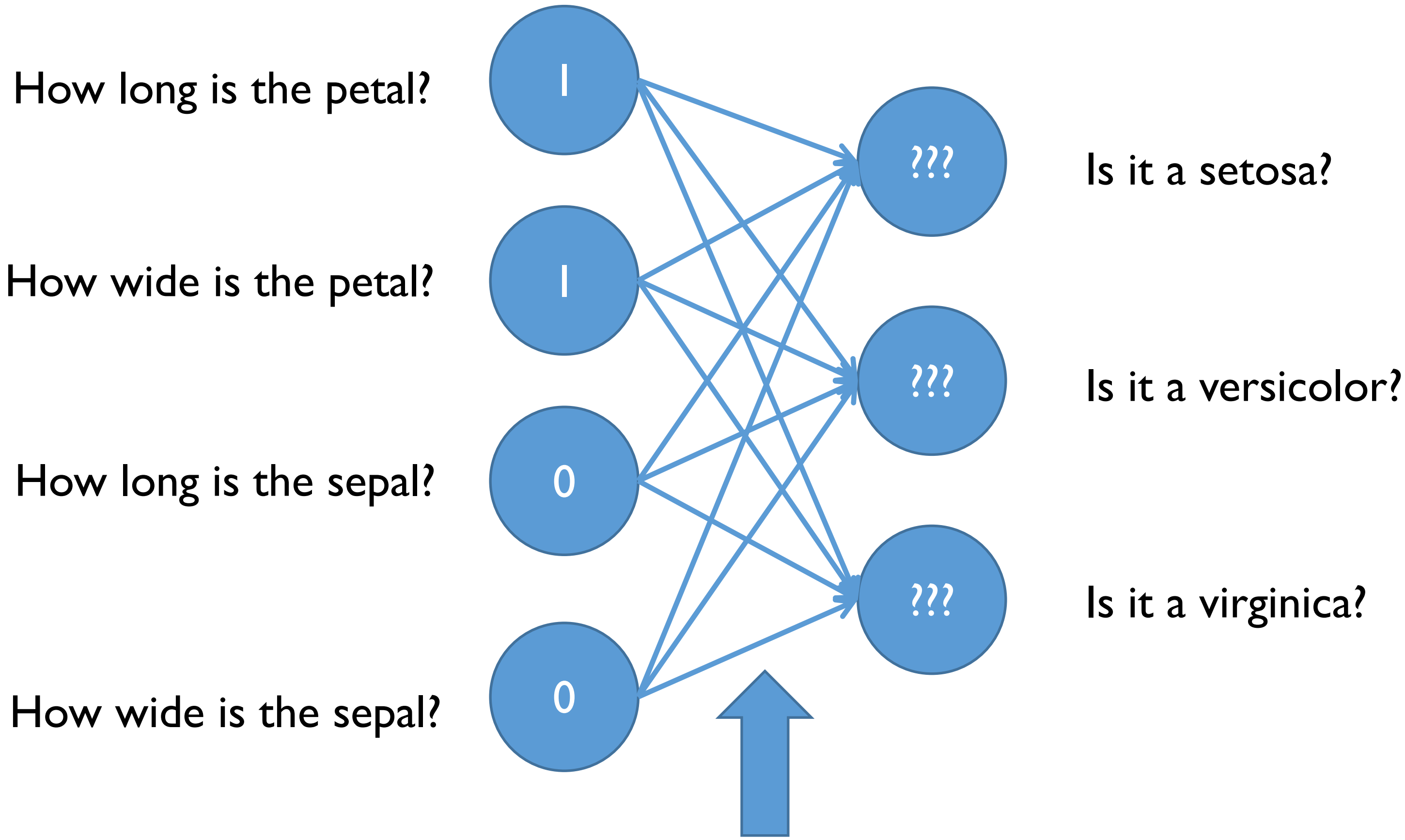
(a classic data set collected by Edgar Anderson, reported by Sir Ronald Fisher 1936)

# This is a "compound" stimulus with many features (at least four)



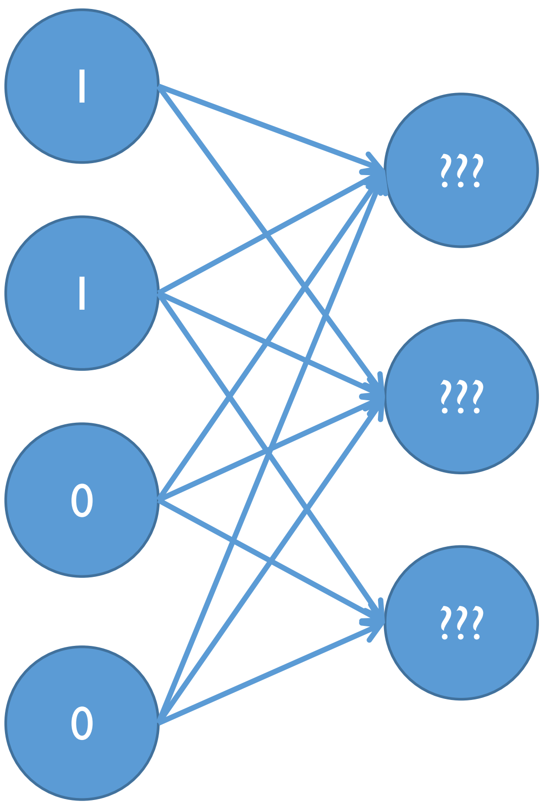

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 5.4 | 3.7 | 1.5 | 0.2 | setosa |
| 5.4 | 3.4 | 1.7 | 0.2 | setosa |
| 6.2 | 2.2 | 4.5 | 1.5 | versicolor |
| 6.7 | 3.0 | 5.0 | 1.7 | versicolor |
| 6.2 | 2.9 | 4.3 | 1.3 | versicolor |
| 6.3 | 3.3 | 6.0 | 2.5 | virginica |
| 5.8 | 2.7 | 5.1 | 1.9 | virginica |
| 7.1 | 3.0 | 5.9 | 2.1 | virginica |
| 7.6 | 3.0 | 6.6 | 2.1 | virginica |
| 6.1 | 2.6 | 5.6 | 1.4 | virginica |

# Our first connectionist network




Each of these links represents an association strength that we can acquire through repeated experience

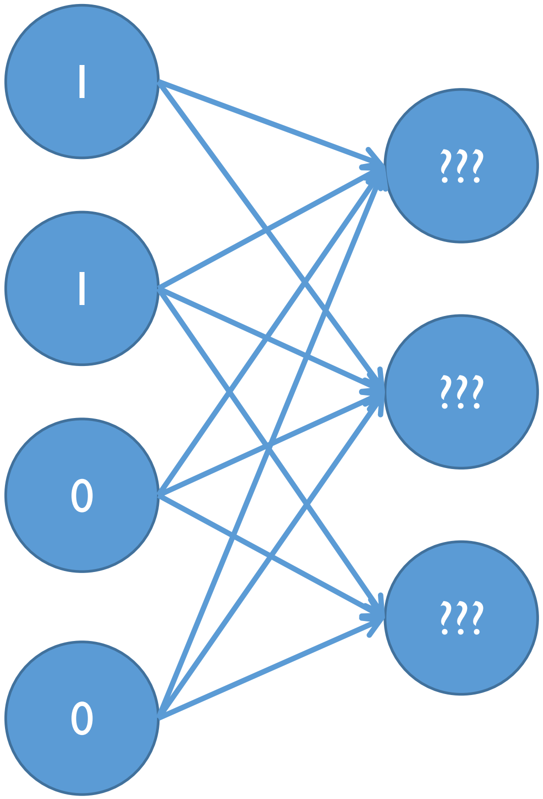
# Our first connectionist network

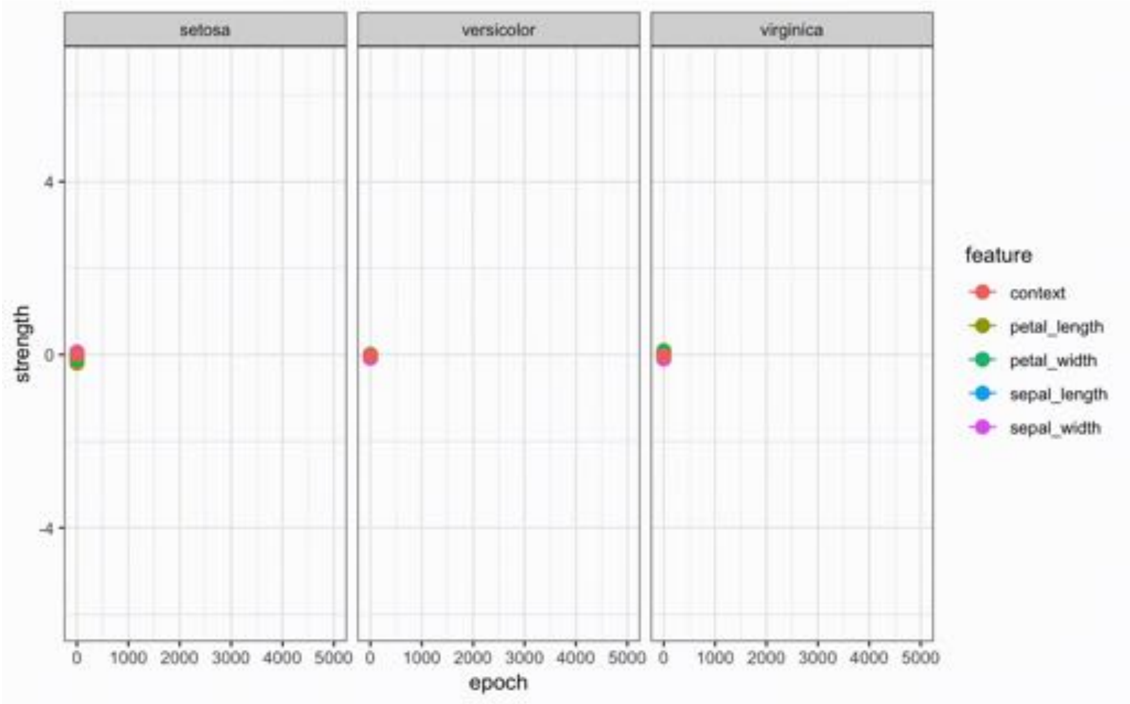



Some technical detail…

- Nonlinear activation function
- Bias parameters for contextual learning
- Delta rule subtly different from R/W

# Learning connection weights



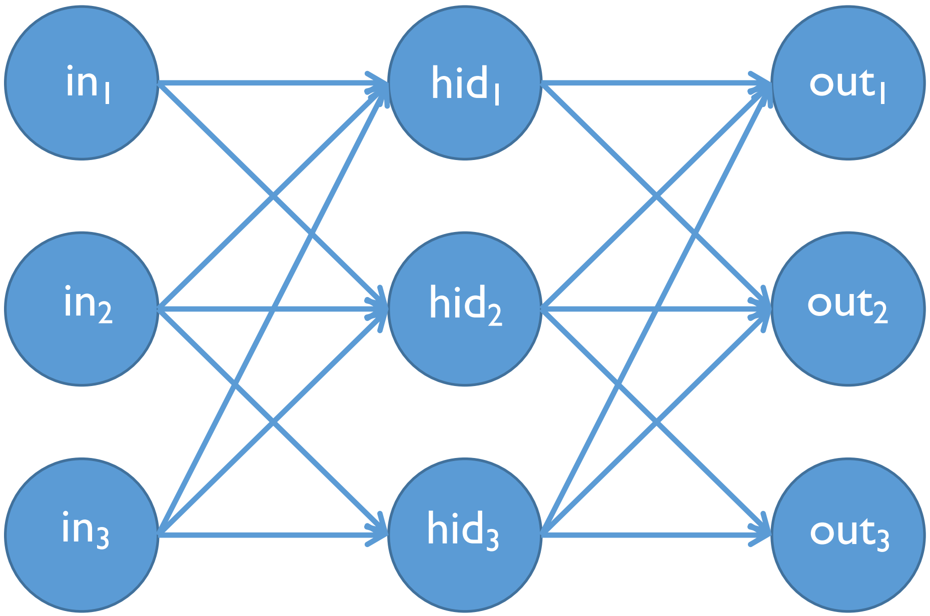




http://compcogscisydney.org/psyr/backprop.html

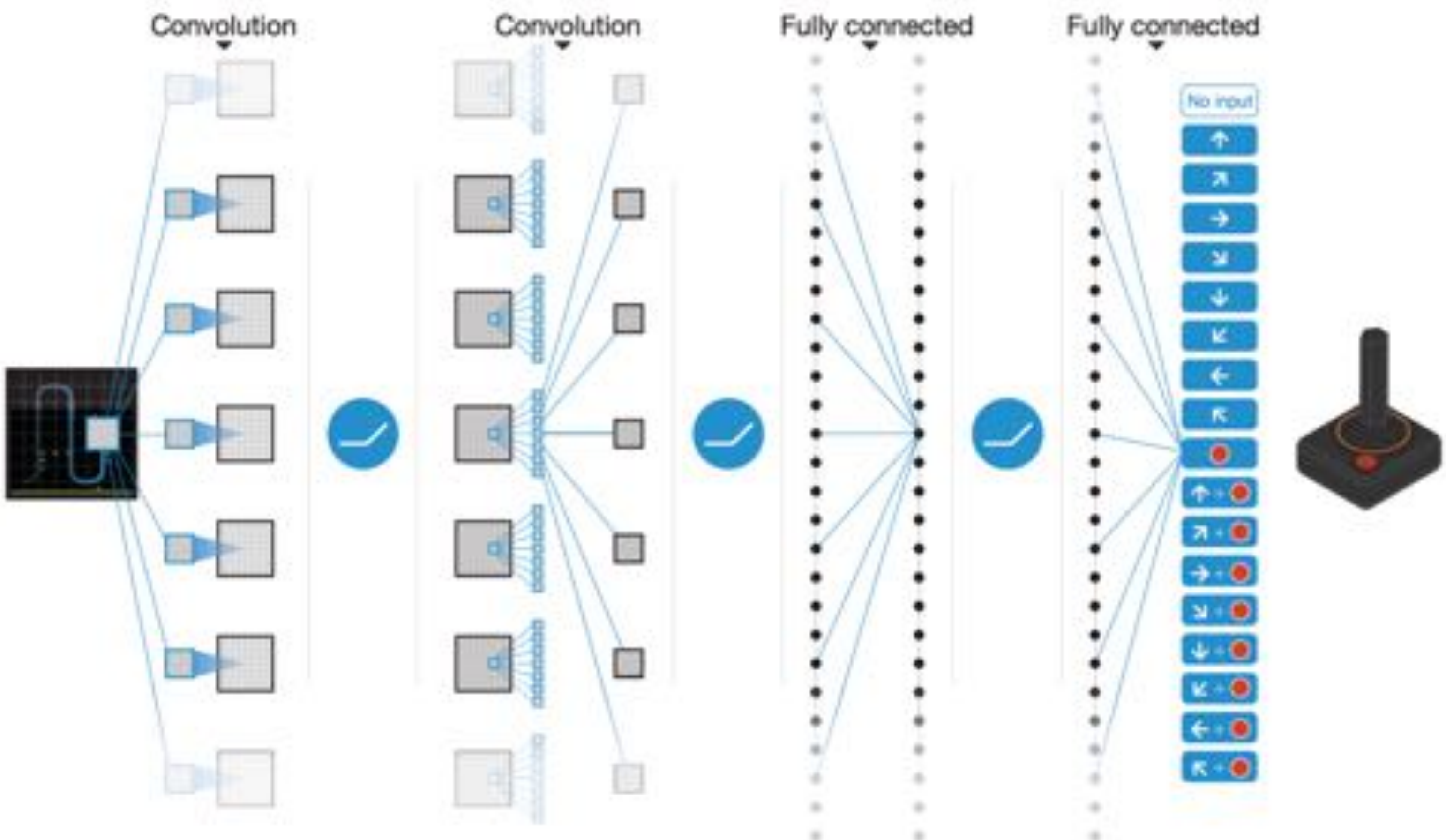
# Building richer networks

(and understanding what they are good at)

# Some networks have hidden layers



# “Deep” networks can have many layers




(there are lots of technical details… more to it than just “add more layers”)

# Google DeepMind

# Human level reinforcement learning?


Better than human-level control of classic Atari games
through Deep Reinforcement Learning.

# Human level reinforcement learning?

Mnih et al (2015)

LETTER

doi:10.1038/nature14236

## Human-level control through deep reinforcement learning

Volodymyr Mnih[1*], Koray Kavukcuoglu[1*], David Silver[1*], Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

**The theory of reinforcement learning provides a normative account[1], deeply rooted in psychological[2] and neuroscientific[3] perspectives on animal behaviour, of how agents may optimize their control of an environment. To use reinforcement learning successfully in situations approaching real-world complexity, however, agents are confronted with a difficult task: they must derive efficient representations of the environment from high-dimensional sensory inputs, and use these to generalize past experience to new situations. Remarkably, humans and other animals seem to solve this problem through a harmonious combination of reinforcement learning and hierarchical sensory processing systems[4,5], the former evidenced by a wealth of neural data revealing notable parallels between the phasic signals emitted by dopaminergic neurons and temporal difference reinforcement learning algorithms[3]. While reinforcement learning agents have achieved some successes in a variety of domains[6–8], their applicability has previously been limited to domains in which useful features can be handcrafted, or to domains with fully observed, low-dimensional state spaces. Here we use recent advances in training deep neural networks[9–11] to develop a novel artificial agent, termed a deep Q-network, that can learn successful policies directly from high-dimensional sensory inputs using end-to-end reinforcement learning. We tested this agent on the challenging domain of classic Atari 2600 games[12]. We demonstrate that the deep Q-network agent, receiving only the pixels and the game score as inputs, was able to surpass the performance of all previous algorithms and achieve a level comparable to that of a professional human games tester across a set of 49 games, using the same algorithm, network architecture and hyperparameters. This work bridges the divide between high-dimensional sensory inputs and actions, resulting in the first artificial agent that is capable of learning to excel at a diverse array of challenging tasks.**

agent is to select actions in a fashion that maximizes cumulative future reward. More formally, we use a deep convolutional neural network to approximate the optimal action-value function

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \mid s_t = s, a_t = a, \pi\right],$$

which is the maximum sum of rewards $r_t$ discounted by $\gamma$ at each time-step $t$, achievable by a behaviour policy $\pi = P(a|s)$, after making an observation ($s$) and taking an action ($a$) (see Methods)[19].

Reinforcement learning is known to be unstable or even to diverge when a nonlinear function approximator such as a neural network is used to represent the action-value (also known as $Q$) function[20]. This instability has several causes: the correlations present in the sequence of observations, the fact that small updates to $Q$ may significantly change the policy and therefore change the data distribution, and the correlations between the action-values ($Q$) and the target values $r + \gamma \max_{a'} Q(s', a')$. We address these instabilities with a novel variant of Q-learning, which uses two key ideas. First, we used a biologically inspired mechanism termed experience replay[21–23] that randomizes over the data, thereby removing correlations in the observation sequence and smoothing over changes in the data distribution (see below for details). Second, we used an iterative update that adjusts the action-values ($Q$) towards target values that are only periodically updated, thereby reducing correlations with the target.

While other stable methods exist for training neural networks in the reinforcement learning setting, such as neural fitted Q-iteration[24], these methods involve the repeated training of networks *de novo* on hundreds of iterations. Consequently, these methods, unlike our algorithm, are too inefficient to be used successfully with large neural networks. We parameterize an approximate value function $Q(s,a;\theta_i)$ using the deep convolutional neural network shown in Fig. 1, in which $\theta_i$ are the param-


Better than human-level control of classic Atari games through Deep Reinforcement Learning.

https://storage.googleapis.com/deepmind-media/dqn/DQNNaturePaper.pdf

# Human level reinforcement learning?

Mnih et al (2015)

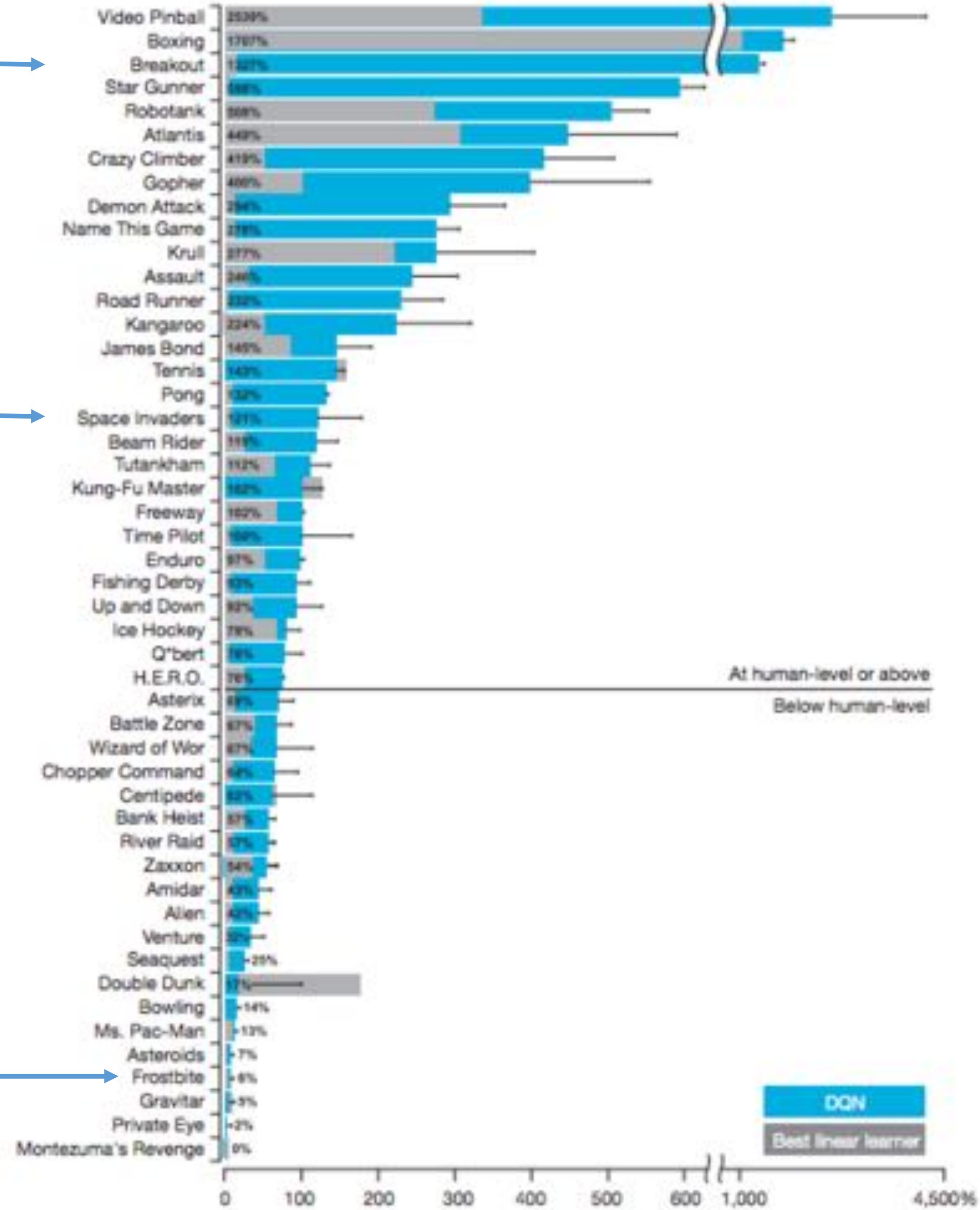
The theory of reinforcement learning provides a normative account, deeply rooted in psychological and neuroscientific perspectives on animal behaviour, of how agents may optimize their control of an environment. To use reinforcement learning successfully in situations approaching real-world complexity, however, agents are confronted with a difficult task: they must derive efficient representations of the environment from high-dimensional sensory inputs, and use these to generalize past experience to new situations


Better than human-level control of classic Atari games through Deep Reinforcement Learning.

Don’t try to defeat an AI at breakout

Don’t try to defeat an AI at space invaders

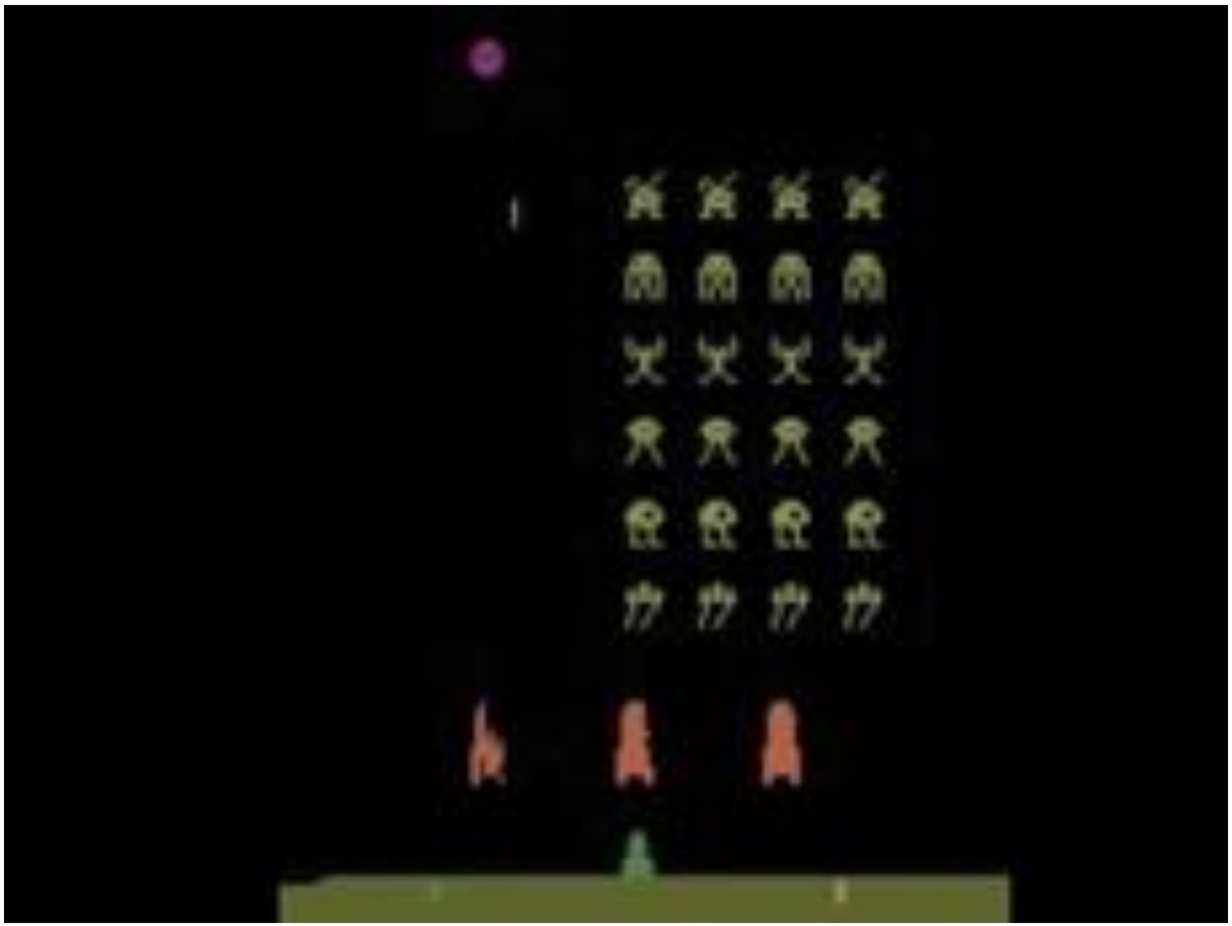
Do challenge an AI to play Frostbite!



# DQN plays Breakout

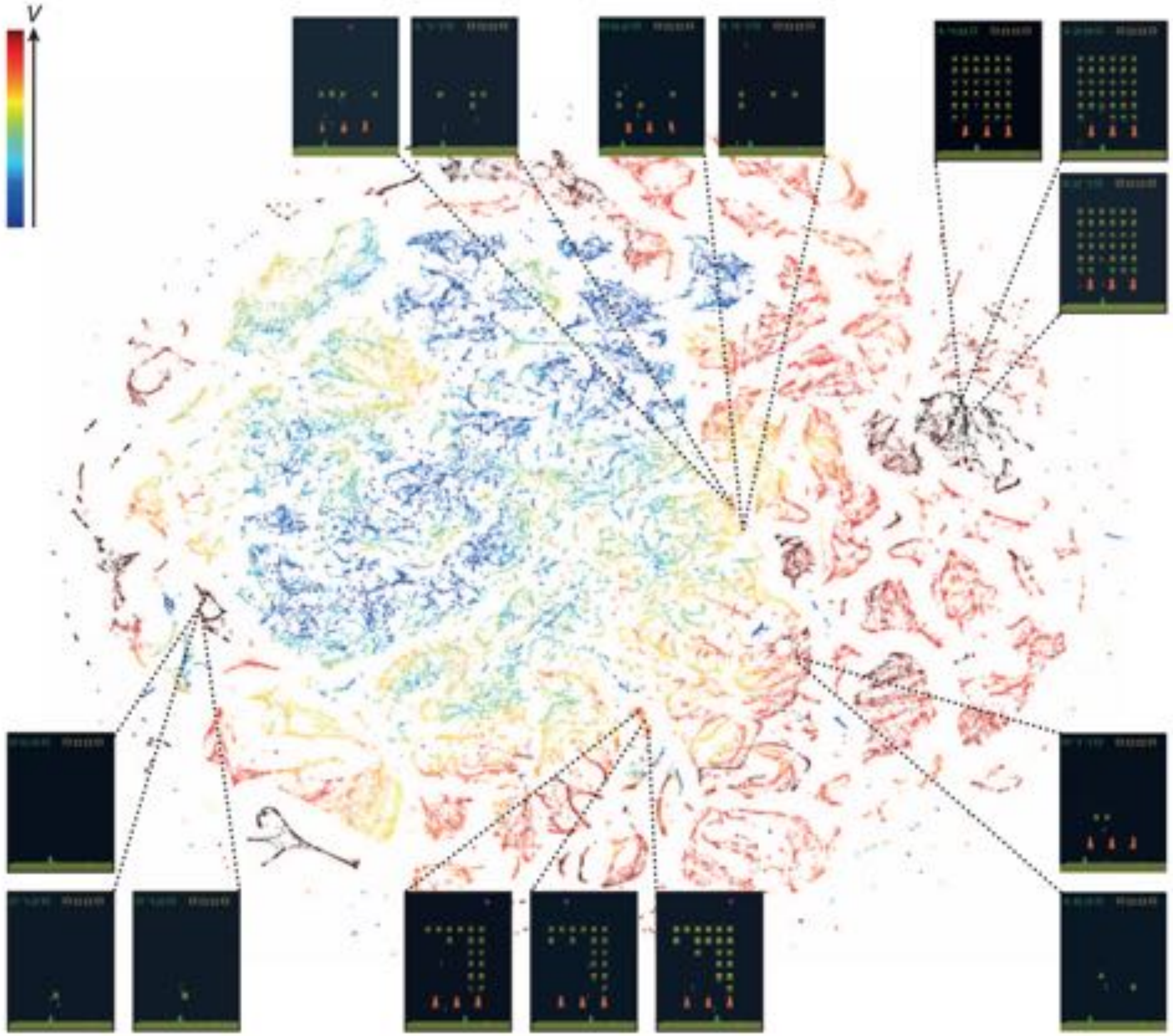

https://youtu.be/TmPfTpjtdgg

# DQN plays Space Invaders



https://youtu.be/W2CAghUiofY

# What does it learn?



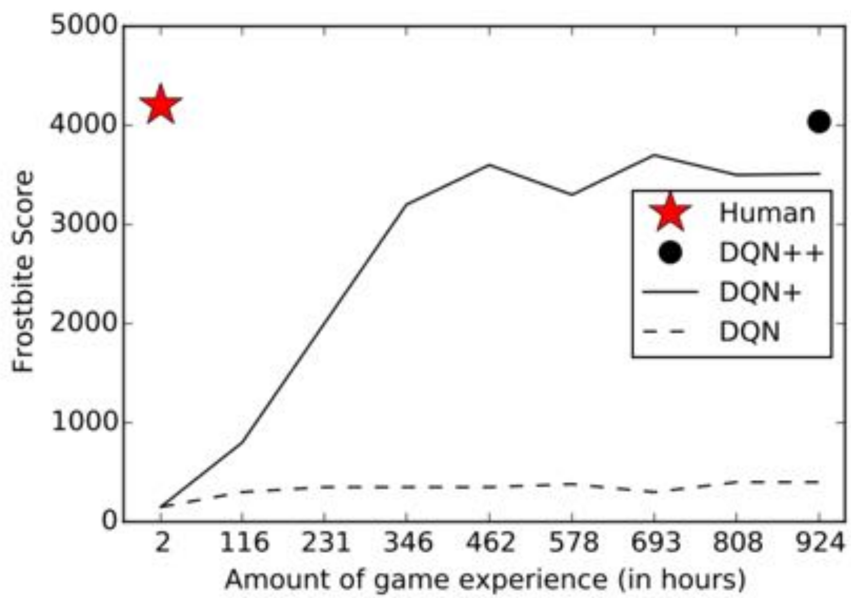
# What doesn't it learn?

(this is a human playing Frostbite)


https://youtu.be/1d4D5KhpKa8

# Why is Frostbite hard?

- It's a structured game with sparse rewards
- Needs "forward planning" to play well
- Not really a "pattern recognition" problem




http://web.stanford.edu/class/psych209/Readings/LakeEtAlBBS.pdf

# When pattern recognition goes awry

Very cool blog post by Janelle Shane

http://aiweirdness.com/post/171451900302/do-neural-nets-dream-of-electric-sheep

A herd of sheep grazing on a lush green hillside
Tags: grazing, sheep, mountain, cattle, horse

A close up of a lush green field
Tags: grass, field, sheep, standing, rainbow, man

These are scenes that "usually" have sheep so the network (this one is Microsoft, not Google) tags it with sheep

# When pattern recognition goes awry

Left: A man is holding a dog in his hand
Right: A woman is holding a dog in her hand
Image: @SouperSarah

NeuralTalk2: A flock of birds flying in the air
Microsoft Azure: A group of giraffe standing next to a tree
Image: Fred Dunn, https://www.flickr.com/photos/gratopictures - CC-BY-NC

A goat being held by a child is labelled a "dog"

Goats in trees become birds or giraffes

# When pattern recognition goes awry


Paint a sheep orange and
it becomes a flower

Thanks!