Reinforcement Learning

Lecture 10: Extended methods

Chris G. Willcocks

Durham University

Lecture overview

1 More approaches

- DQN characteristics
- distributed and recurrent RL
- R2D2 performance
- More exploration approaches

2 More rewards

- NGU: intrinsic motivation and curiosity
- The reward-is-enough hypothesis

3 More architectures

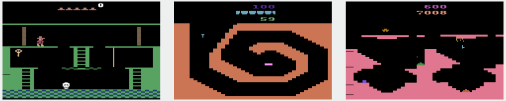
- Dreamer and DreamerV2
- AlphaStar and looking forward
- The bitter lesson
- self-play and league-play

More approaches long-term reward horizons

Characteristics: DQN

DQNs optimise a function (neural network) to predict the Q-value (the expected reward) for a given state and aciton.

DQN doesn't work very well for long-term credit assignments:



Recap: function approximation

The function can be approximated:

```
Q = np.zeros([n_states, n_actions])
a_p = Q[s,:]

# action—value table is approximated:
a_p = DeepNeuralNetwork(s)
```

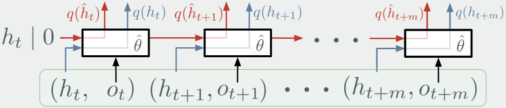
Remember this usually requires several extra tricks to work:

- Double DQN
- Prioritised replay
- Distributed RL

Definition: R2D2

Recurrent Replay Distributed DQN (R2D2) [1] uses RNNs, training on a sequence of $m = 80$ observations o_t and hidden states h_t:

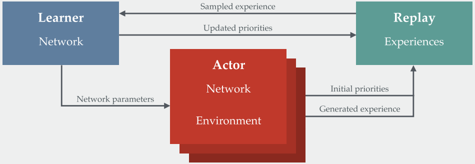
Computation of $\triangle Q$



Therefore it can backpropagate through the history, updating where earlier actions led to long-term future reward.

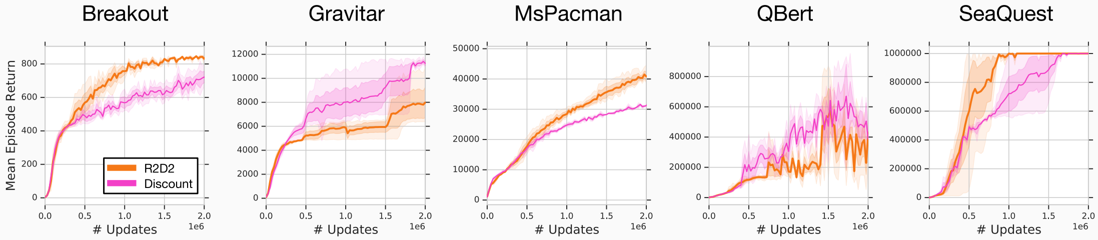
Definition: distributed RL

In distributed RL [2], a central learner (with some parameters θ) receives experience from multiple parallel workers $w_1, w_2, \ldots, w_n$ which run episodes independently:



Distributed and recurrent RL R2D2 performance

These graphs shows R2D2 performance for $\gamma = 0.99$ (pink) vs $\gamma = 0.997$ (orange):



Watch R2D2 play Gravitar

Watch R2D2 play other Atari

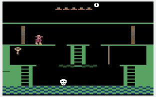
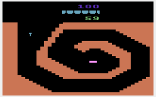
Exploration vs exploitation

R2D2 is not good at balancing exploration vs exploitation. There are other exploration strategies besides taking random actions:

- random exploration, as before:
  - ϵ-greedy
  - softmax
- optimisim in the face of uncertainty
  - estimate uncertainty of the value
  - prefer exploring states/actions with higher uncertainty
- information state space
  - the agent information is part of the state description
  - quantifies state information value

Exploration in Gravitar and AoE

Randomly choosing isn't always good:

Definition: intrinsic reward

Never Give Up (NGU) [3] extends R2D2 by adding an intrinsic reward R', which is where the agent adds its own reward on top of the environment reward:

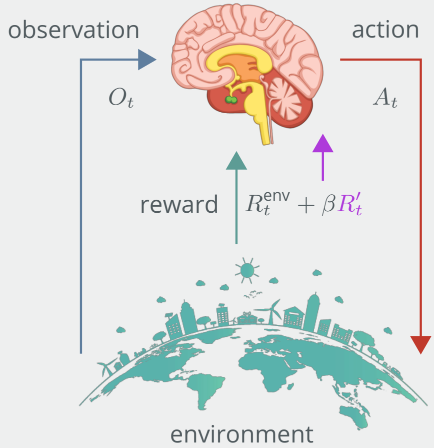
$$R_t = R_t^{\text{env}} + \beta R_t',$$

where β weights the exploration according to its intrinsic reward (e.g. curiosity).

Specifically, it adds a reward for finding things that it has not yet seen before.

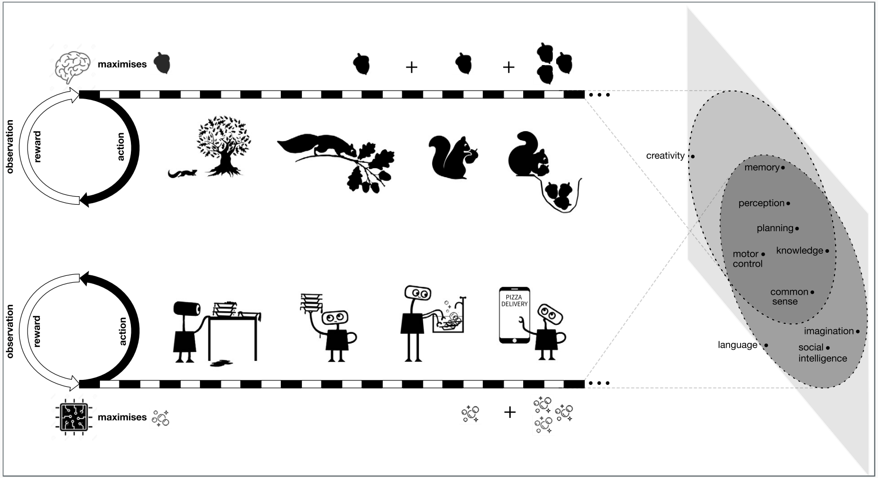
- intrinsic motivation
- curiosity
- novelty

Example: intrinsic reward



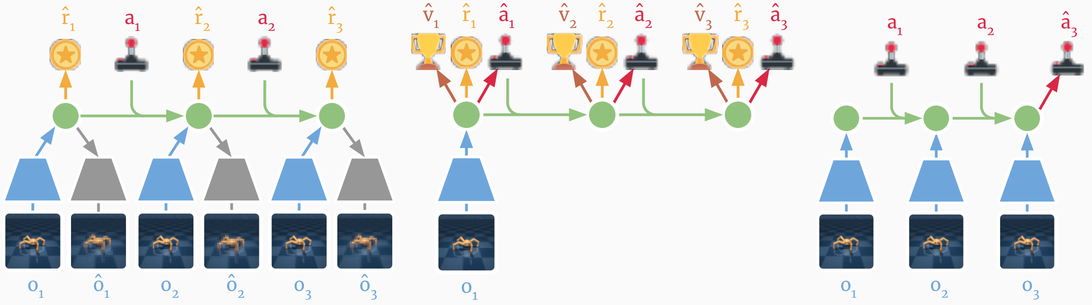
More rewards the reward-is-enough hypothesis

Reward is enough [4] (Silver & Sutton). Others argue for intrinsic rewards in practice.



More architectures latent recurrent imagination

Dreamer [5] and DreamerV2 [6] use a recurrent neural network to 'imagine' and plan ahead, all in the latent (feature representation) space:



(a) Learn dynamics from experience (b) Learn behavior in imagination (c) Act in the environment

Initialize dataset $\mathcal{D}$ with S random seed episodes. Initialize neural network parameters θ, ϕ, ψ randomly.
while *not converged* **do**
  for *update step* $c = 1..C$ **do**
    `// Dynamics learning`
    Draw B data sequences $\{(a_t, o_t, r_t)\}_{t=k}^{k+L} \sim \mathcal{D}$.
    Compute model states $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$.
    Update θ using representation learning.
    `// Behavior learning`
    Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ from each s_t.
    Predict rewards $\mathrm{E}\big(q_\theta(r_\tau \mid s_\tau)\big)$ and values $v_\psi(s_\tau)$.
    Compute value estimates $\mathrm{V}_\lambda(s_\tau)$ via Equation 6.
    Update $\phi \leftarrow \phi + \alpha \nabla_\phi \sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)$.
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi \sum_{\tau=t}^{t+H} \frac{1}{2} \big\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau) \big\|^2$.
  `// Environment interaction`
  $o_1 \leftarrow$ `env.reset()`
  for *time step* $t = 1..T$ **do**
    Compute $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$ from history.
    Compute $a_t \sim q_\phi(a_t \mid s_t)$ with the action model.
    Add exploration noise to action.
    $r_t, o_{t+1} \leftarrow$ `env.step(`a_t`)`.
  Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_t, a_t, r_t)_{t=1}^{T}\}$.

Model components

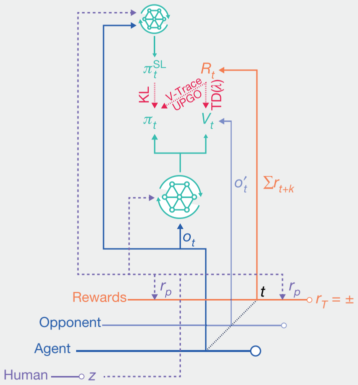
Representation	$p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$
Transition	$q_\theta(s_t \mid s_{t-1}, a_{t-1})$
Reward	$q_\theta(r_t \mid s_t)$
Action	$q_\phi(a_t \mid s_t)$
Value	$v_\psi(s_t)$

Hyper parameters

Seed episodes	S
Collect interval	C
Batch size	B
Sequence length	L
Imagination horizon	H
Learning rate	α

Architecture

AlphaStar [7] uses many components, supervised learning, and league-play.



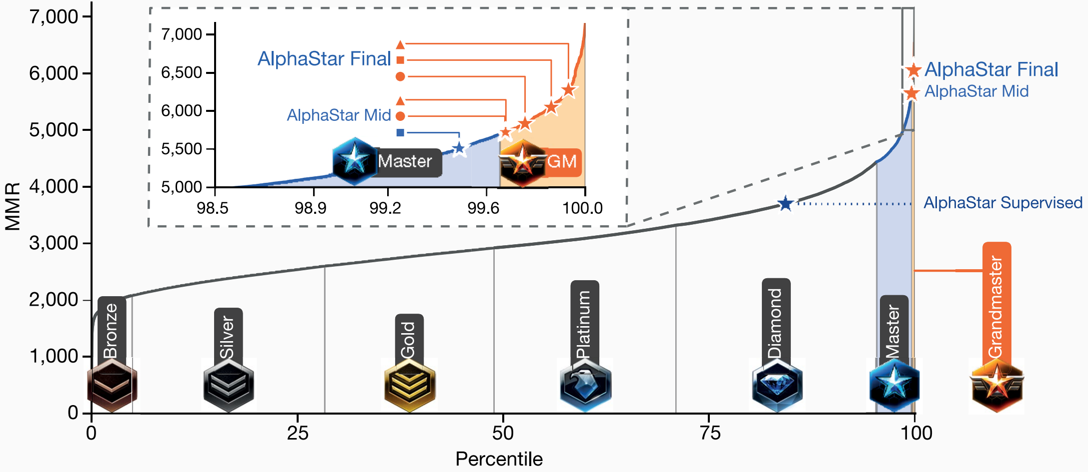
The bitter lesson

Rich Sutton, 2019

☕ Rich Sutton's bitter lesson is that, despite it being tempting to incorporate domain knowledge, general purpose agents win by a large margin.

Link to article

- AI researchers have often tried to build knowledge into their agents
- this always helps in the short term, and is personally satisfying to the researcher
- but in the long run it plateaus and even inhibits further progress
- breakthrough progress eventually arrives by an opposing approach based on scaling computation by search and learning



7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
MMR
0
25
50
75
100
Percentile
Bronze
Silver
Gold
Platinum
Diamond
Master
Grandmaster
AlphaStar Final
AlphaStar Mid
AlphaStar Supervised
7,000
6,500
6,000
5,500
5,000
98.5
98.9
99.2
99.6
100.0
AlphaStar Final
AlphaStar Mid
Master
GM

Take away points

Summary

In summary:

- learn the foundations and concepts of the field, so you can speak the lingo...
- ...but you may want to approach overly complex papers more like an engineer
    - run the code and dismantle it back down to the concepts that make it work
- sample efficiency is an issue, which can be traded for with model-based imagination
- general purpose agents are the future

References I

[1] Steven Kapturowski et al. "Recurrent experience replay in distributed reinforcement learning". In: International Conference on Learning Representations. 2018.

[2] Dan Horgan et al. "Distributed Prioritized Experience Replay". In: International Conference on Learning Representations. 2018.

[3] Adrià Puigdomènech Badia et al. "Never Give Up: Learning Directed Exploration Strategies". In: International Conference on Learning Representations. 2020.

[4] David Silver et al. "Reward is enough". In: Artificial Intelligence (2021), p. 103535.

[5] Danijar Hafner et al. "Dream to Control: Learning Behaviors by Latent Imagination". In: International Conference on Learning Representations. 2020.

[6] Danijar Hafner et al. "Mastering Atari with Discrete World Models". In: International Conference on Learning Representations. 2021.

[7] Oriol Vinyals et al. "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: Nature 575.7782 (2019), pp. 350–354.