Course summary and Research Perspectives

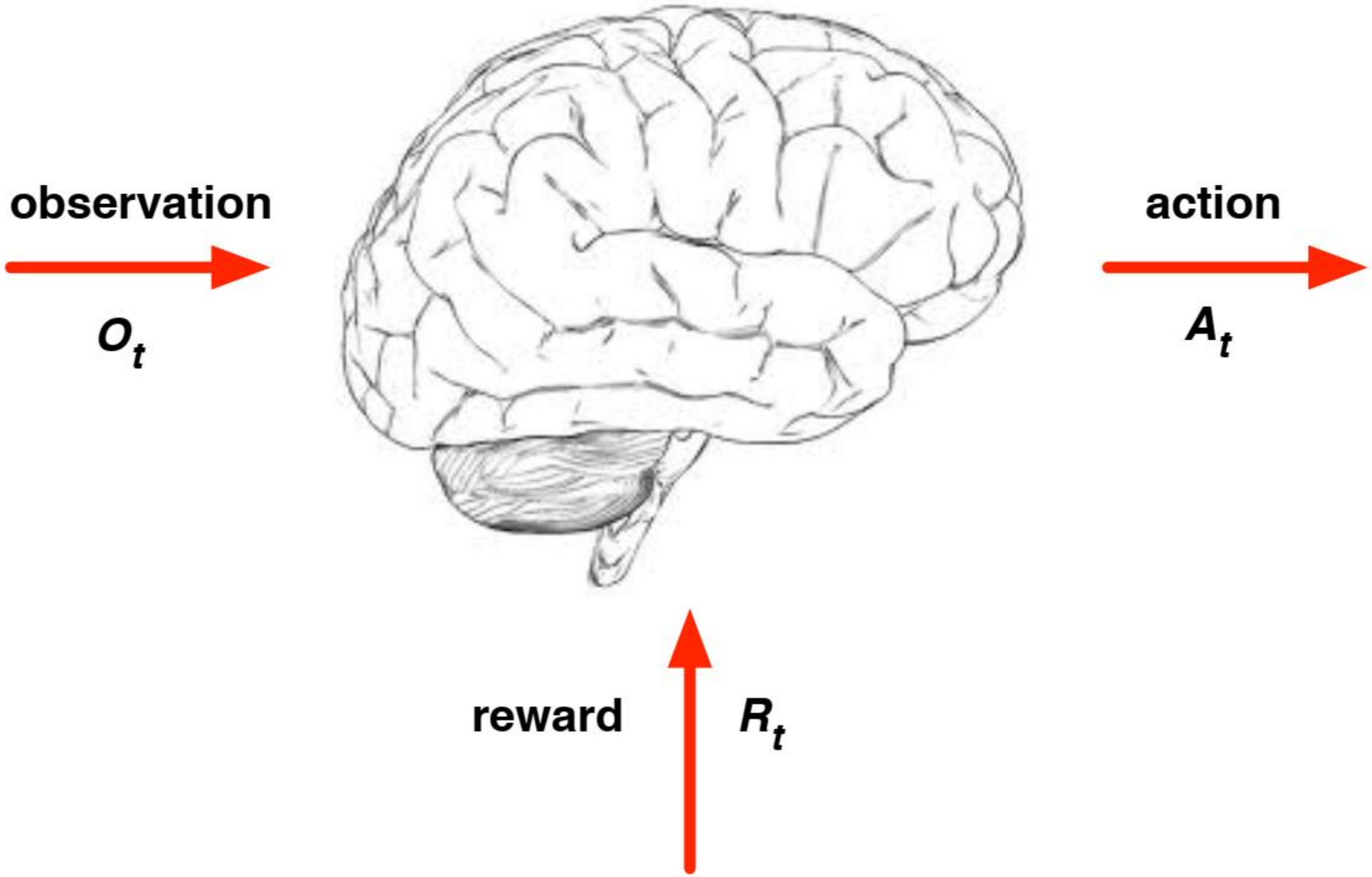
DAVIDE BACCIU – BACCIU@DI.UNIPI.IT

IN SUPREMÆ DIGNITATIS 1343
UNIVERSITÀ DI PISA

Outline

- ✓ Course wrap-up
- ✓ An illustrated ex-post taxonomy of RL
- ✓ Research topics overview
- ✓ Final projects and seminars

Course Wrap-Up



It had all started with this picture

Markov Decision Process

Definition (Markov Decision Process)

A Markov Reward Process is a tuple $\langle \mathcal{S}, \mathcal{A}, \boldsymbol{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a finite set of states
- $\mathcal{A}$ is a finite set of actions a
- $\boldsymbol{P}$ is a state transition matrix, s.t. $P^a_{ss'} = P(S_{t+1} = s' | S_t = s, A_t = a)$
- $\mathcal{R}$ is a reward function, s.t. $\mathcal{R}^a_s = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
- γ is a discount factor, $\gamma \in [0,1]$

Key Components of an RL Agent…

An RL agent may be directly or indirectly trying to learn a

- ✓ Policy - Agent's behaviour function
- ✓ Value function - How good is each state and/or action
- ✓ Model - Agent's representation of the environment

$$S_1, A_1, R_2, \ldots, R_n, S_n$$

...and its *raison d'être*

Choose actions such as to maximize the discounted future reward (the return)

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T$$

3 Types of Reinforcement Learning (plus one)

Value-based

- ✓Learn the state or state-action value
- ✓Act by choosing best action in state
- ✓Exploration is a necessary add-on

Policy-based

- ✓Learn the stochastic policy function that maps state to action
- ✓Act by sampling policy
- ✓Exploration is baked in (trial-and-error)

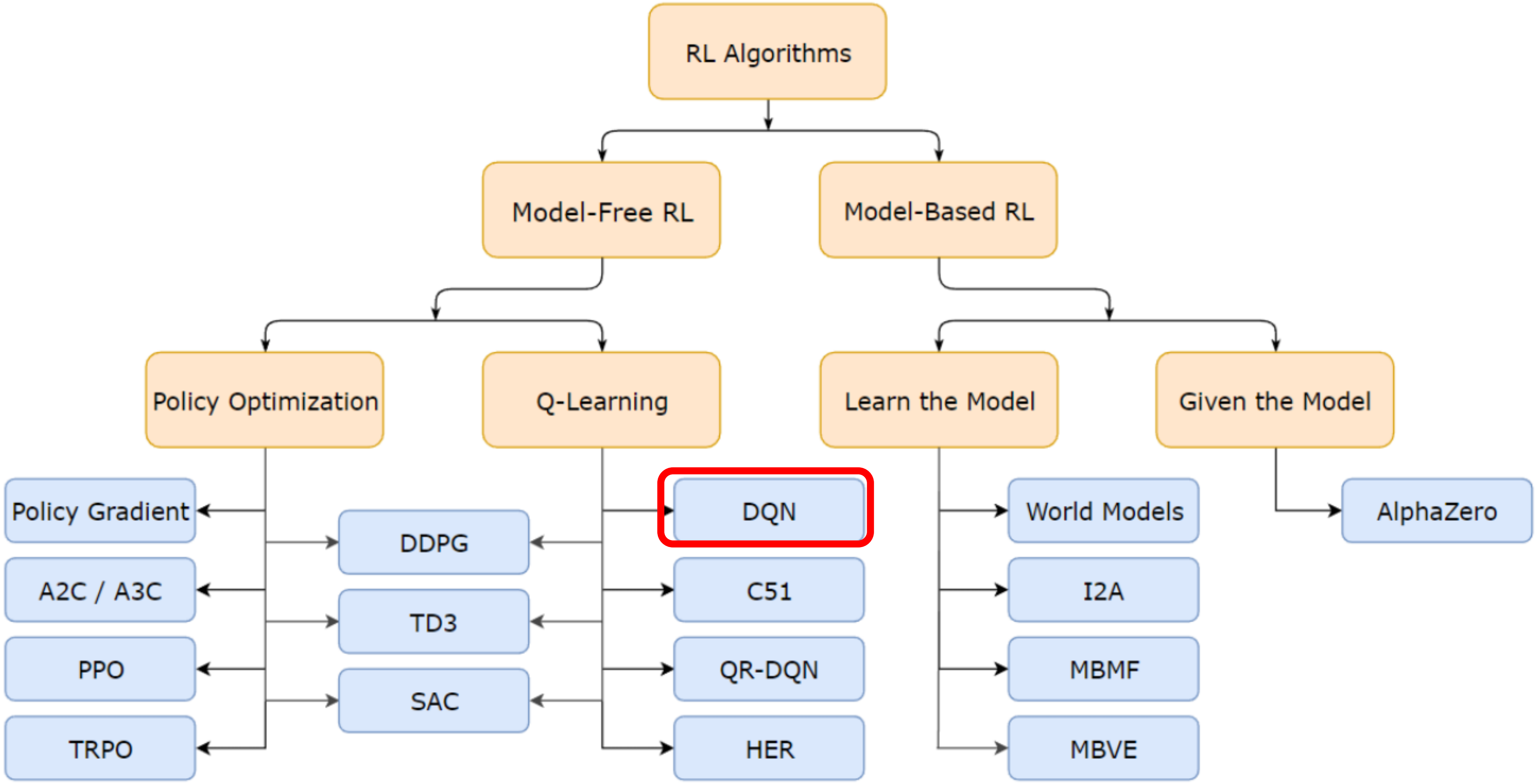
Model-based

- ✓Learn the model of the world, then plan using the model
- ✓Update model often
- ✓Re-plan often

Inverse Reinforcement Learning

- ✓Use demonstrations to learn a reward function and train a policy out of it
- ✓Does not need full demonstrations
- ✓Good generalization

A Taxonomy for RL



Q-Learning

Off-policy learning of action-values $Q(s, a)$

- ✓Use any policy to estimate Q that maximizes future reward
- ✓Independent of the policy being followed
- ✓Only requirement is to keep updating all (s, a) pairs

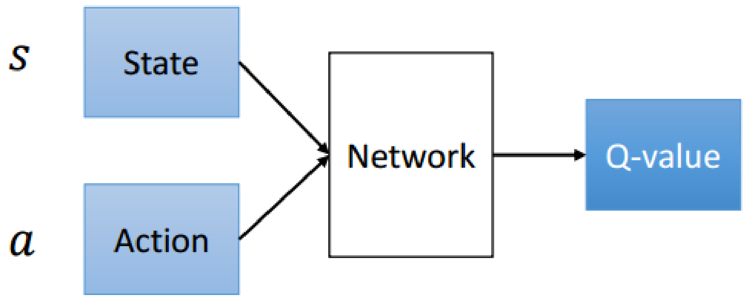
$$Q(S, A) \leftarrow Q(S, A) + \alpha \left(R + \max_{a'} \gamma Q(S', a') - Q(S, A) \right)$$

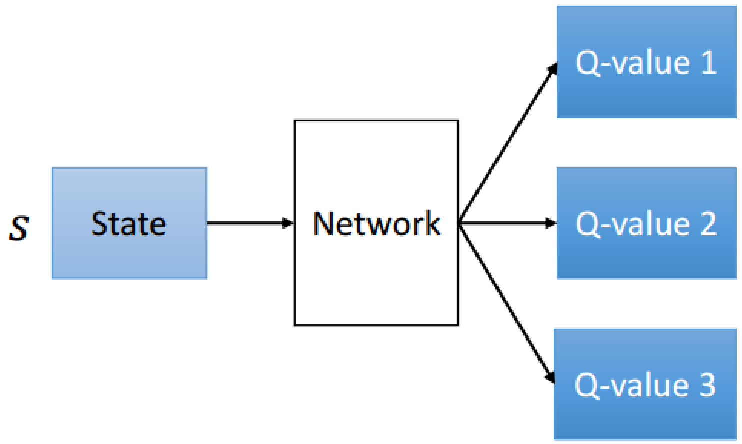
DQN – Deep Q-Learning

- ✓ Use a neural network to approximate the Q-function

$$Q_\theta(S, A) \approx Q^*(S, A)$$

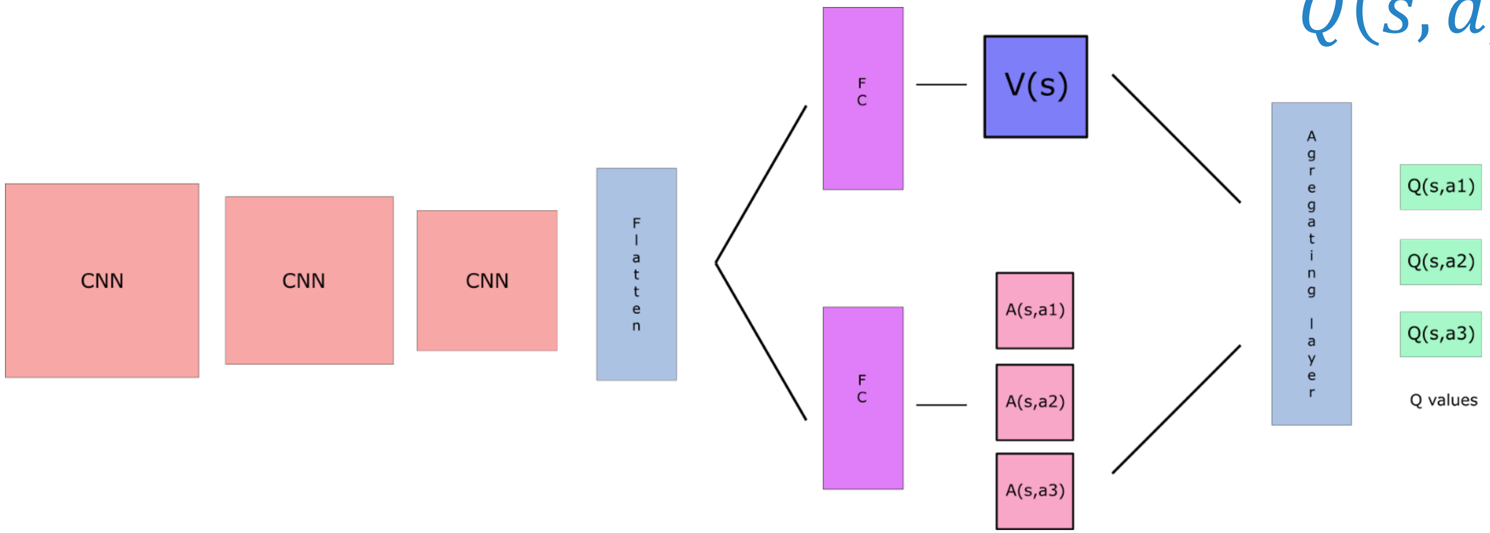
- ✓ Plus some tricks
  - ✓ Fixed targets
  - ✓ Experience replay buffer





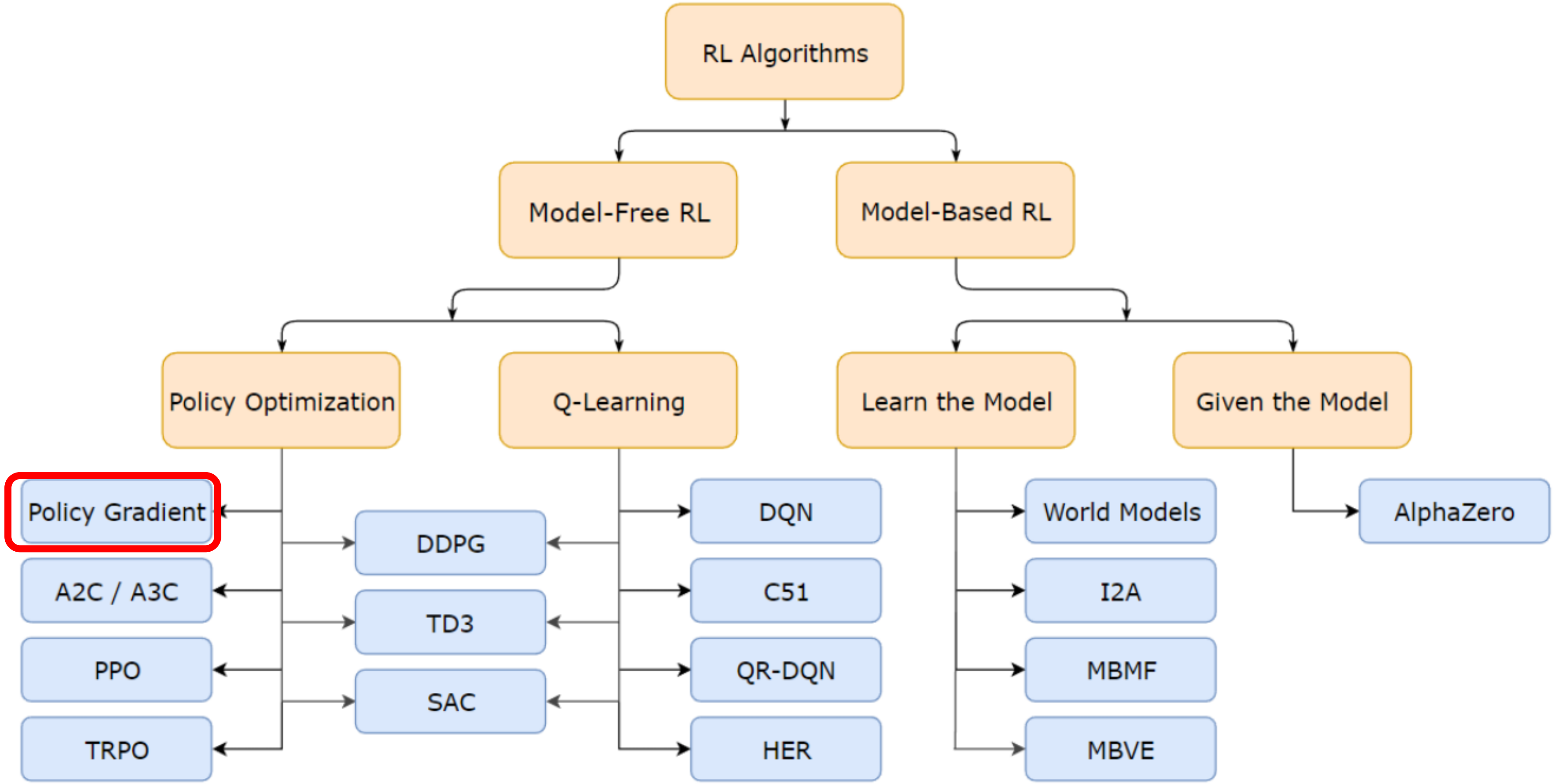
Dueling DQN (DDQN)

$$Q(s,a) = V(s) + A(s,a)$$

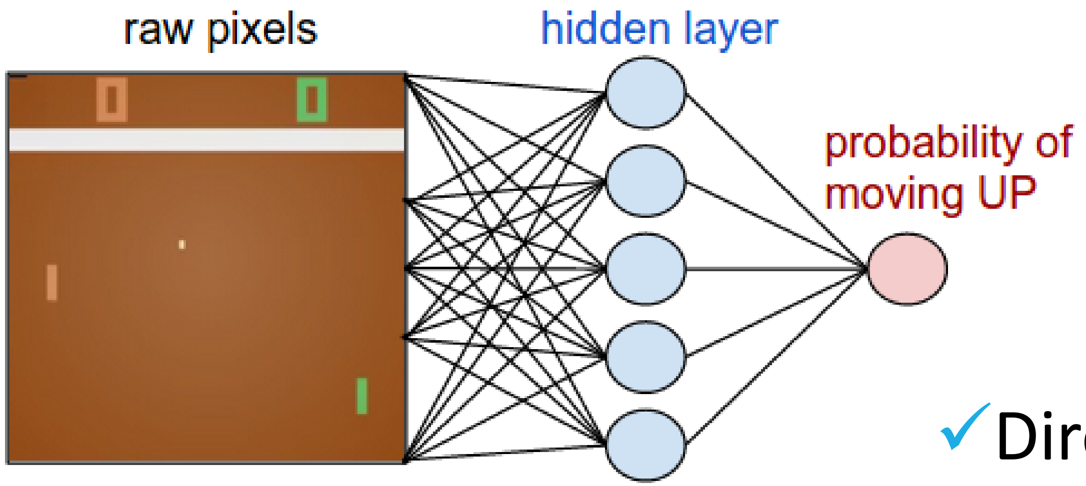


- ✓ $V(s)$ - The value of being is state s
- ✓ $A(s,a)$ - The advantage of taking action a in state s versus all other possible actions at that state
- ✓ Useful for states where action choice does not affect $Q(s,a)$

A Taxonomy for RL



Policy Networks

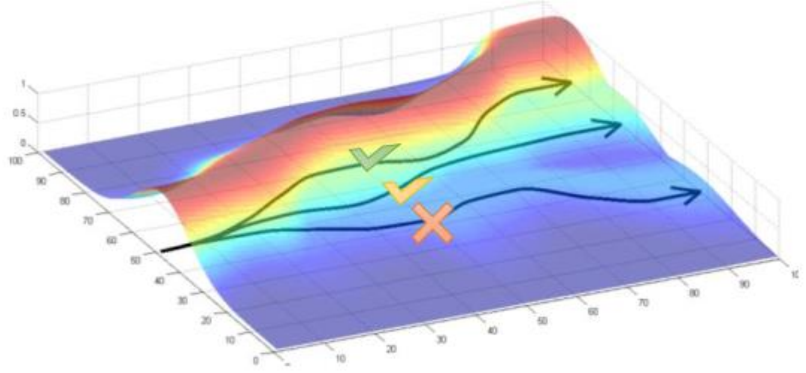


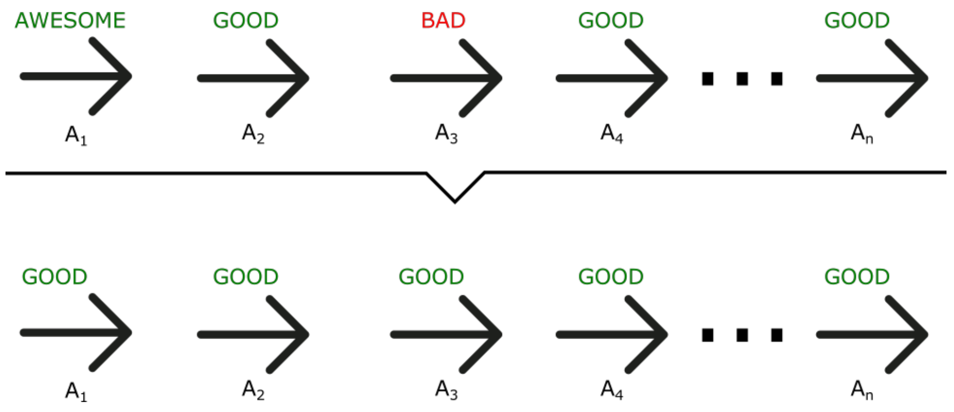
- ✓Directly learn the policy instead of the value function
  - ✓Can learn stochastic policies
  - ✓Faster convergence
  - ✓Good in continuous action spaces

Policy Gradient

REINFORCE - Policy gradient that increases probability of good actions and decreases probability of bad action

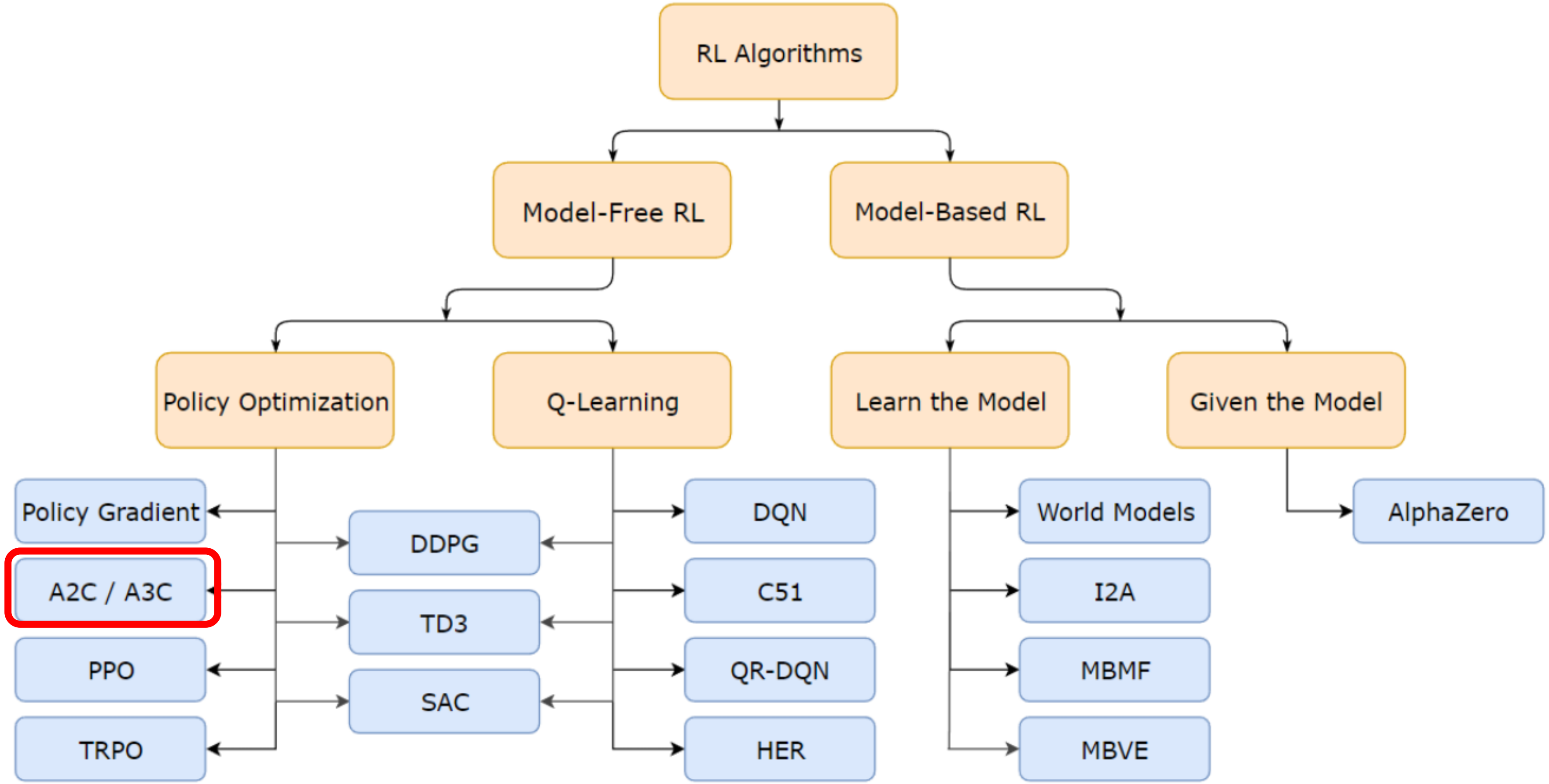
$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) G_t]$$





Working in expectation means sparsely «different» rewards are averaged out by frequent rewards

A Taxonomy for RL



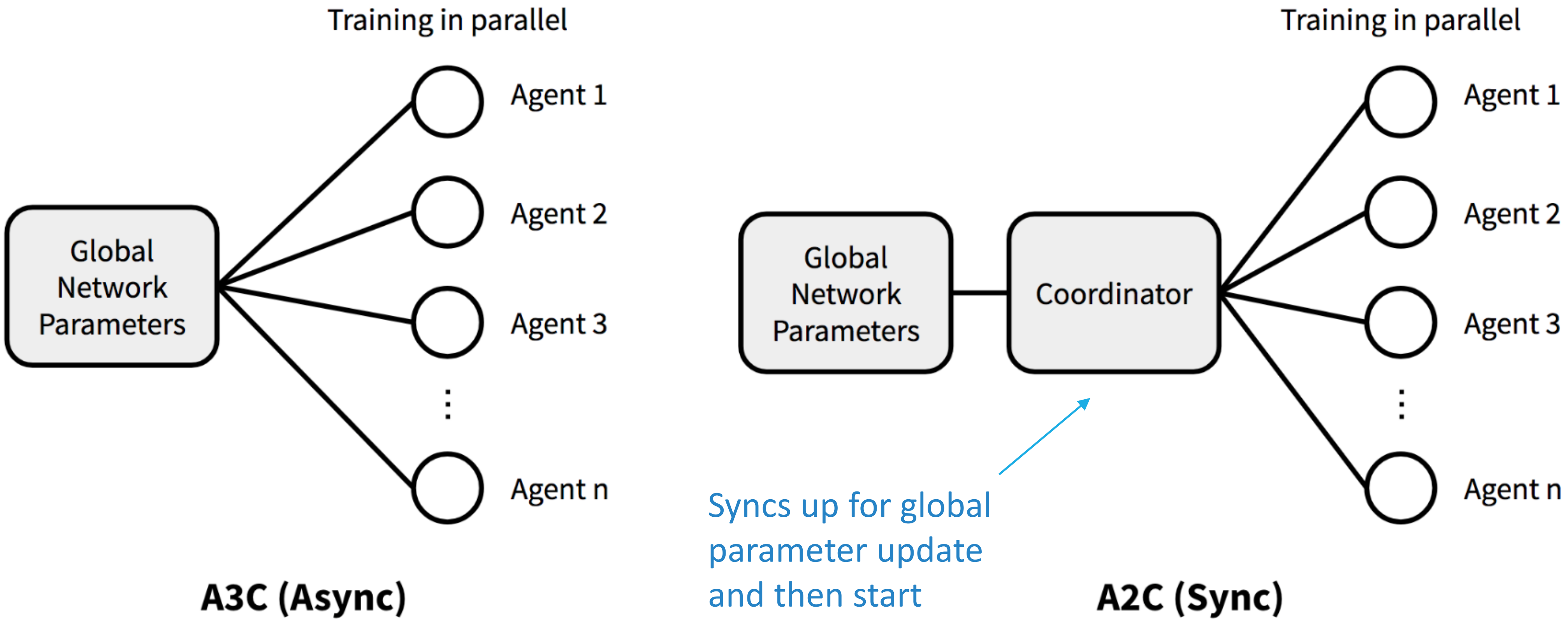
Actor Critic Networks

- ✓ Combine DQN (value-based) and REINFORCE (policy-based)

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_\theta(s,a)]$$

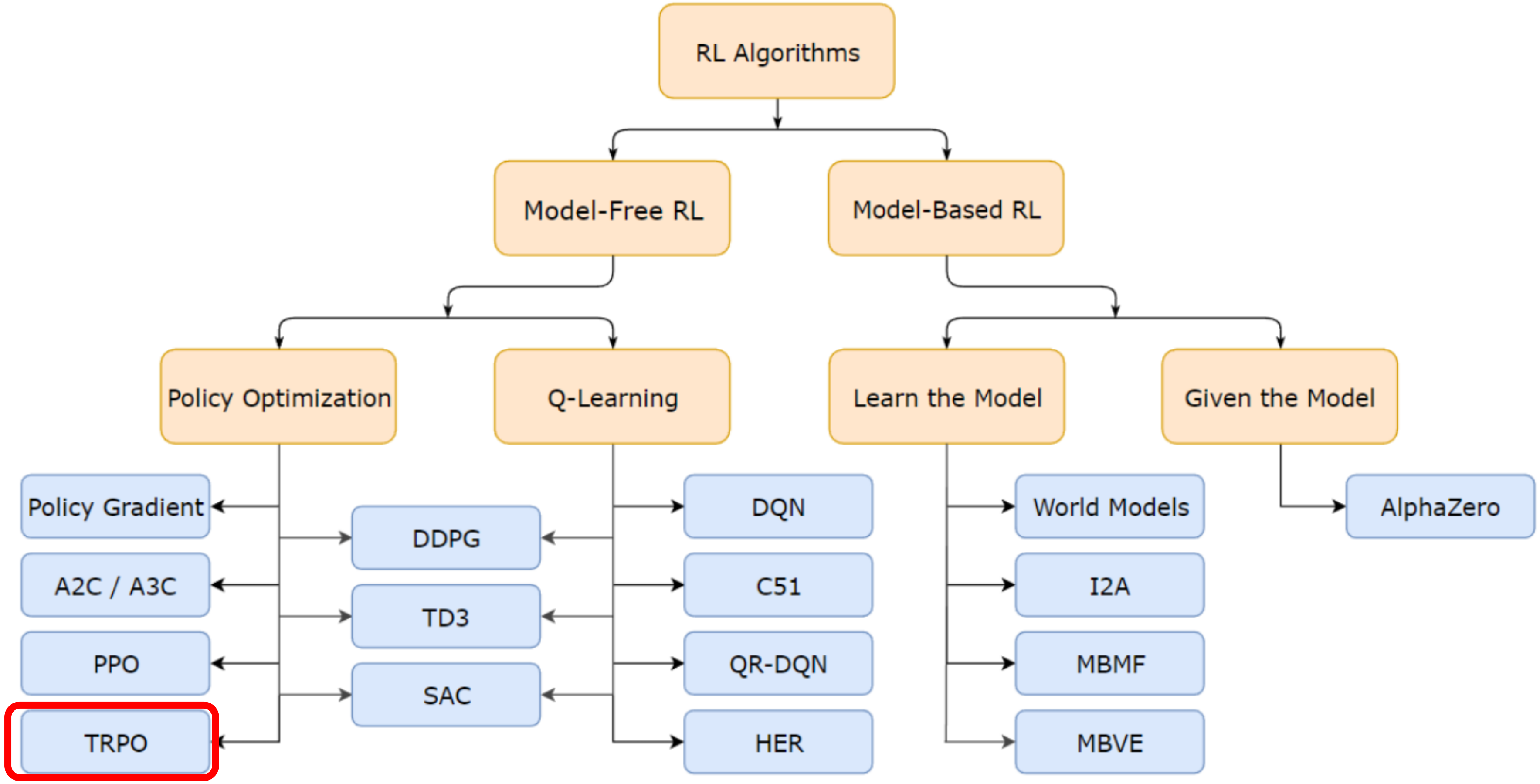
- ✓ Two networks
  - ✓ Actor is policy-based - Samples the action from a policy
  - ✓ Critic is value-based - Measures how good the chosen action is

Advantage Networks (A2C/A3C)



Syncs up for global parameter update and then start each iteration with the same policy

A Taxonomy for RL

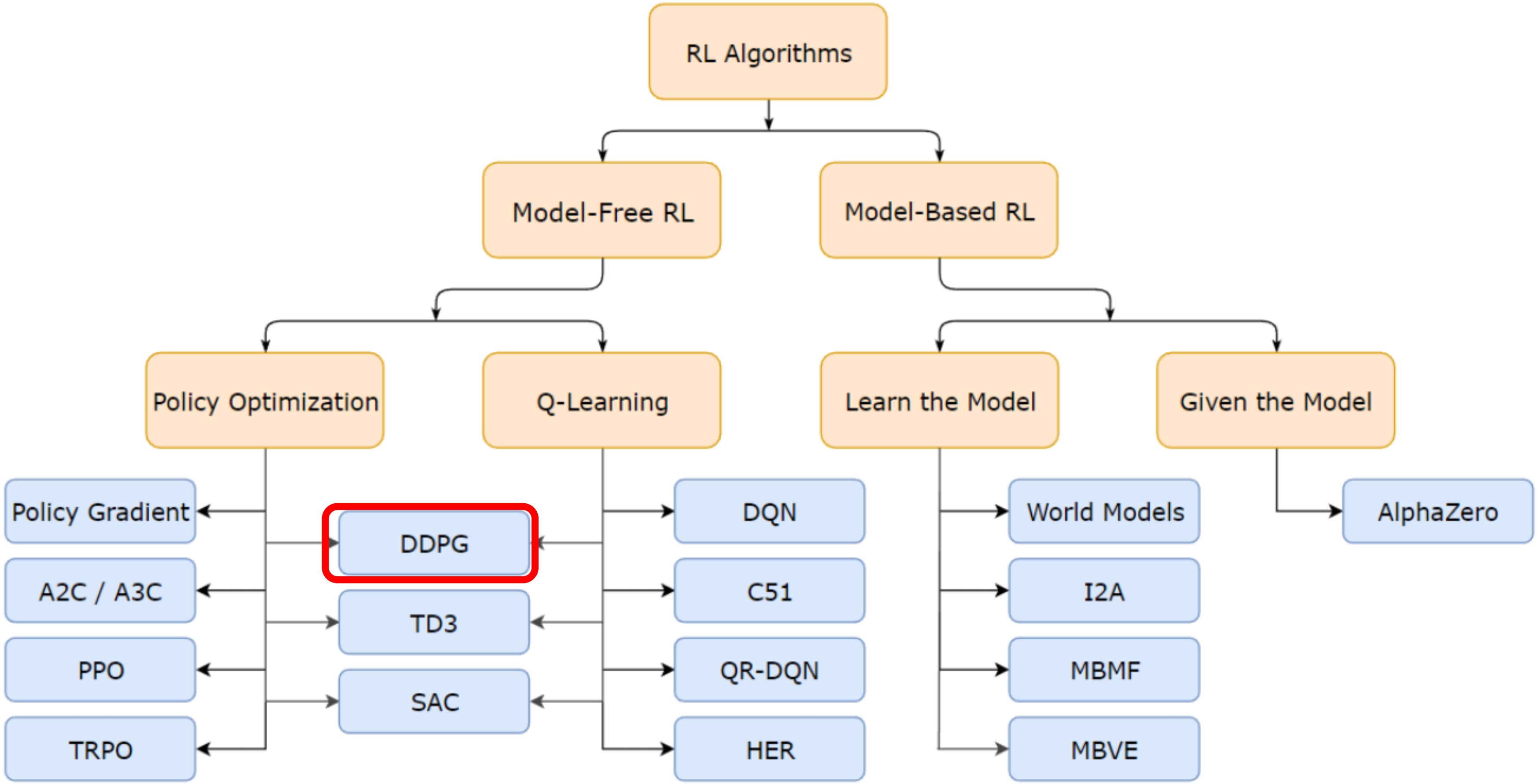


Trust Region Policy Optimization (TRPO)

- ✓ On the importance in avoiding strong discontinuities between old and new policies
- ✓ Avoid taking bad actions that collapse the training performance.
- ✓ Optimizes expected advantage under new policy state distribution
- ✓ Uses importance sampling to align old-new policies in expectation (regularizes to stay close to old policy)

First pick step size, then direction

A Taxonomy for RL



Deep Deterministic Policy Gradients (DDPG)

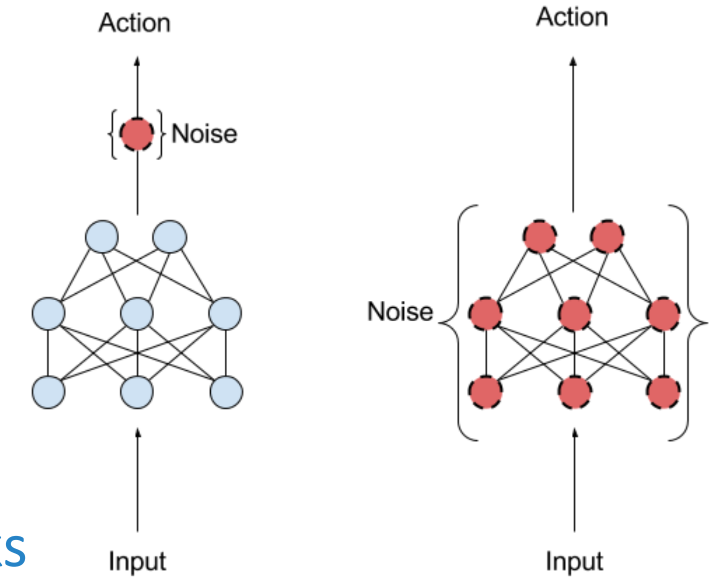
- ✓Actor-Critic framework for learning a deterministic policy
  - ✓Critic estimates value of current policy by DQN

$$J(w) = \left(r + \gamma Q_{w^-}(s', \pi_{u^-}(s')) - Q_w(s,a)\right)^2$$

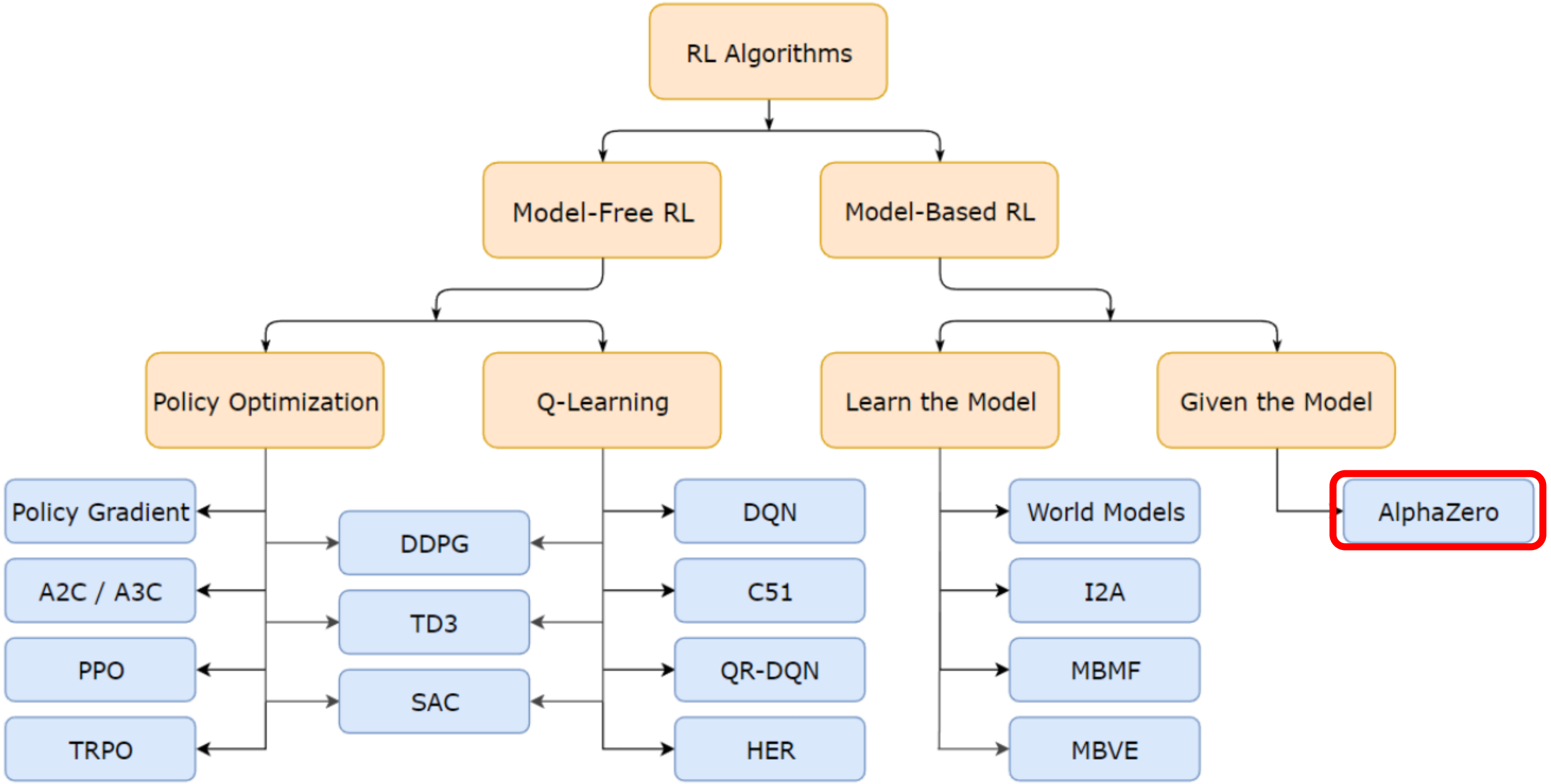
  - ✓Actor updates policy in direction that improves Q

$$\frac{\partial J(u)}{\partial u} = \frac{\partial Q_w(s,a)}{\partial a}\frac{\partial a}{\partial u}$$

- ✓DDPG is the continuous analogue of Deep Q Networks (DQN)
- ✓Exploration - Add noise to actions reducing scale of the noise as training progresses

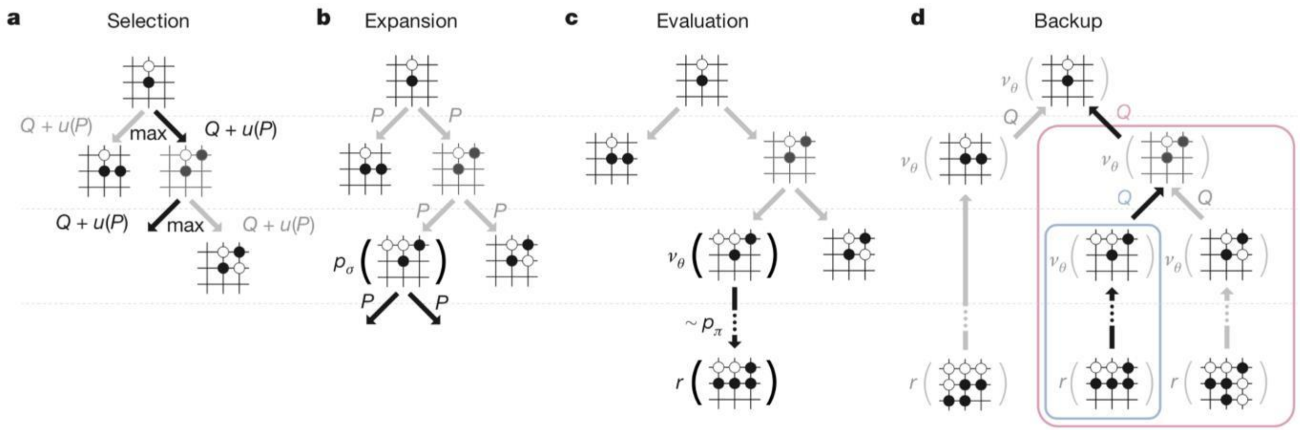


A Taxonomy for RL



AlphaGo Zero – Taking the human out of the equation

- ✓ Monte Carlo Tree Search (MCTS)
  - ✓ Balance exploitation/exploration (going deep on promising positions or exploring new underplayed positions)
  - ✓ Use a neural network as “intuition” for which positions to expand as part of MCTS (same as AlphaGo)



- ✓ Use MCTS intelligent look-ahead (instead of human games) to improve value estimates of play options
- ✓ Multi-task learning – Two headed network that outputs
  1. Move probability
  2. Probability of winning
- ✓ Updated architecture using residual networks

Exploration Vs Exploitation

- ✓Optimism in the face of uncertainty

$$a_t = \arg\max_{a \in \mathcal{A}} Q(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

- ✓Despite what you could have expected, a hideout for heavily theoretical research

RL Research Sneak Peek

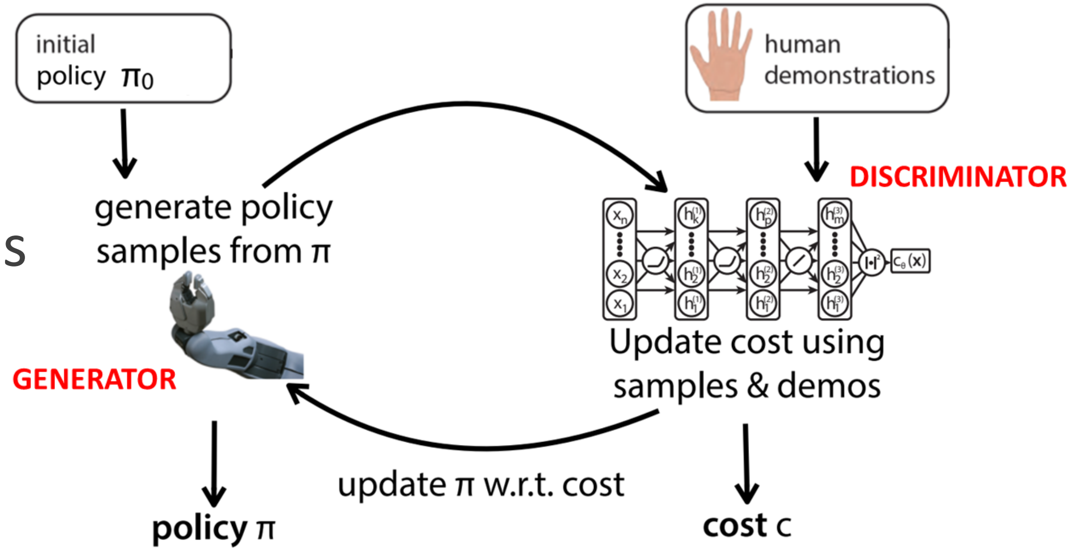
Other Topics We Did Barely Mention

- ✓ Inverse Reinforcement Learning
- ✓ Partial Observability
- ✓ Hierarchical reinforcement learning
- ✓ Curriculum learning
- ✓ Transfer Learning
- ✓ Meta-Learning
- ✓ Continual Learning
- ✓ Multi Agent Reinforcement Learning
- ✓ Scalable reinforcement learning

Some can be good candidates for final projects and seminars

Inverse Reinforcement Learning (IRL)

- ✓Given a policy or examples of the target policy obtain the reward function
- ✓Useful when reinforcement function is unknown or too complex
- ✓More robust than learning from examples (behaviour cloning)
- ✓It heavily relies on generative and adversarial deep learning

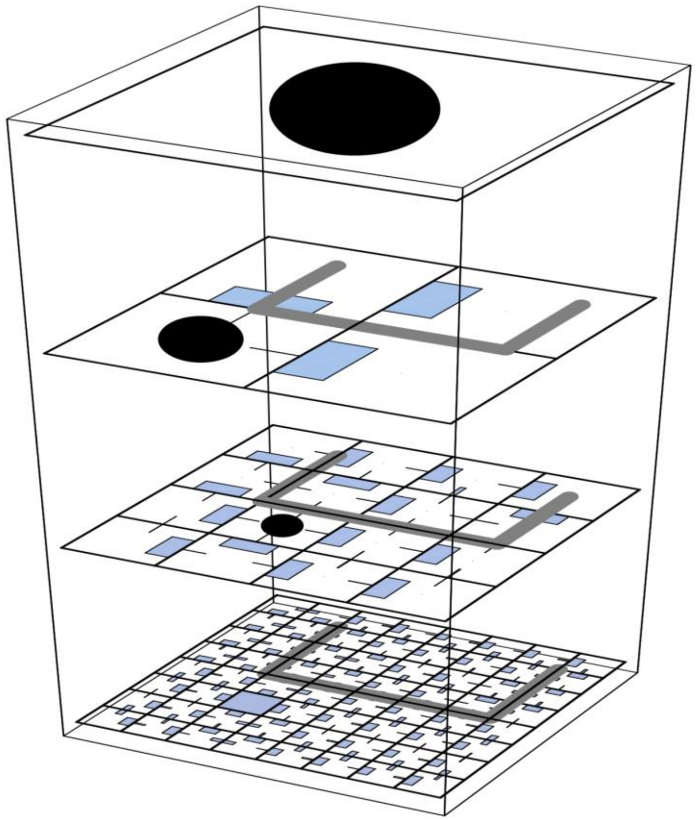


Partially Observability

- ✓Often the agent has not complete information of the true state and uses its perception as state
- ✓Formalize as a POMDP
  - ✓MDP extended with set of observations and probability of each observation given the true state
  - ✓Agent work with a belief vector of probabilities of being in each state
  - ✓Solve with dedicated algorithms
- ✓Use memory to disambiguate the true state
  - ✓Fixed window of last perception (DQN)
  - ✓Recurrent neural networks

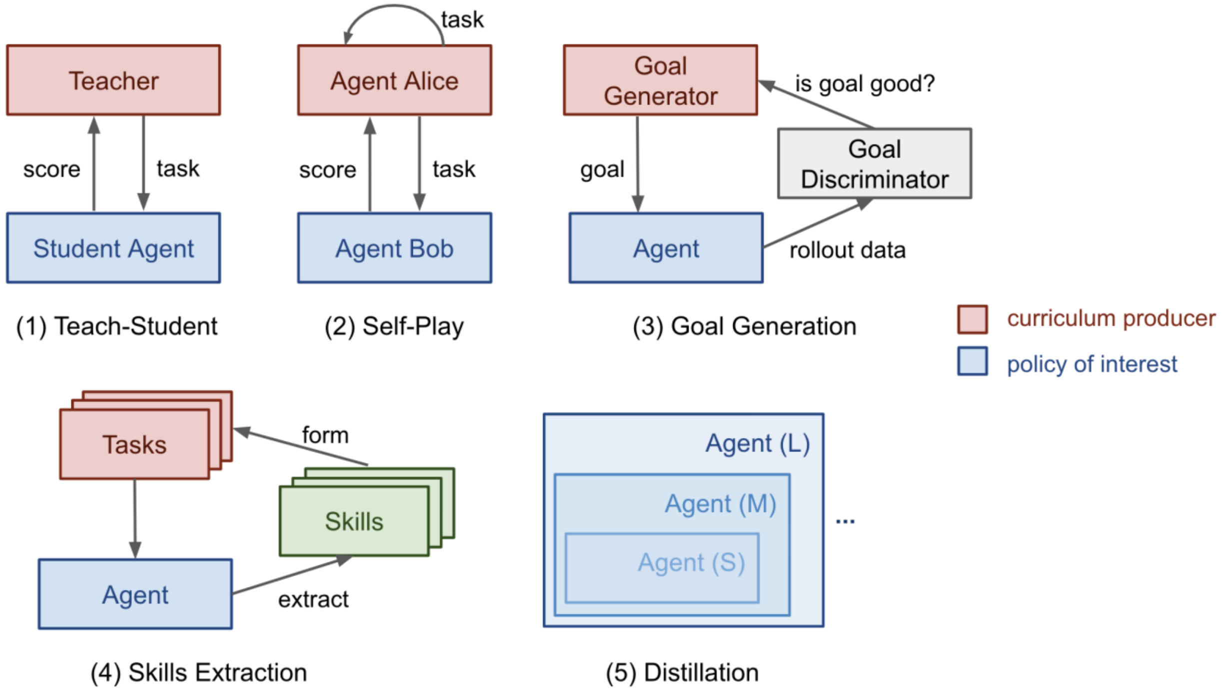
Hierarchical Reinforcement Learning (HRL)

- ✓Reinforcement learning problems suffer from serious scalability issues
  - ✓Often a complex task can be decomposed in simpler tasks
  - ✓Learning can be simplified when these tasks are learnt first
- ✓Key HRL intuition
  - ✓Extend the set of available actions to macro-actions
  - ✓Sequences of elementary actions the agent can choose from
  - ✓Reuse elementary actions to learn other tasks
  - ✓Discover groupings automatically
- ✓Feudal learning, Hierarchical Abstract Machines, MAXQ, HIRO, h-DQN, Abstract MDP



Curriculum Learning

- ✓Learning step by step and only those tasks that you are ready to learn
- ✓Curriculum learning propose subgoals to be learnt in sequence (curriculum) to solve a complex tasks
- ✓See below for a constantly updated review



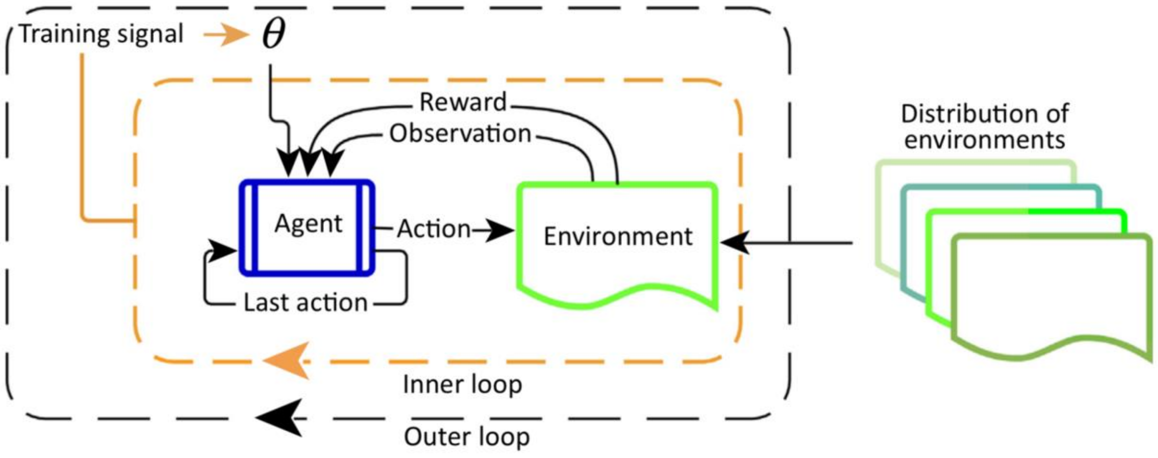
https://lilianweng.github.io/lil-log/2020/01/29/curriculum-for-reinforcement-learning.html

Transfer Learning

- ✓Can we extend knowledge generated in one task to a different task?
- ✓Changes in the task: different dynamics, different reward and/or different actions.
- ✓Different kinds of information to transfer to transfer (Q-values, policy, reward, samples, model, features, etc.)

Meta-Learning

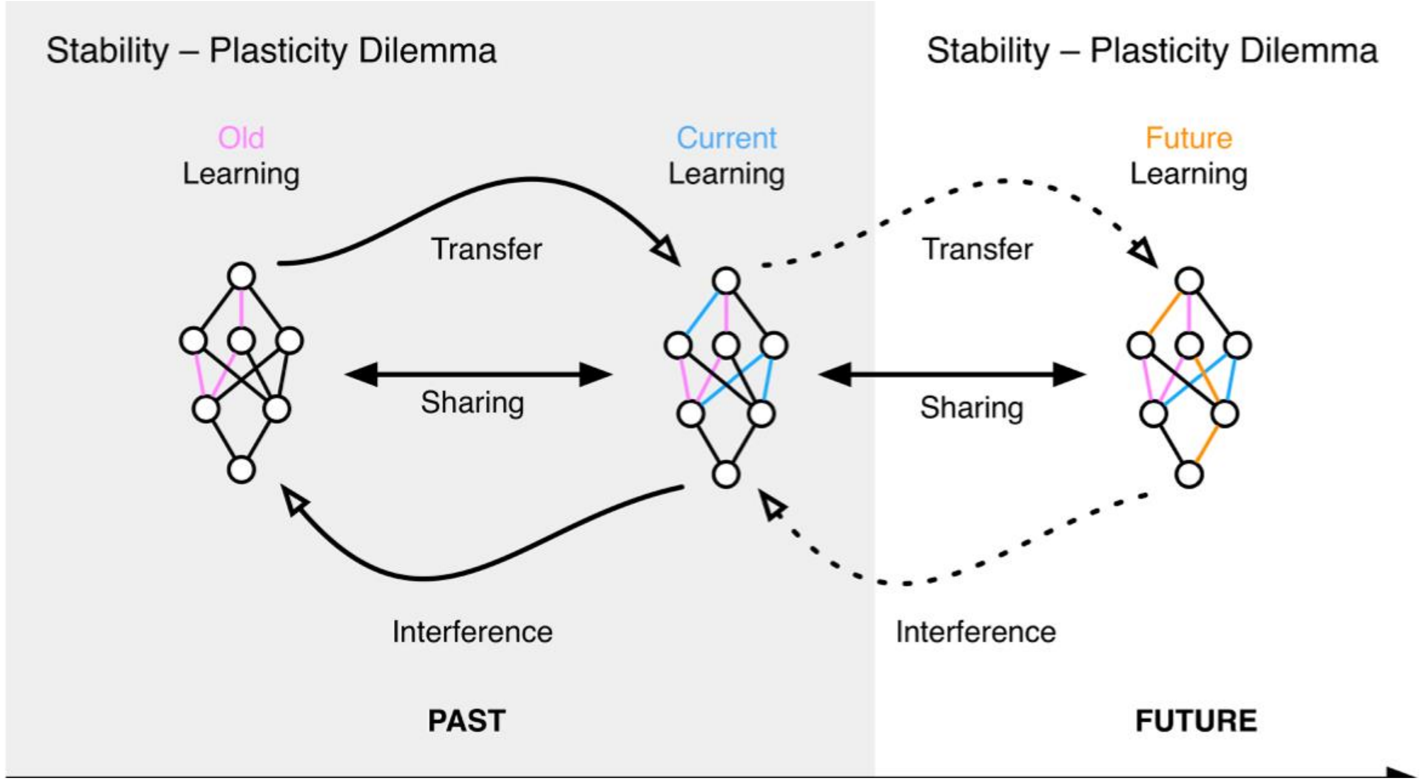
Developing an agent that can solve unseen tasks fast and efficiently



- ✓Three key components
  - ✓Memory - Acquire and memorize the knowledge about the current task
  - ✓Meta-learning algorithm - Update parameters to optimize solving an unseen task fast at test time
  - ✓Distribution learning – Means to infer a distribution over tasks and MDPs to allow fast adaptation at test time

Continual Learning

- ✓ Progressive memory
- ✓ Experience replay
  - ✓ Explicit memory
  - ✓ Generative models
- ✓ Task-space projection and regularization



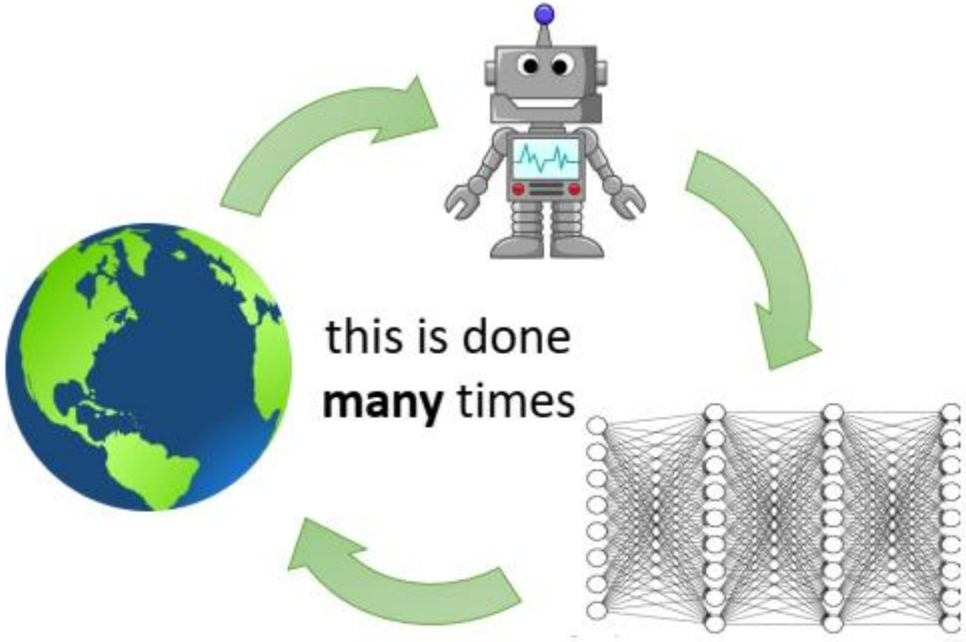
Multi Agent Reinforcement Learning (MARL)

- ✓RL when multiple agents are operating on the same environment
- ✓Use of game theory and assumptions about the other agents
- ✓Depending on the goals of the agent, we have cooperative or competitive learning

https://youtu.be/kopoLzvh5jY

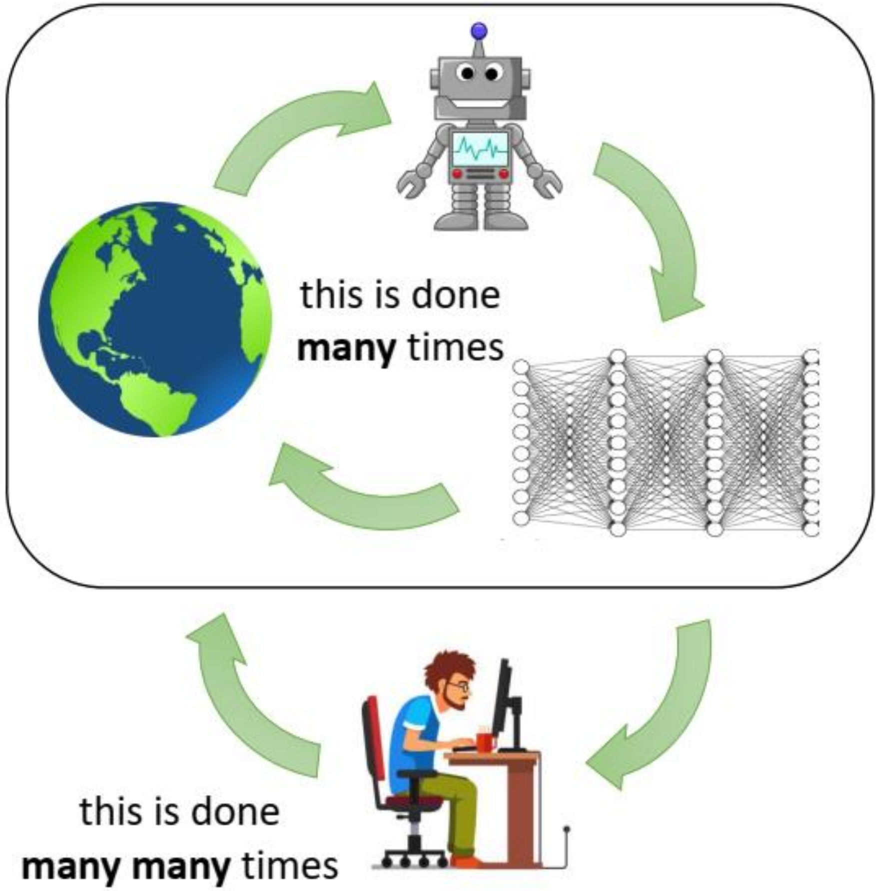
Scalable RL

RL has a big problem



Scalable RL

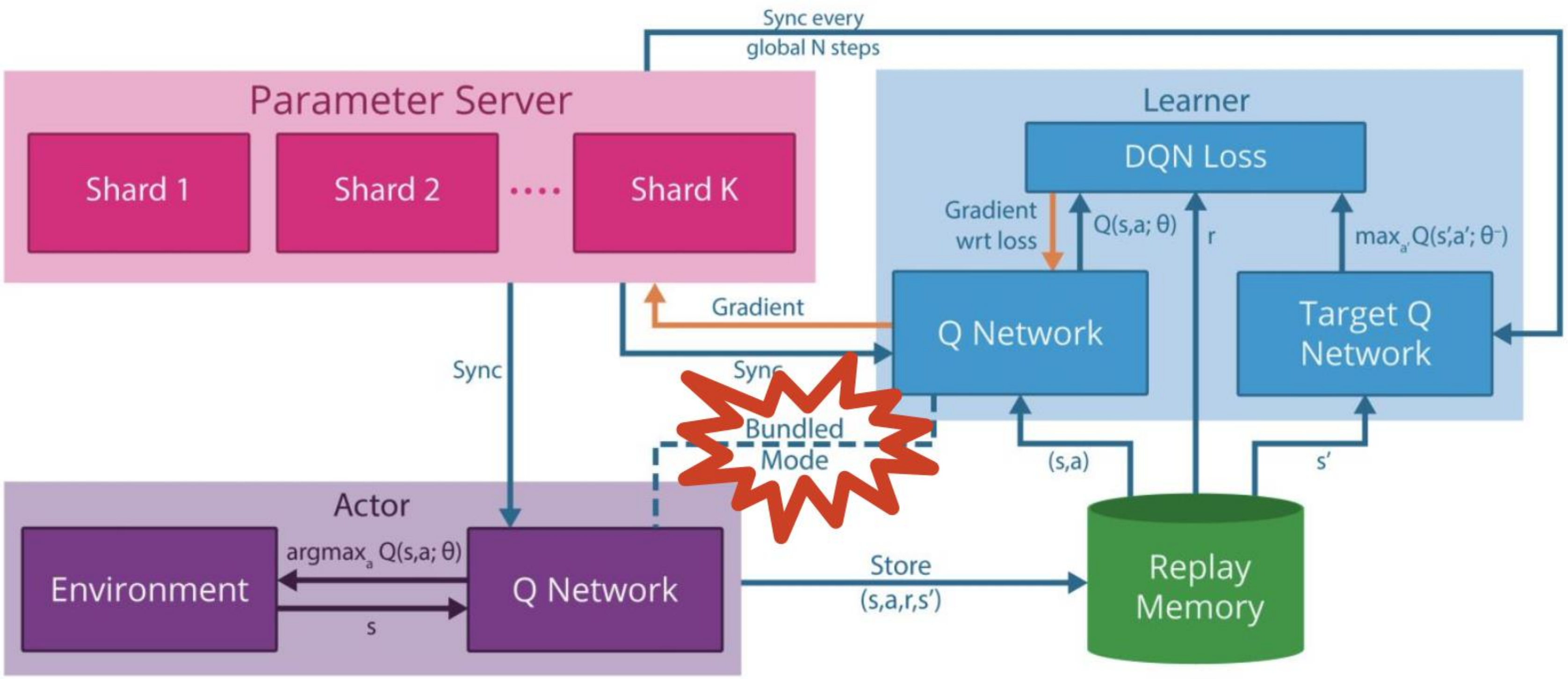
RL has a big problem



…those who work in RL have an even bigger one

Scalable RL – Seeking solution in distributed computing

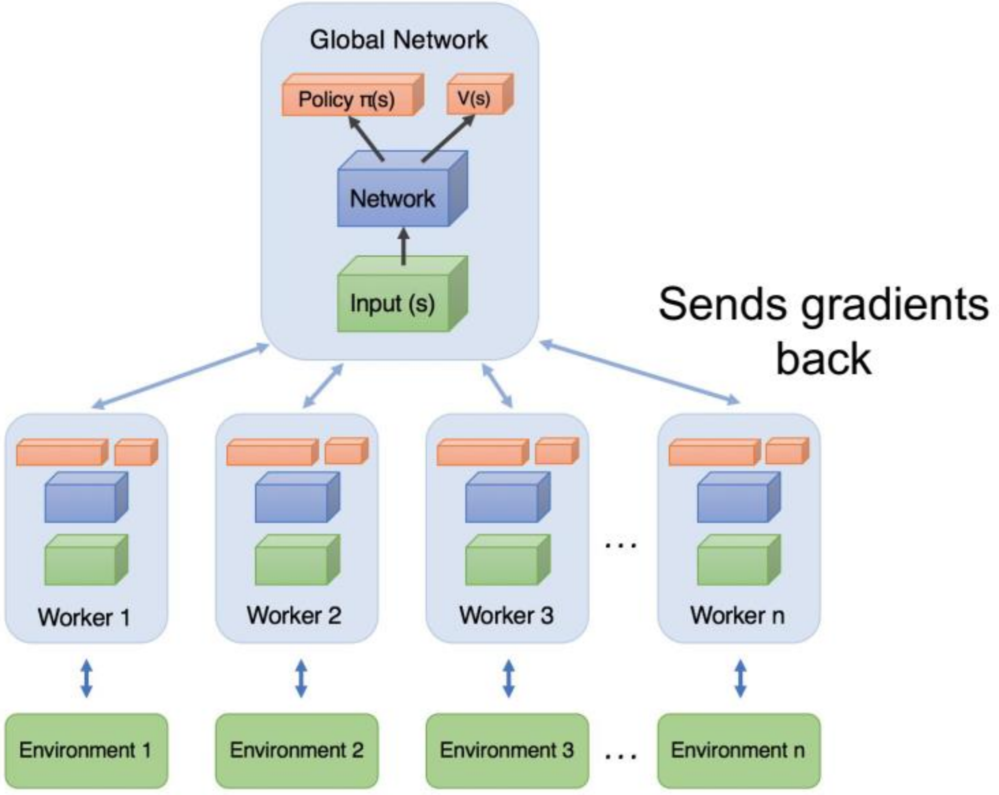
General Reinforcement Learning Architecture (GORILA)



Nair et al, Massively Parallel Methods for Deep Reinforcement Learning, 2015

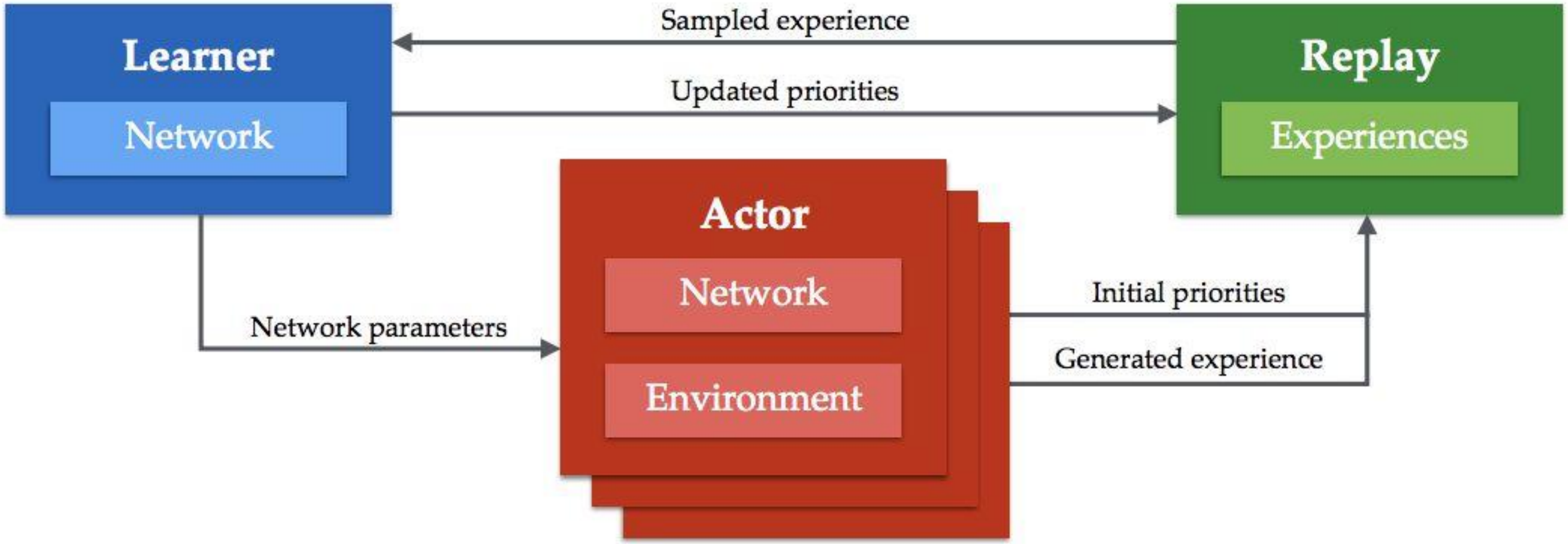
Scalable RL – Seeking solution in distributed computing

Asynchronous Advantage Actor Critic (A3C)



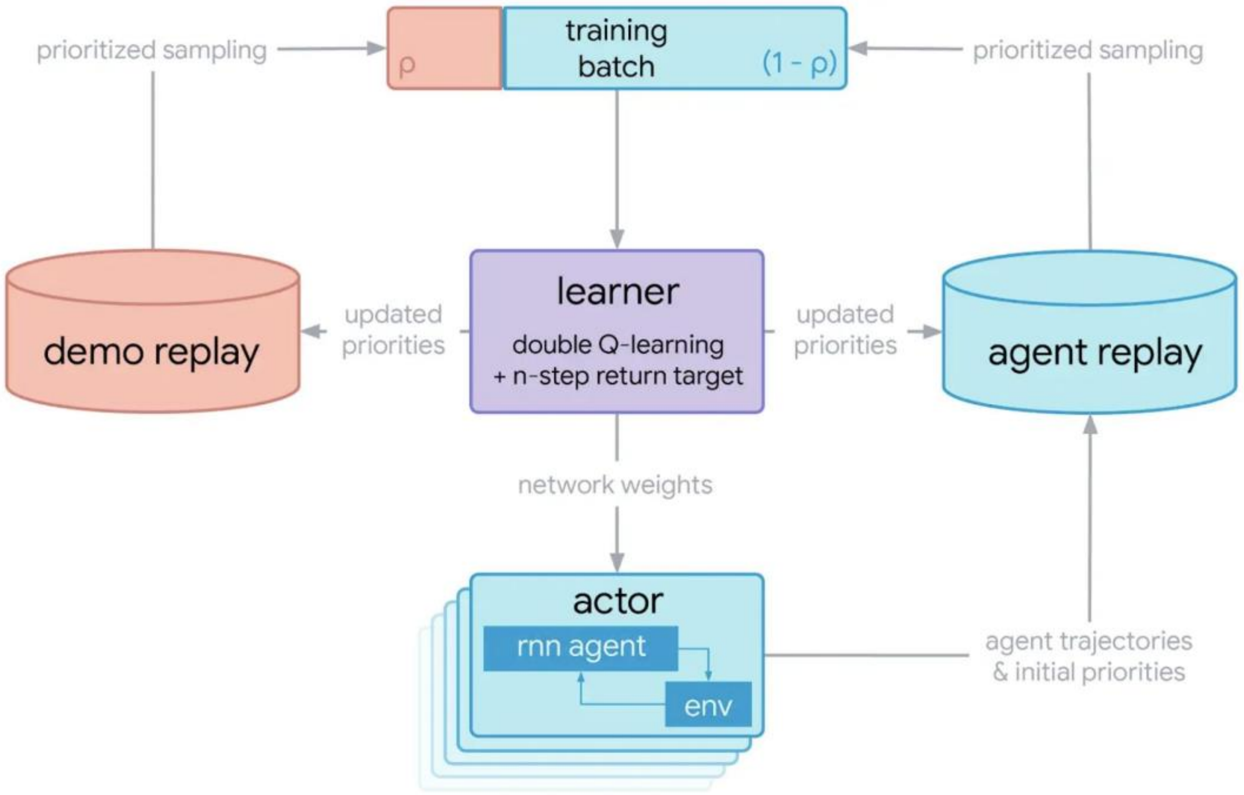
Scalable RL – Seeking solution in distributed computing

Ape-X/R2D2 with Off-Policy Learning



Scalable RL – Seeking solution in distributed computing

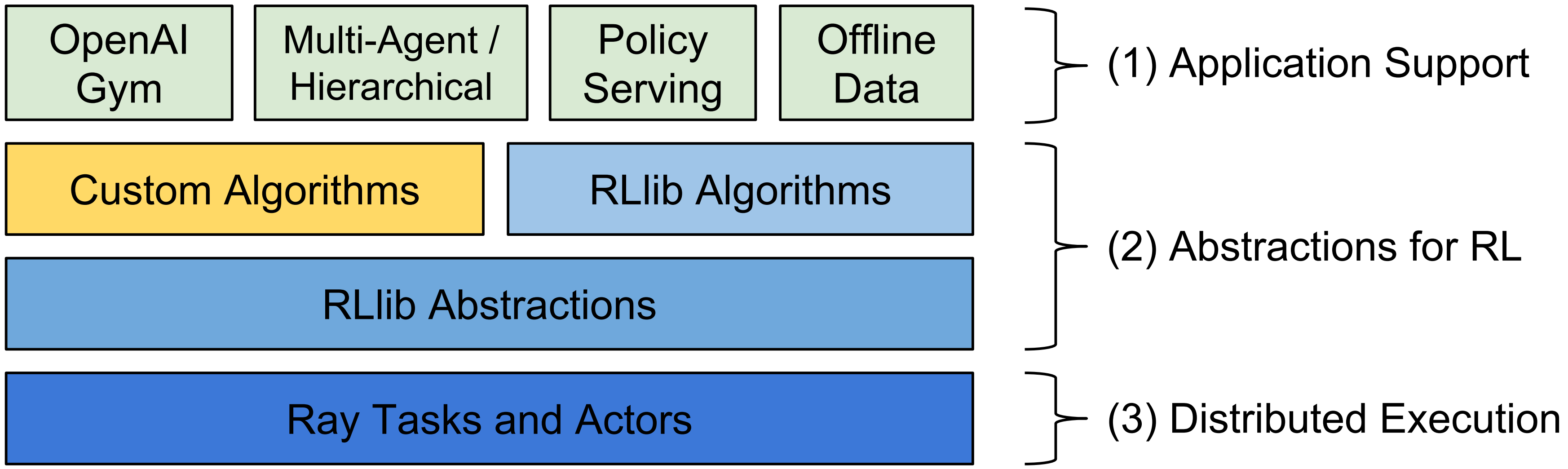
R2D3 with Imitation Learning



RLlib: Scalable Reinforcement Learning

- ✓ Builds on Ray to provide higher-level RL abstractions
- ✓ Hierarchical parallel task model with stateful workers
  - ✓ Flexible enough abstraction to capture a broad range of RL workloads

http://rllib.io



Seminars and Exams

SSSA Student Seminars

- ✓Student seminar lectures (n.2) in early October as a conclusion of this course
  - ✓15 minutes presentation
  - ✓5 minutes Q&A on the content of the presentation
- ✓ Seminar content
  - ✓Read 3 relevant papers on a topic of interest for the course; summarize their content and confront the methods
  - ✓Implement 1 RL method from literature and attempt a validation on a simple application; describe the model, its implementation and the validation results
- ✓Seminar ideas: additional lectures on Moodle & research topics in this lectures
- ✓Everybody welcome to attend!

Ph.D. Students

- ✓ Read 3 (or more) relevant papers on a topic of interest for the course and summarize their content in a report (6-10 pages single column, NeurIPS format)
- ✓ Sketch/propose a novel RL method/application: report your idea in sufficient detail in a short paper (6-10 pages single column, NeurIPS format)
- ✓ Implement a RL-based application and validate it: prepare a short presentation to report the results (10-15 slides describing the model, the implementation and the results)
- ✓ Contact me and agree on alternative ways (e.g. using RL in your Ph.D. project)
- ✓ No presentation required, only delivery of material
- ✓ Please complete within the year 2020

Final Wrap

Conclusions

- ✓Many thanks for being part of this highly experimental course
- ✓Hopefully the total accumulated rewards have not be too painful and you have enjoyed the course
- ✓Stay tuned on Moodle for the seminars in October