Reinforcement Learning

Deep Reinforcement Learning
(Material not examinable)

Subramanian Ramamoorthy
School of Informatics

4 April, 2017

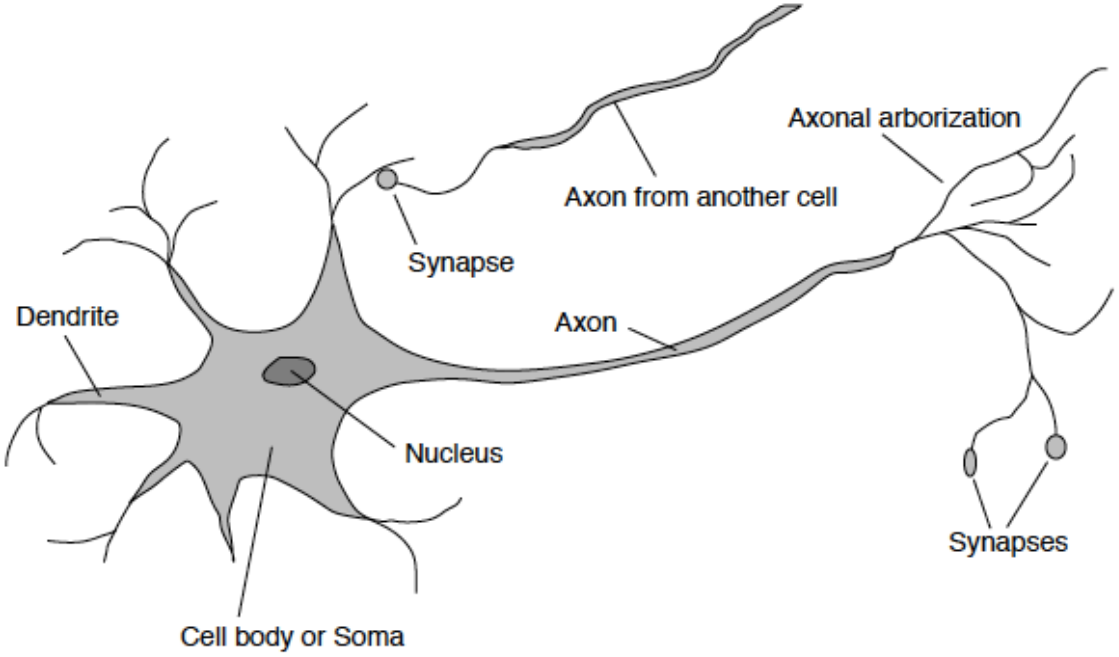
Plan for Lecture

- Background: What is a neural network? What is "deep"?
- Experiments in the Arcade Learning Environment – DeepMind's architecture at a high level
- Value Iteration Networks, using original slides from Tamar et al., from NIPS 2016

Background

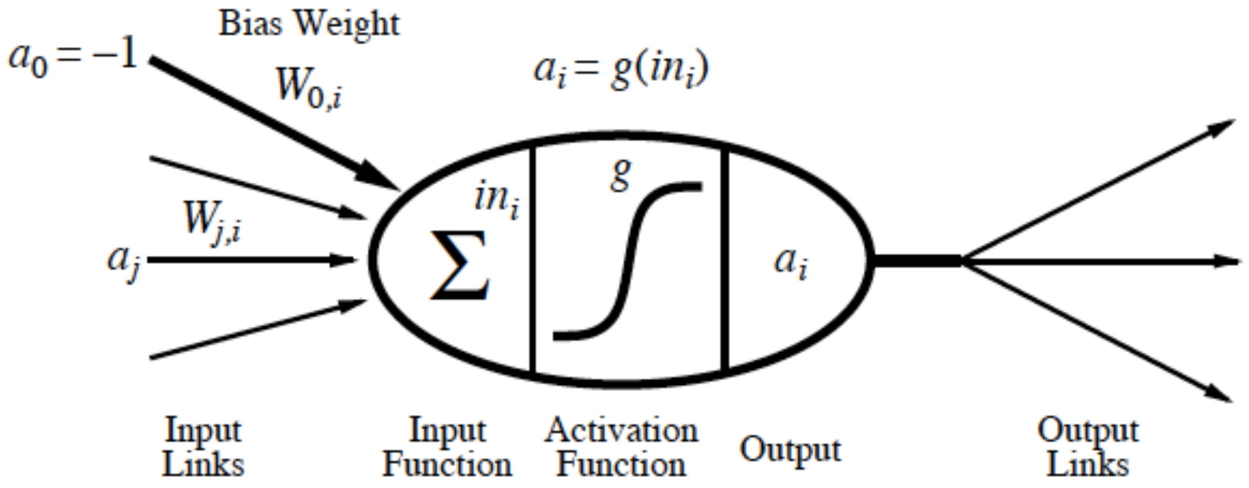
Biological Neural Networks

10^{11} neurons of > 20 types, 10^{14} synapses, 1ms–10ms cycle time
Signals are noisy "spike trains" of electrical potential



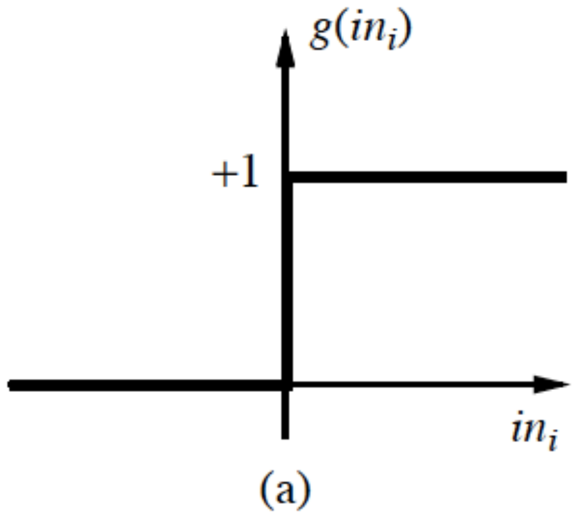
A McCulloch - Pitts "Unit"

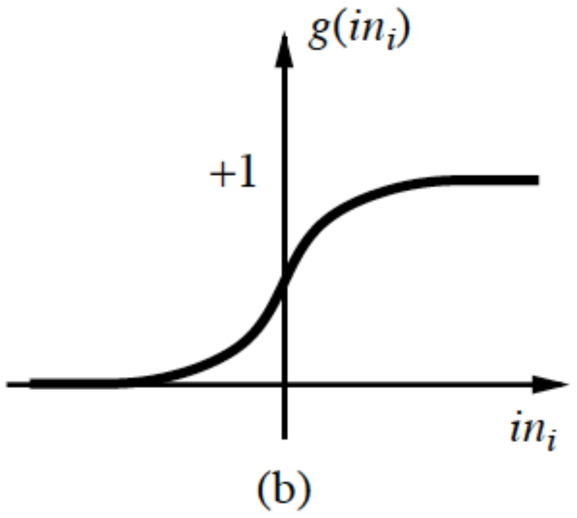
$$a_i \leftarrow g(in_i) = g\left(\Sigma_j W_{j,i} a_j\right)$$



A gross over-simplication of real neurons, but its purpose is to develop understanding of what networks of simple units can do

Activation Functions



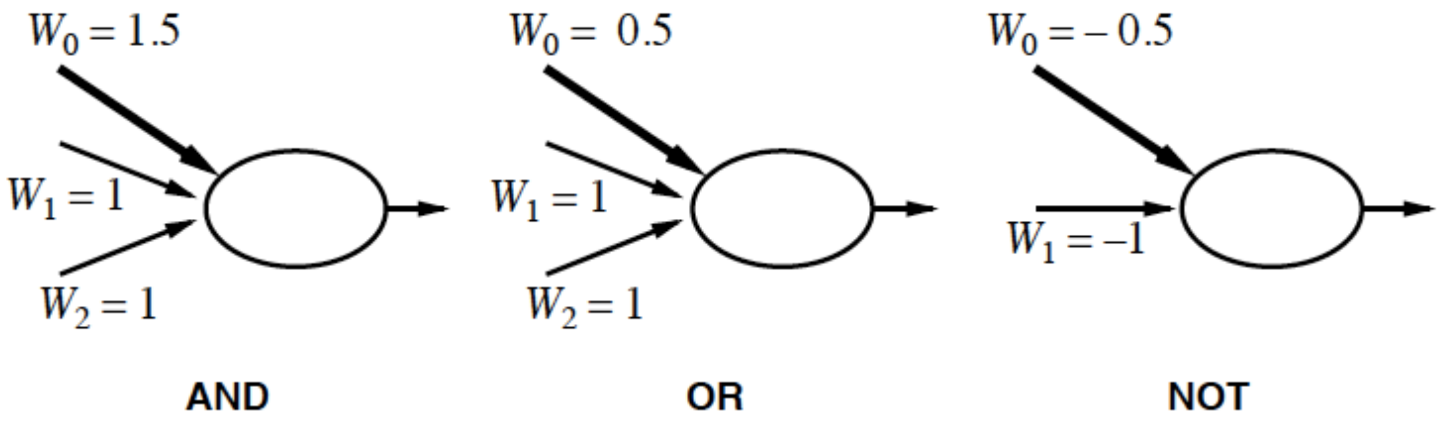


(a) is a step function or threshold function

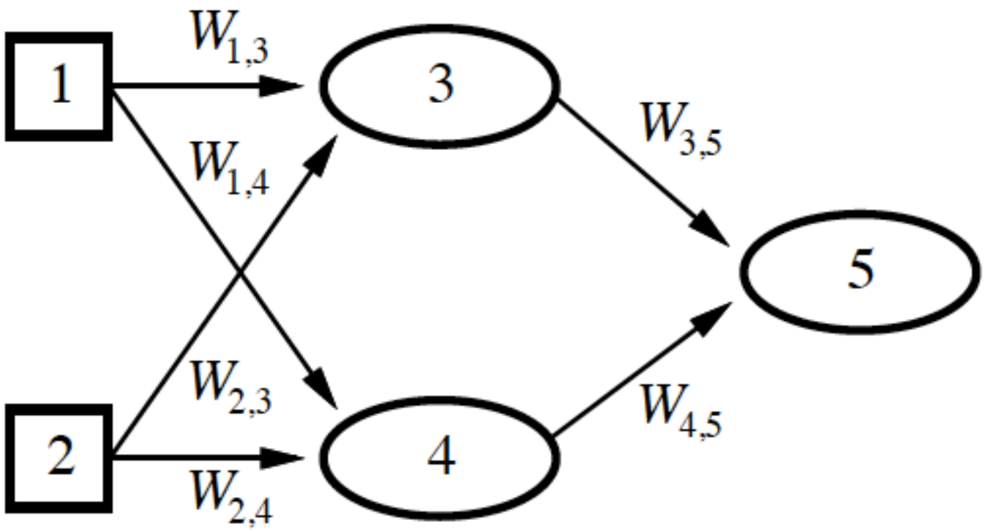
(b) is a sigmoid function $1/(1+e^{-x})$

Changing the bias weight $W_{0,i}$ moves the threshold location

“Every Logical Function can be Implemented”



Feed-forward Network

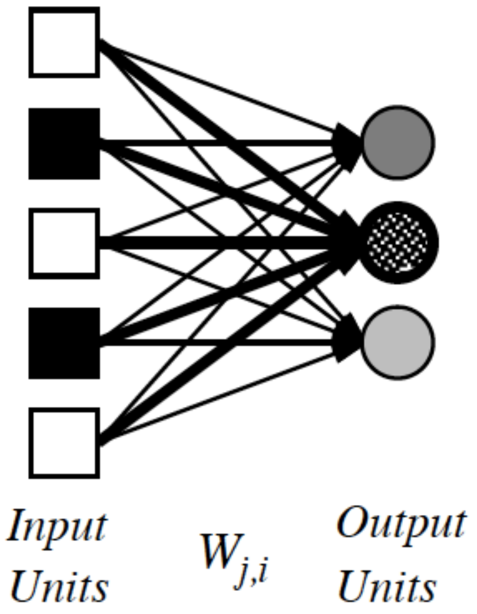


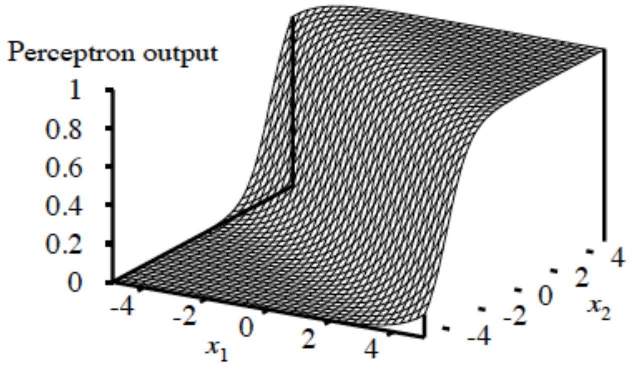
Feed-forward network = a parameterized family of nonlinear functions:

$$\begin{aligned} a_5 &= g(W_{3,5} \cdot a_3 + W_{4,5} \cdot a_4) \\ &= g(W_{3,5} \cdot g(W_{1,3} \cdot a_1 + W_{2,3} \cdot a_2) + W_{4,5} \cdot g(W_{1,4} \cdot a_1 + W_{2,4} \cdot a_2)) \end{aligned}$$

Adjusting weights changes the function: do learning this way!

Single-layer Perceptron



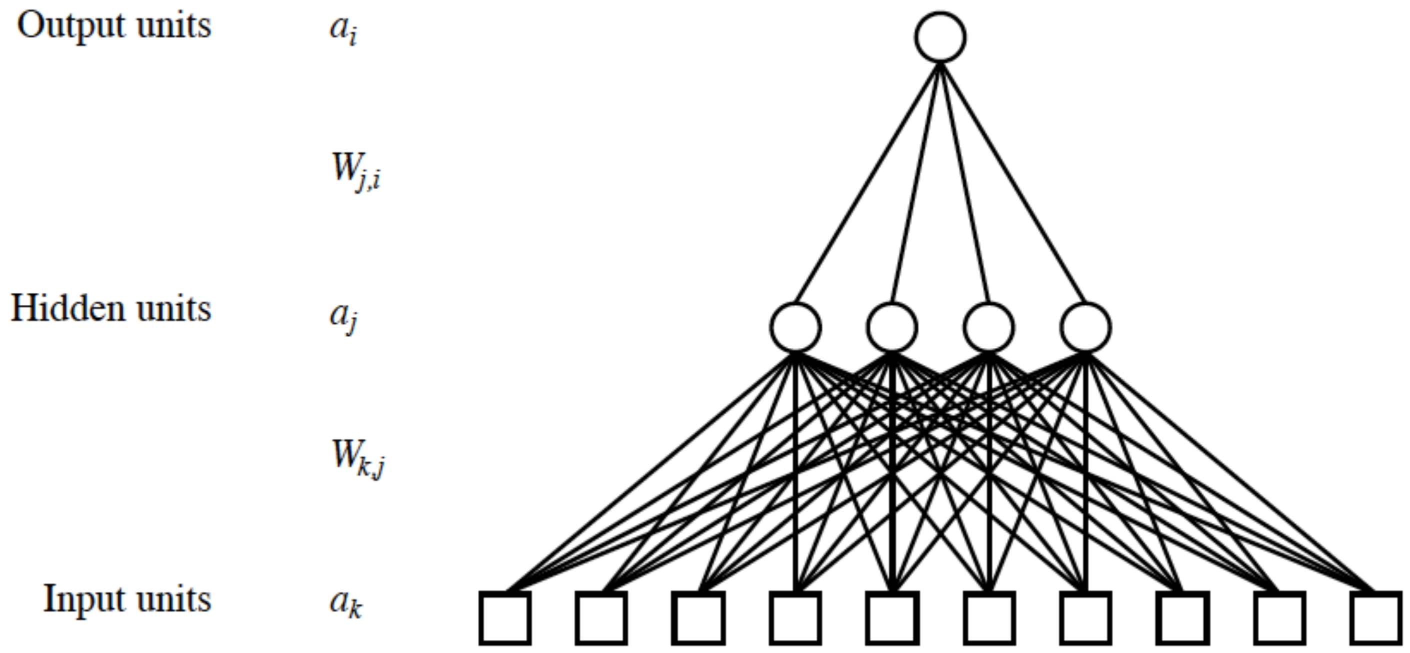


Output units all operate separately—no shared weights

Adjusting weights moves the location, orientation, and steepness of cliff

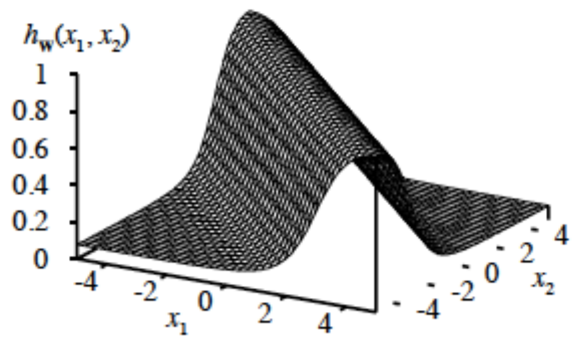
Multi-layer Perceptrons

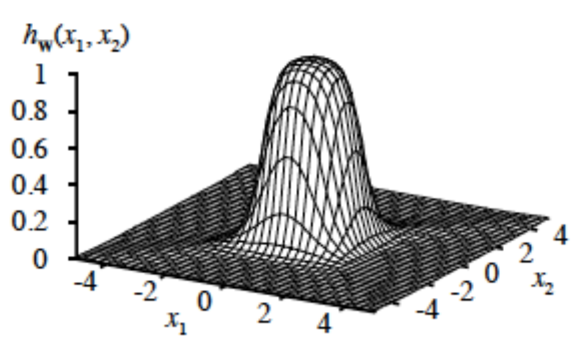
Layers are usually fully connected;
numbers of hidden units typically chosen by hand



Expressiveness of MLPs

All continuous functions can be approximated with 2 layers, and all functions with 3 layers (Universal approximation theorems, e.g., Cybenko 1989)



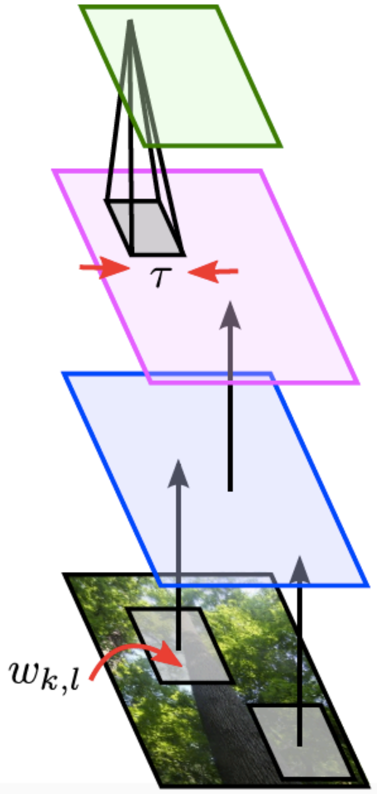


Towards Deep Architectures: Biological Vision is Hierarchical

$$\begin{tikzcd}
\text{object} & \text{trees} & \text{Inferotemporal cortex} \\
\text{object parts} \arrow[u] & \text{bark, leaves, etc.} \arrow[u] & \text{V4: different textures} \\
\text{primitive features} \arrow[u] & \text{oriented edges} \arrow[u] & \text{V1: simple and complex cells} \\
\text{input image} \arrow[u] & \text{forest image} \arrow[u] & \text{photo-receptors retina}
\end{tikzcd}$$

[Source: Richard Turner, U. Cambridge]

Building Block of a "Convolutional" Neural Network



$$x_{i,j} = \max_{|k|<\tau,|l|<\tau} y_{i-k,j-l}$$

mean or subsample also used

pooling stage

$$y_{i,j} = f(a_{i,j})$$

e.g. $f(a) = [a]_+$

$f(a) = \text{sigmoid}(a)$

non-linear stage

$$a_{i,j} = \sum_{k,l} w_{k,l} z_{i-k,j-l}$$

only parameters

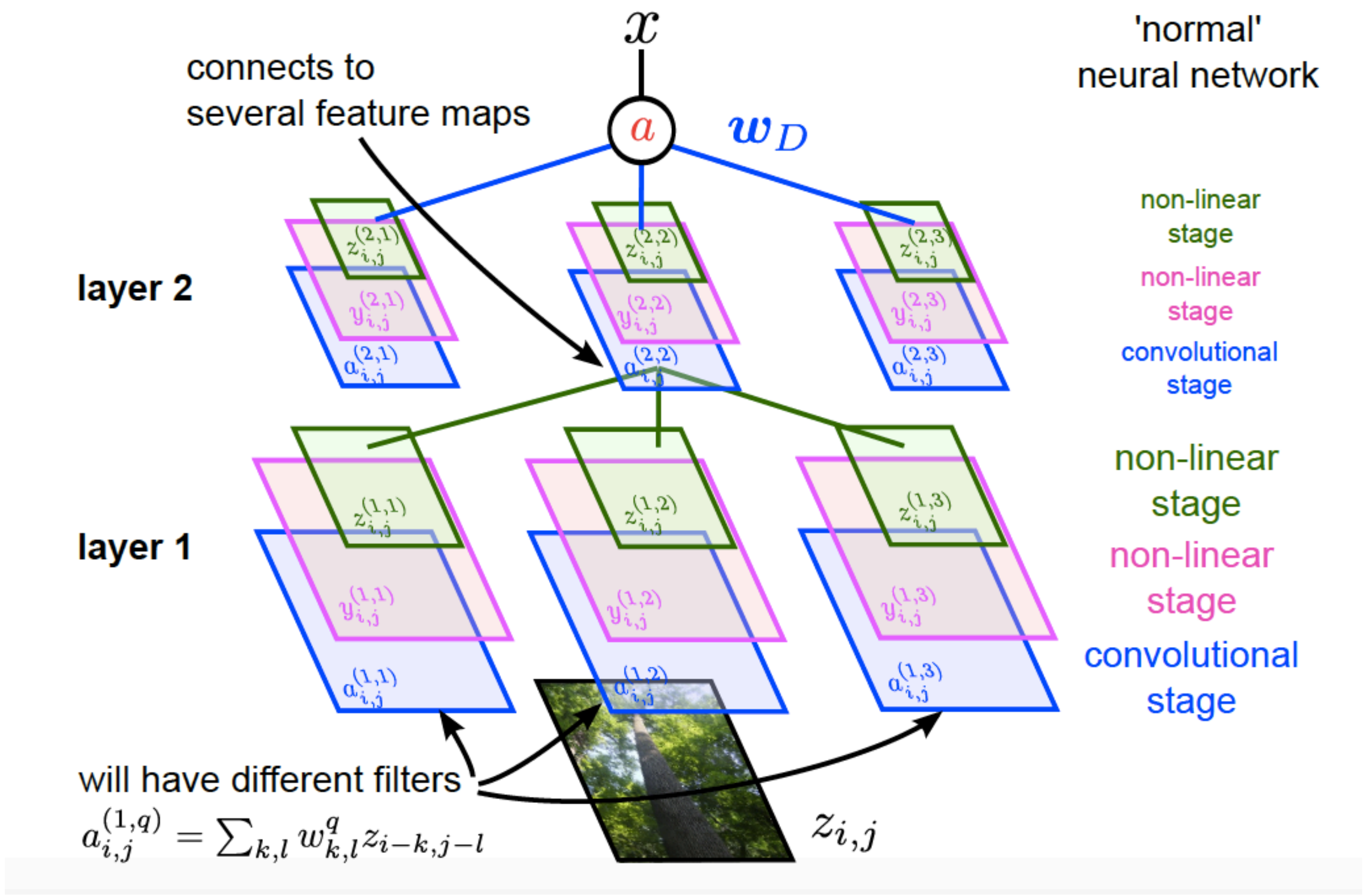
convolutional stage

$z_{i,j}$

input image

[Source: Richard Turner, U. Cambridge]

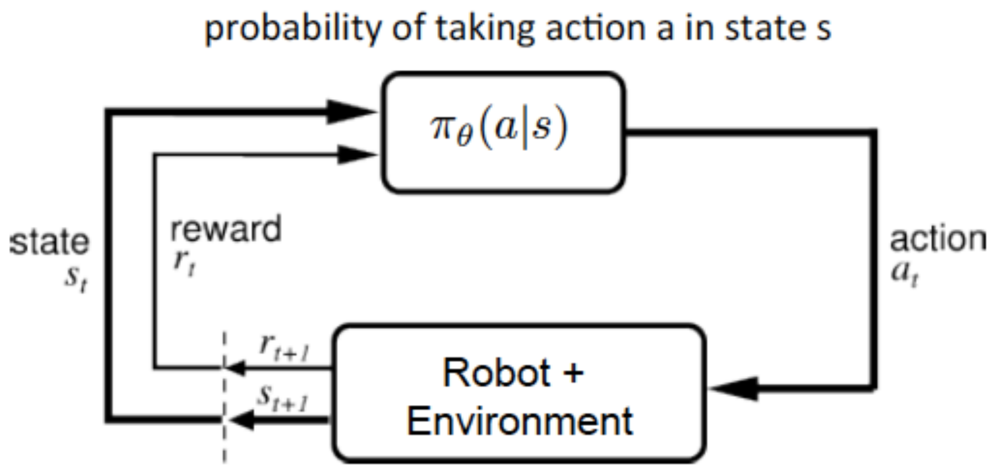
Fully Convolutional Neural Network



[Source: Richard Turner, U. Cambridge]

V. Mnih et al., Human-level control through deep reinforcement learning, *Nature* **518**:529, 2015.

Core Problem in RL; Policy Gradient Formulation

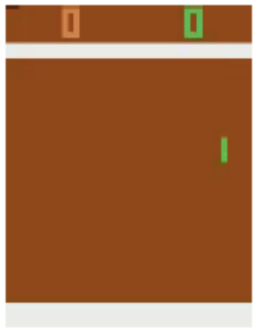


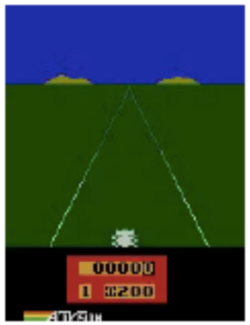
- Goal:

$$\max_\theta \quad \mathrm{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$$

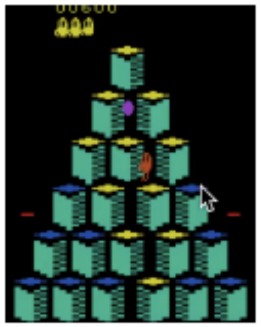
Further challenges:
- Stability
- Credit assignment
- Exploration

Can we go from Pixels to Actions?


Pong


Enduro

Beamrider


Q*bert

Deep Q-learning is still Q-learning

- Function approximation of the$\mathcal{Q}$-function

$$\mathcal{Q}(s; a; w) \approx \mathcal{Q}(s; a)$$

- $\delta = r + \gamma \max_{a'} \mathcal{Q}\left(s', a'; w\right) - \mathcal{Q}\left(s, a; w\right)$
- Update by stochastic gradient descent

$$\begin{aligned} \Delta w &= -\eta \frac{\partial \delta^2}{\partial w} \\ &= \eta \left(r + \gamma \max_{a'} \mathcal{Q}\left(s', a'; w\right) - \mathcal{Q}\left(s, a; w\right) \right) \frac{\partial \mathcal{Q}\left(s, a; w\right)}{\partial w} \end{aligned}$$

Deep Q-learning: Lessons from RL with Function Approximation

Countermeasures against divergence of $Q(s, a; w)$

[David Silver, DeepMind]

- Diversify data to reduce episodic correlation
- Use lots of data, i.e. many quadruples (s, a, r, s')
- Use one network for $Q(s, a; w_0)$ and another network for $Q(s', a'; w_1)$
- Use information about the reward distribution
- Normalisation to get robust gradients

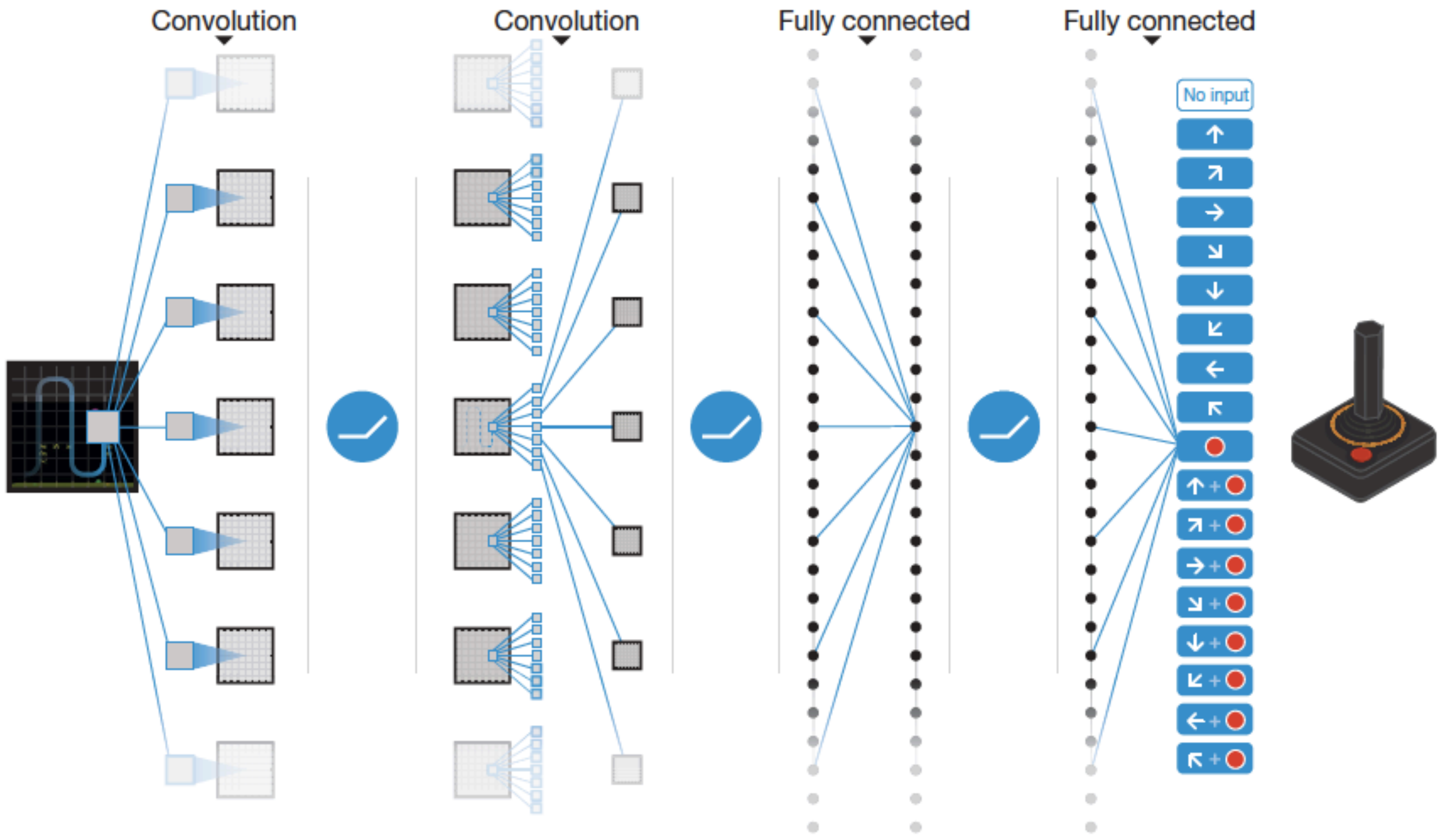
Deep Q-Network

- Represent artificial agent by deep neural network: DQN
- Learn successful policies directly from high-dimensional sensory inputs
- End-to-end reinforcement learning
- Experiments involved testing classic Atari 2600 games
- Reward: Game score
- Achieved a level comparable to a professional human games tester across a set of 49 games (about half of the games better than human)
- Unchanged meta-parameters for all games

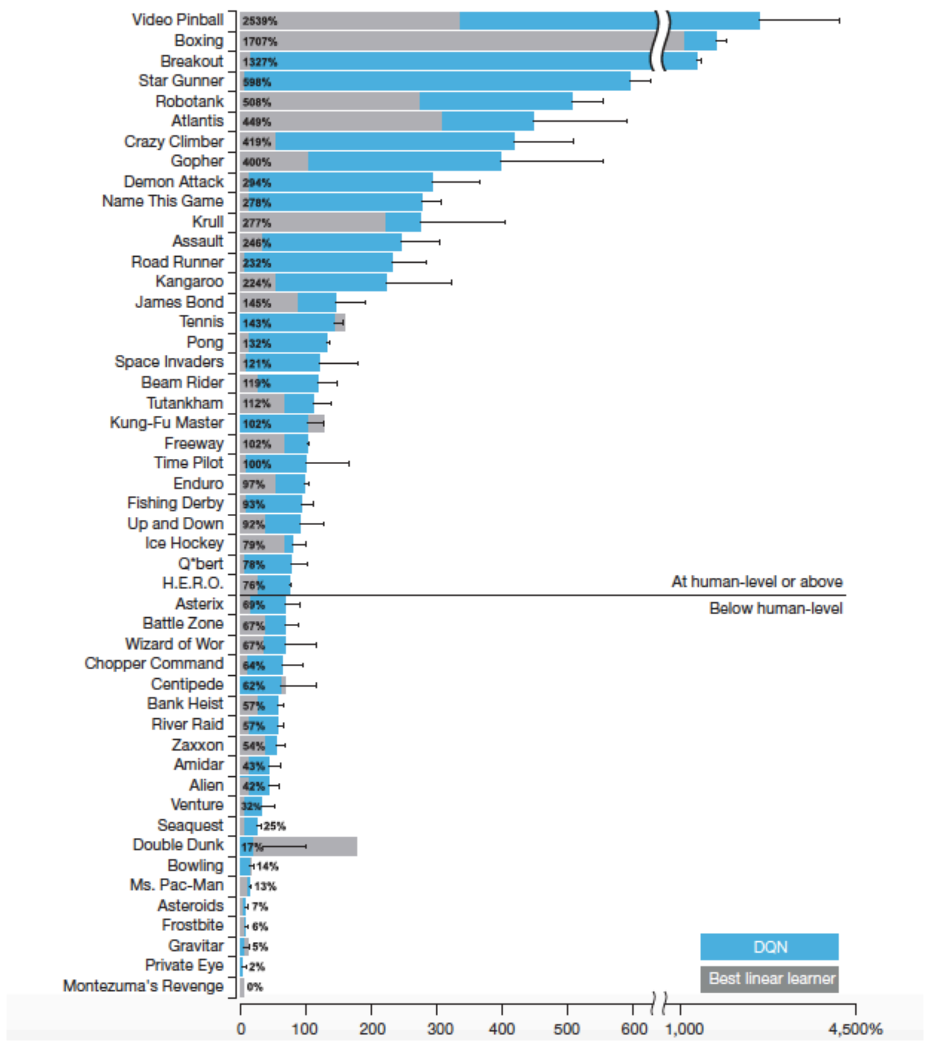
DRL Algorithm

```
initialize replay memory $D$
initialize action-value function $Q$ with random weights
observe initial state $s$
repeat
    select an action $a$
        with probability $\varepsilon$ select a random action
        otherwise select $a = \text{argmax}_{a'} Q(s,a')$
    carry out action $a$
    observe reward $r$ and new state $s'$
    store experience $<s, a, r, s'>$ in replay memory D

    sample random transitions $<ss, aa, rr, ss'>$ from replay memory D
    calculate target for each minibatch transition
        if ss' is terminal state then $tt = rr$
        otherwise $tt = rr + \gamma \max_{a'} Q(ss', aa')$
    train the Q network using $(tt - Q(ss, aa))^2$ as loss

    $s = s'$
until terminated
```

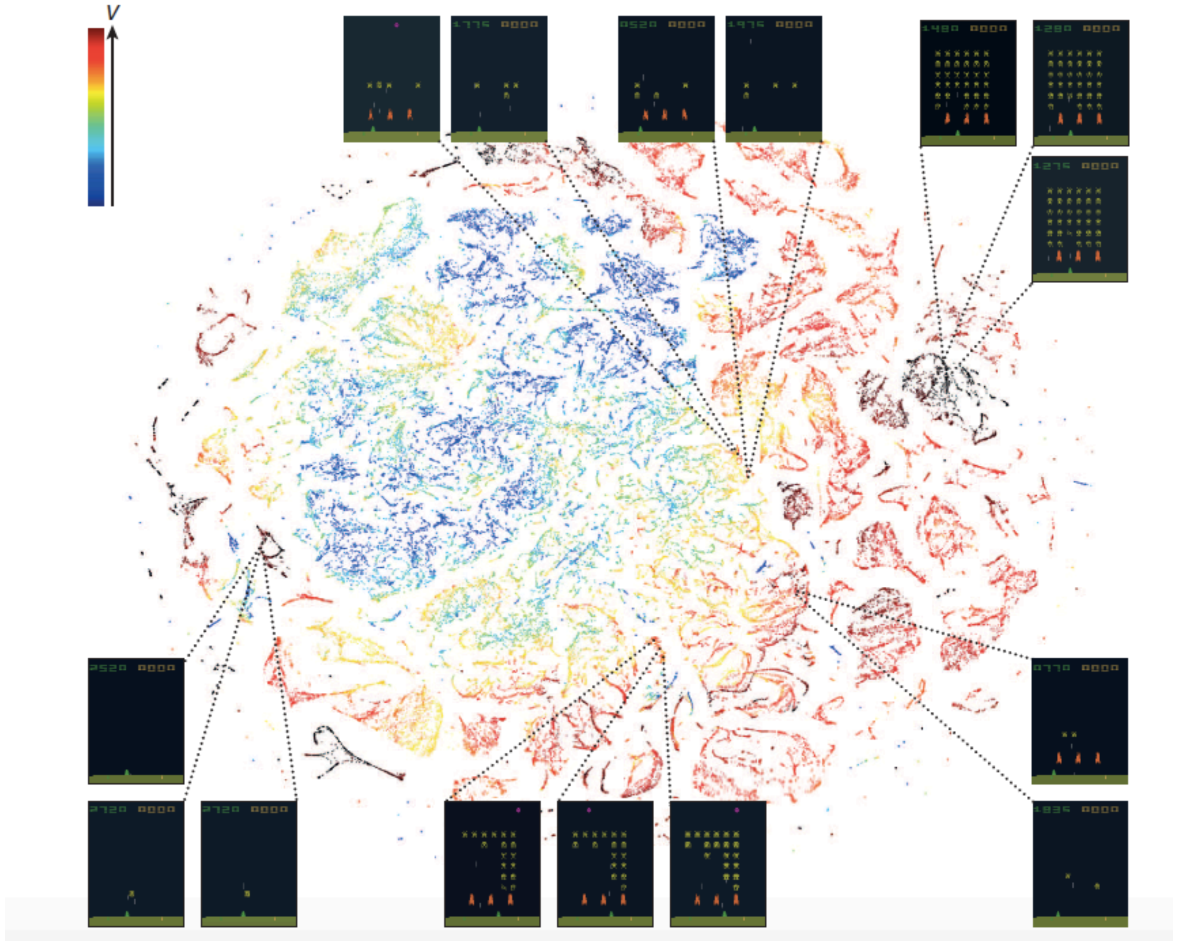
CNN for DQN


Convolution
Convolution
Fully connected
Fully connected
No input

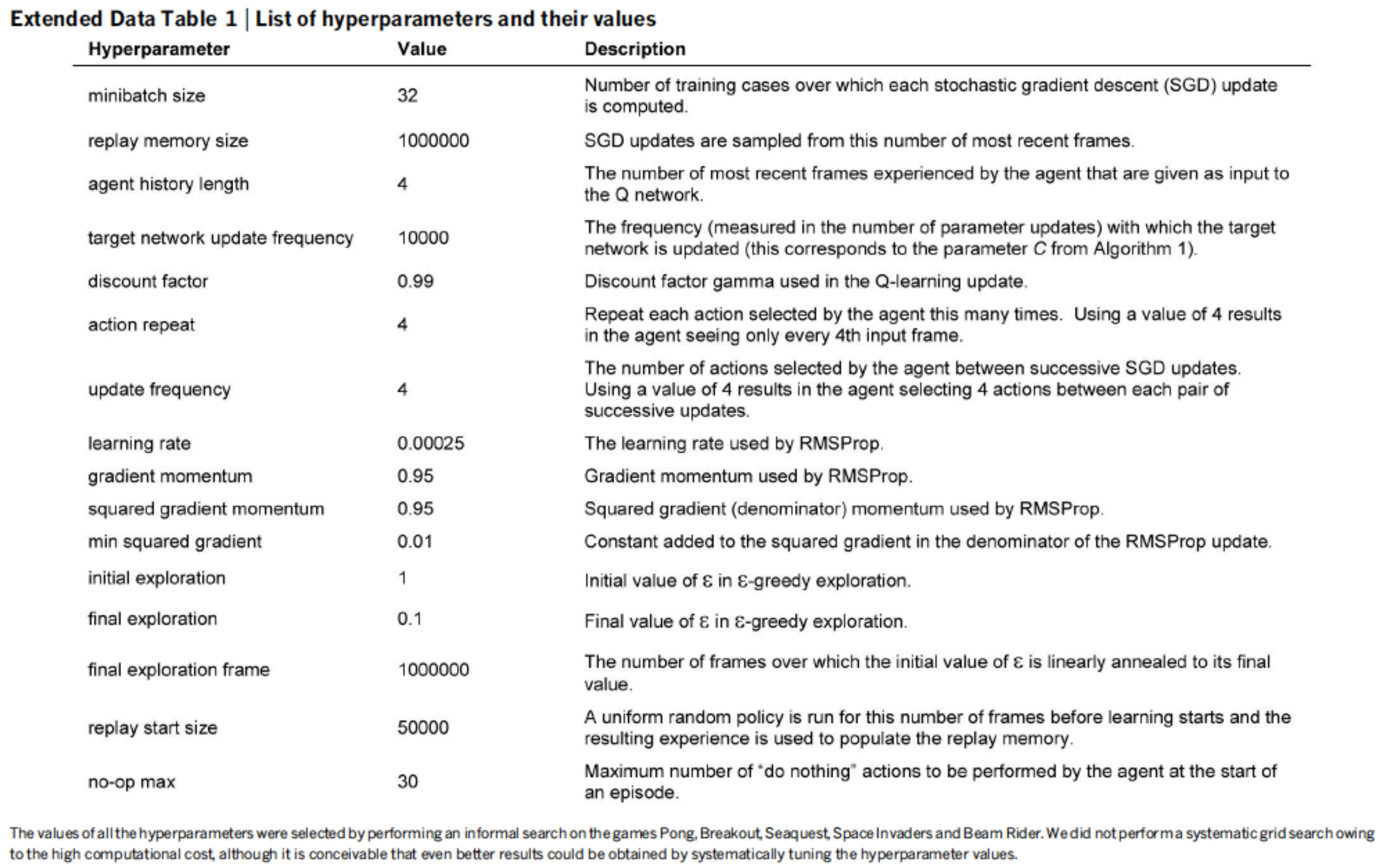
Results on ALE



Space invaders: Visualising last hidden layer (t-distributed stochastic neighbour embedding)



Hyperparameters



Extended Data Table 1 | List of hyperparameters and their values

Hyperparameter	Value	Description
minibatch size	32	Number of training cases over which each stochastic gradient descent (SGD) update is computed.
replay memory size	1000000	SGD updates are sampled from this number of most recent frames.
agent history length	4	The number of most recent frames experienced by the agent that are given as input to the Q network.
target network update frequency	10000	The frequency (measured in the number of parameter updates) with which the target network is updated (this corresponds to the parameter *C* from Algorithm 1).
discount factor	0.99	Discount factor gamma used in the Q-learning update.
action repeat	4	Repeat each action selected by the agent this many times. Using a value of 4 results in the agent seeing only every 4th input frame.
update frequency	4	The number of actions selected by the agent between successive SGD updates. Using a value of 4 results in the agent selecting 4 actions between each pair of successive updates.
learning rate	0.00025	The learning rate used by RMSProp.
gradient momentum	0.95	Gradient momentum used by RMSProp.
squared gradient momentum	0.95	Squared gradient (denominator) momentum used by RMSProp.
min squared gradient	0.01	Constant added to the squared gradient in the denominator of the RMSProp update.
initial exploration	1	Initial value of ε in ε-greedy exploration.
final exploration	0.1	Final value of ε in ε-greedy exploration.
final exploration frame	1000000	The number of frames over which the initial value of ε is linearly annealed to its final value.
replay start size	50000	A uniform random policy is run for this number of frames before learning starts and the resulting experience is used to populate the replay memory.
no-op max	30	Maximum number of "do nothing" actions to be performed by the agent at the start of an episode.

The values of all the hyperparameters were selected by performing an informal search on the games Pong, Breakout, Seaquest, Space Invaders and Beam Rider. We did not perform a systematic grid search owing to the high computational cost, although it is conceivable that even better results could be obtained by systematically tuning the hyperparameter values.

A. Tamar et al., Value iteration networks, In Proc. *Neural Information Processing Systems*, 2016

[See slides by authors]