Introduction to Reinforcement Learning

June 27th, 2023

Nick Zolman (UW)
Ludger Paehler (TUM)
Vincent Van Wynendaele (ESPCI-Paris PSL)

dynamicsai-rl-tutorial Public

Pin | Unwatch 2

main | 1 branch | 0 tags | Go to file | Add file | Code

nzolman fixed typo in pip install — cfff383 now — 13 commits

.gitignore	Initial commit	last week
LICENSE	Initial commit	last week
README.md	typo	4 hours ago
dynai.png	updated README	9 hours ago
full_tutorial.ipynb	fixed typo in pip install	now
template.ipynb	fixed typo in pip install	now

README.md

Reinforcement Learning Tutorial

This repository hosts the code for the reinforcment learning tutorial for the AI Institute in Dynamic Systems ML Workshop on Tuesday, June 27th, 2023.

Contents

https://tinyurl.com/dynamicsai-2023

Today's Tutorial

- High-Level Introduction to RL
- Examples
- Building RL Intuition
- The "Sharp Edges" of DRL
- Practical Code Walkthrough

RL: "Learning Through Interaction"

Agent

RL: “Learning Through Interaction”

Agent

Observation

“Sit”

RL: “Learning Through Interaction”

Action

Sit

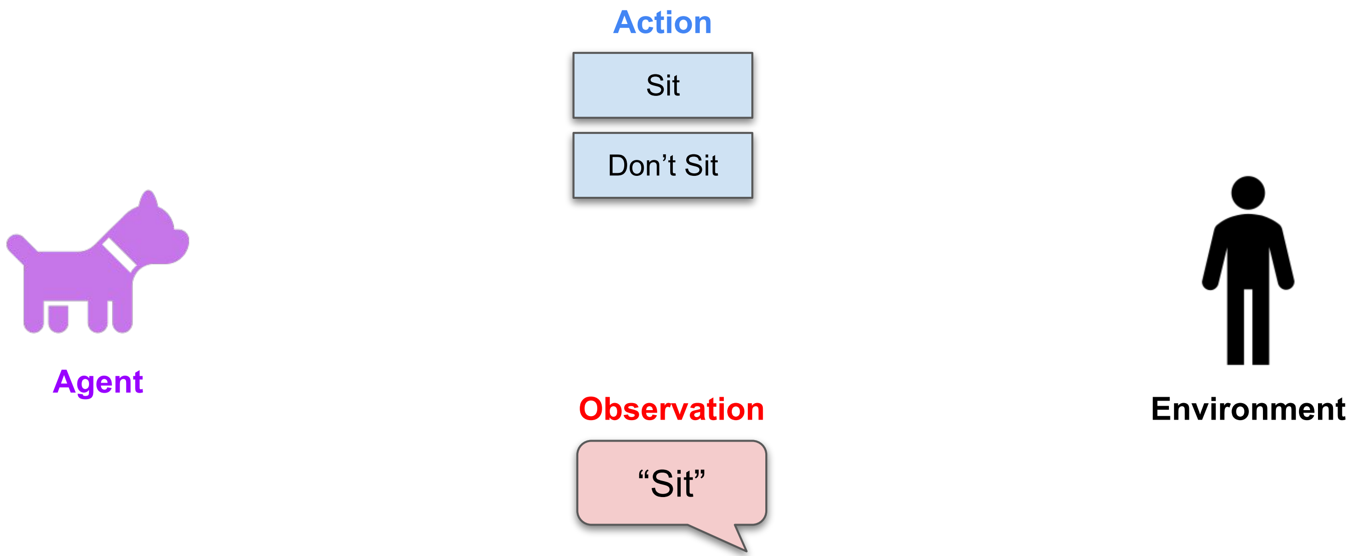
Don’t Sit

Agent

Observation

“Sit”

RL: “Learning Through Interaction”



RL: “Learning Through Interaction”

Agent

Action

Sit

Don’t Sit

Observation

“Sit”

Environment

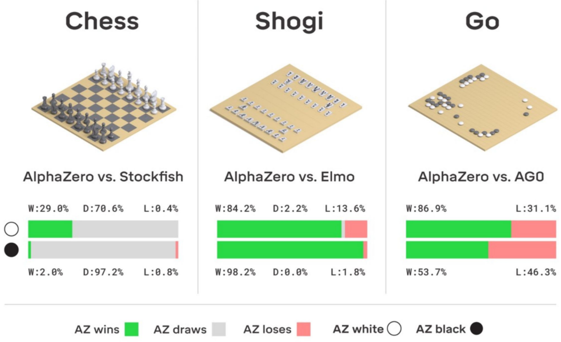
Reward

RL: “Learning Through Interaction”

Today's Tutorial

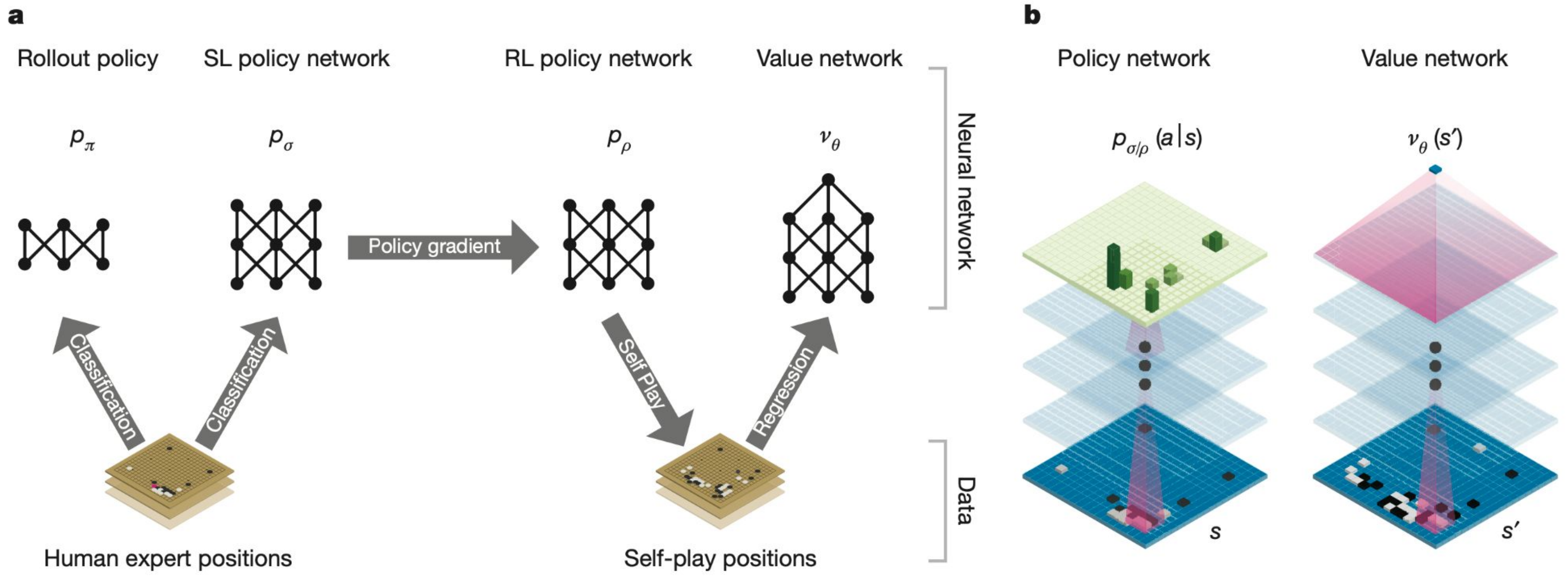
- High-Level Introduction to RL
- **Examples**
- Building RL Intuition
- The "Sharp Edges" of DRL
- Practical Code Walkthrough

The Deepmind Revolution



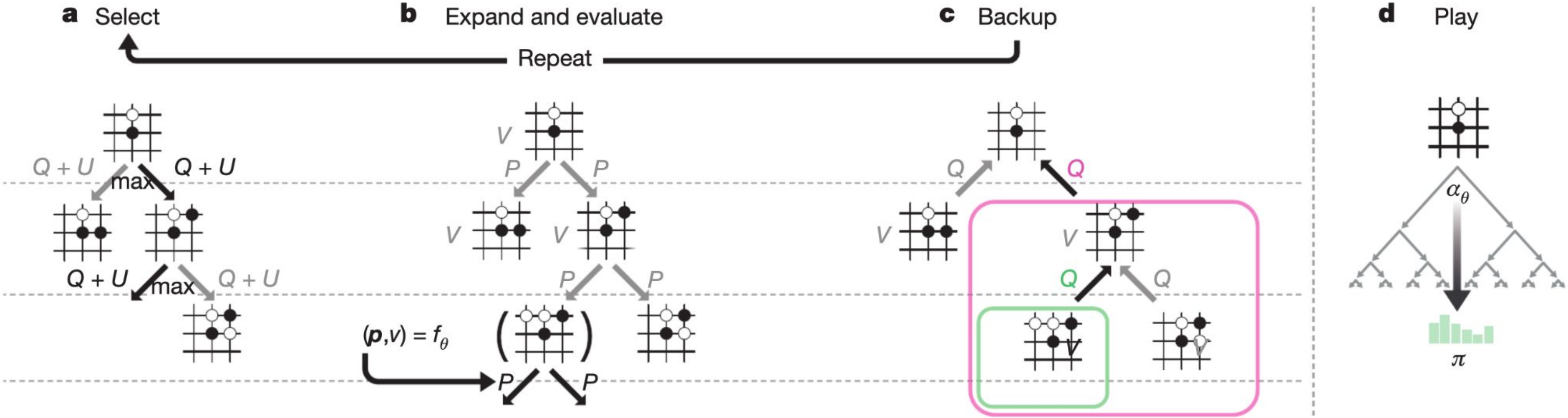
AlphaGo Lee (2016) → AlphaGo Master (2017) → AlphaGo Zero (2017)

The Deepmind Revolution: Alpha Go



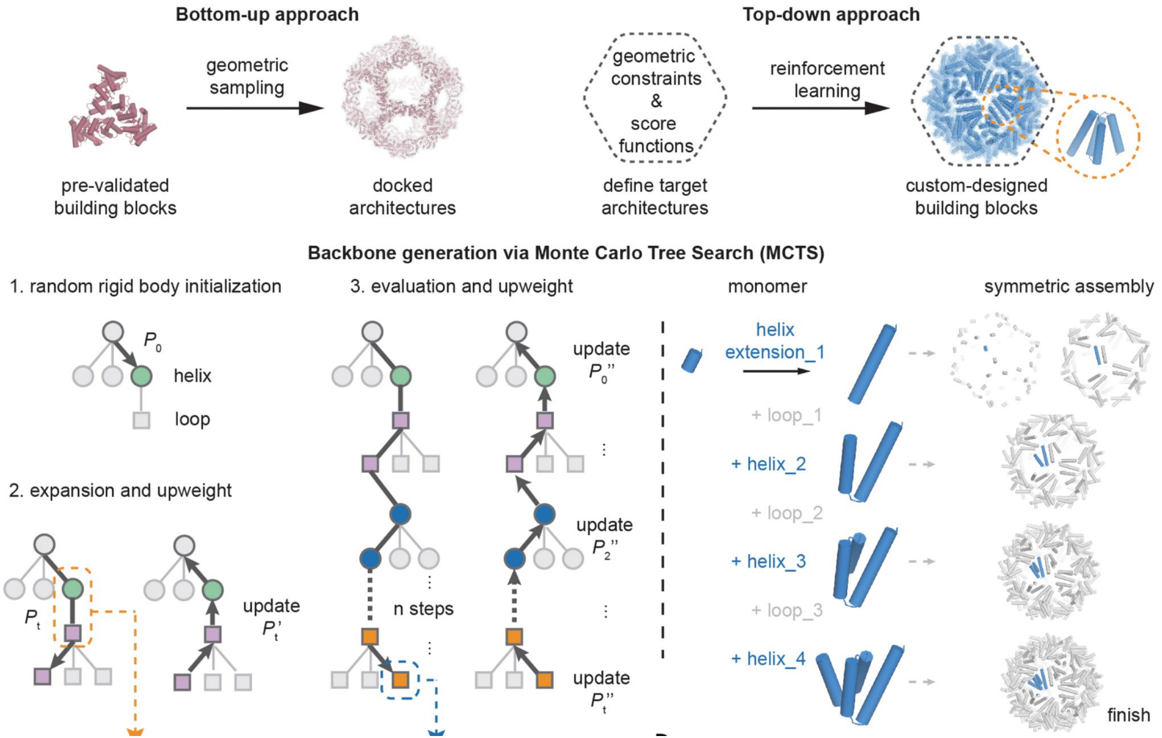
Human Expert Knowledge + Monte Carlo Tree Search

The Deepmind Revolution: Alpha Zero



Monte Carlo Tree Search

Design of Protein Architectures

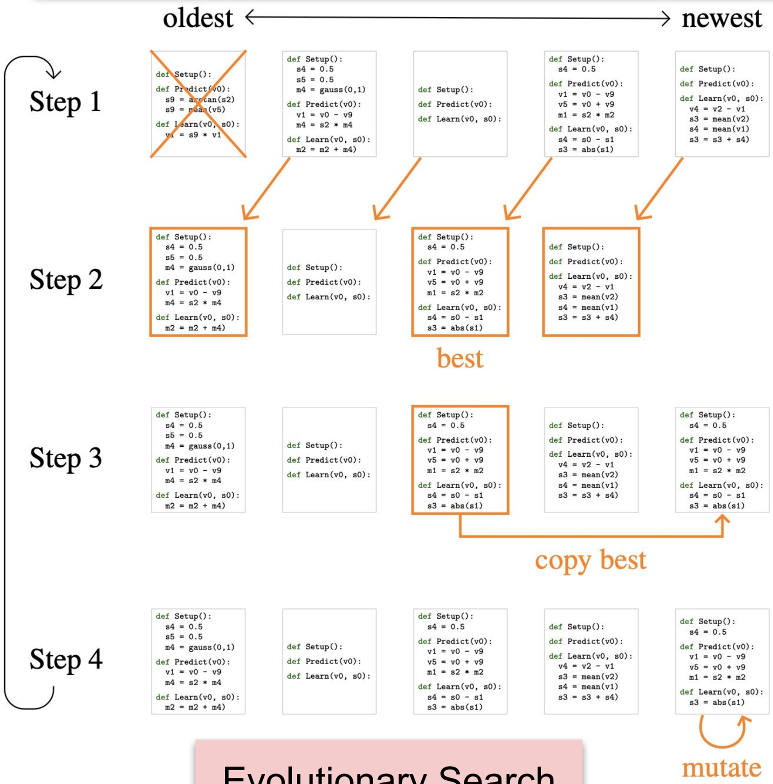


- Gamified design of proteins with the same search algorithm as DeepMind
- >1m molecules as starting blocks

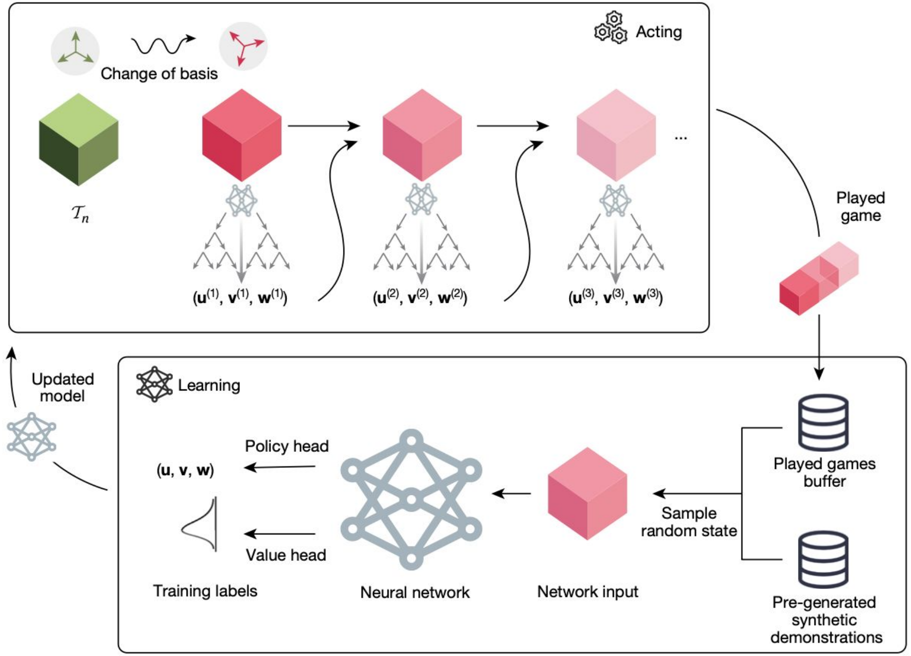
>100 RL-generated molecules manufactured
Under electron microscopes

"Top-down design of protein architectures with reinforcement learning" by Lutz et al.

Evolution of Simulations: AutoML Zero & AlphaTensor

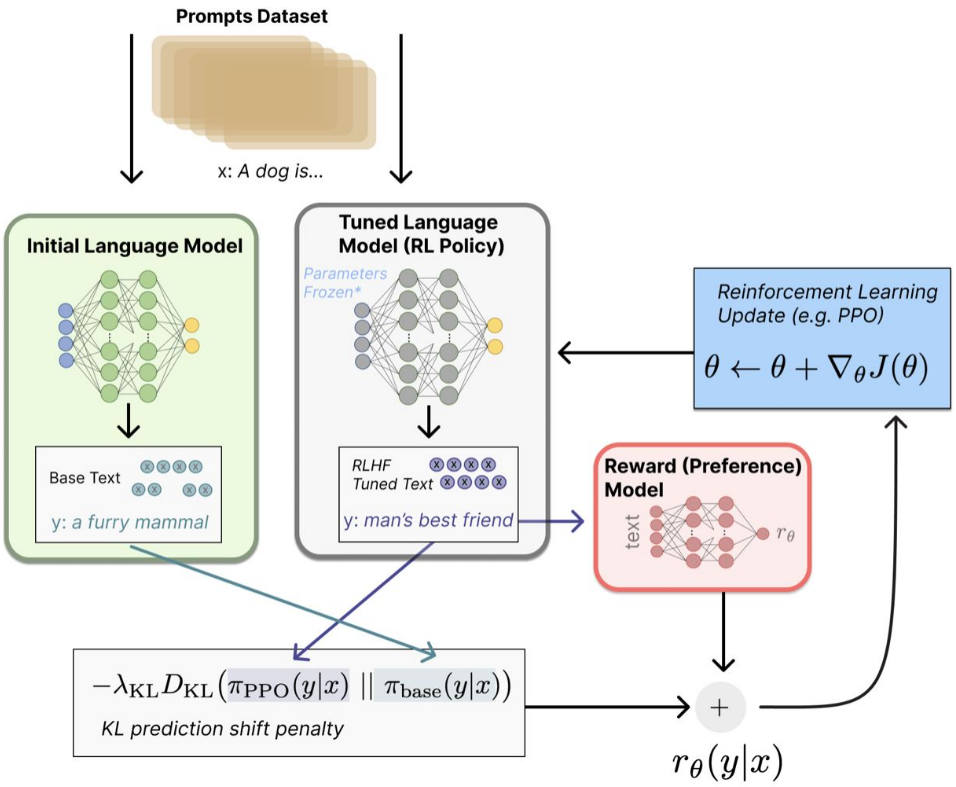


Evolutionary Search



Alpha Zero for Tensor Decompositions

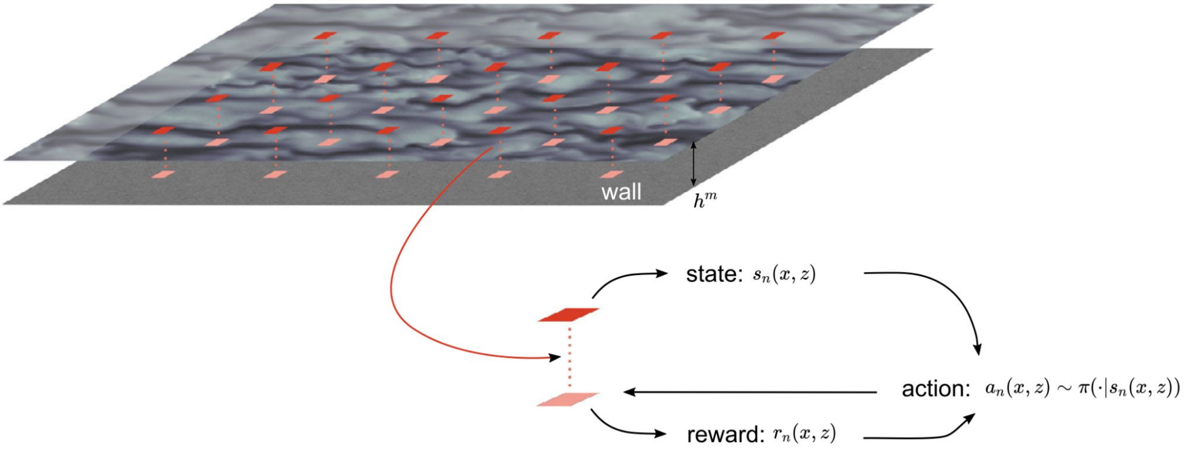
Large Language Model Revolution: RLHF



- Requires a pre-trained large language model
- Fine-tunes LLM with policy gradient method
- Fine-tuning on human preferences
  - *Instruction Tuning*

Requires **Human**-generated Instruction Datasets

Wall Models for Large-Eddy Simulations

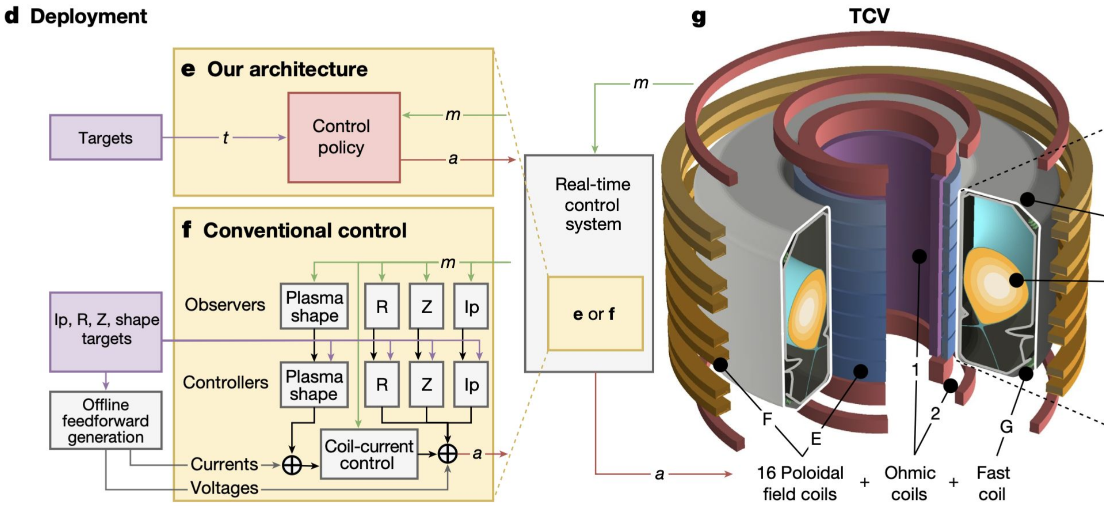


- Large-scale multi-agent RL with equi-spaced agents along the wall to learn the wall-model of an LES
- Trained on the mean wall-shear stress

Tested on Turbulent boundary layers

"Scientific multi-agent reinforcement learning for wall-models of turbulent flows" by Bae and Koumoutsakos

Control of a Tokamak Reactor



1. Physicists defines control objectives
2. Deep RL interacts with Tokamak **simulator**
3. Control policy applied to **Tokamak reactor**

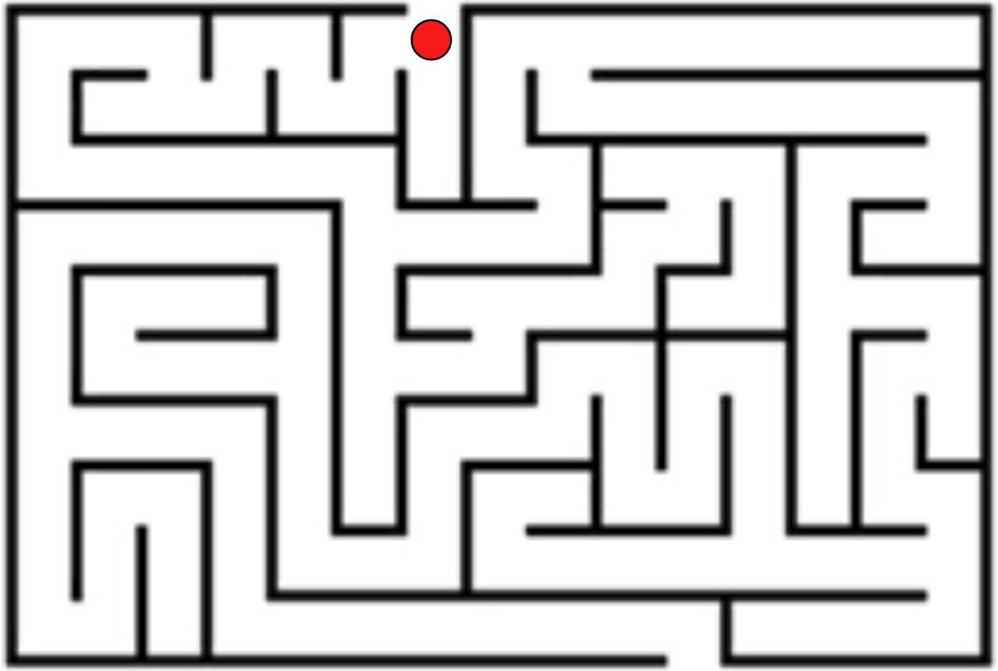
RL Control Policy run
On real Tokamak reactor **<u>in practice!</u>**

"Magnetic Control of Tokamak Plasmas through Deep Reinforcement Learning", Degrave et al.

Today’s Tutorial

- High-Level Introduction to RL
- Examples
- **Building RL Intuition**
- The “Sharp Edges” of DRL
- Practical Code Walkthrough

Breaking Down Reinforcement Learning

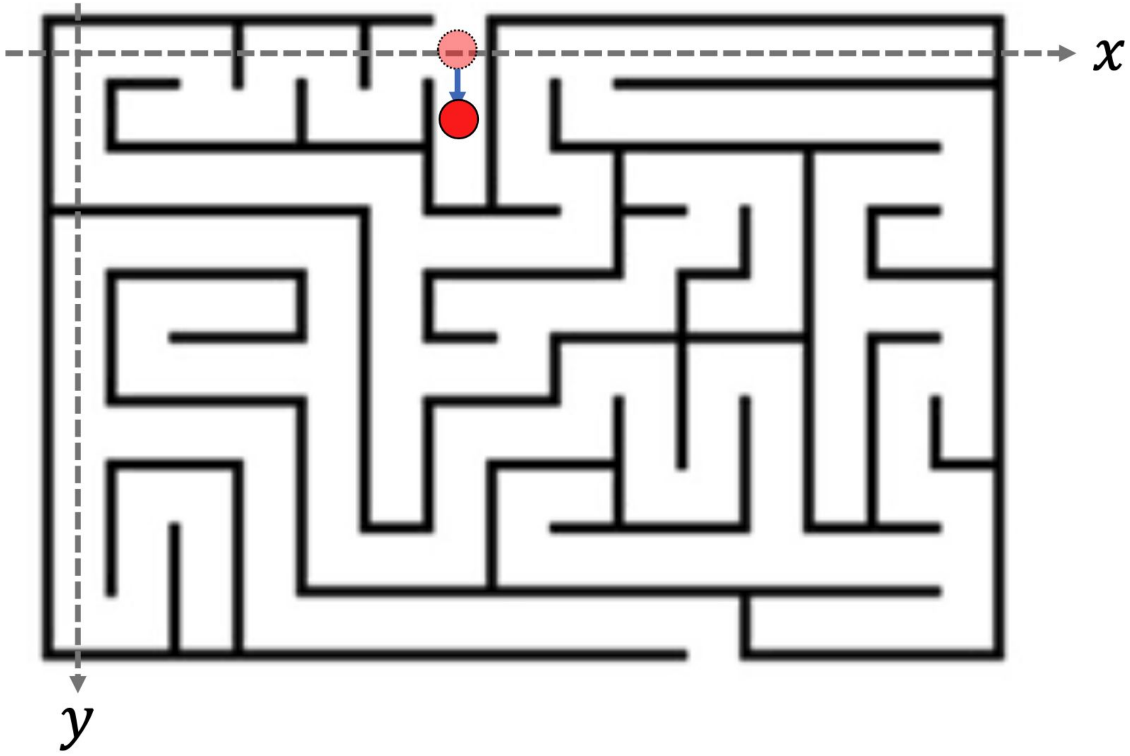


State: Where you are

Action: left/right/up/down

Next state: Where you are after the action

Breaking Down Reinforcement Learning



State: s = (x, y) = (6,0)

Actions:

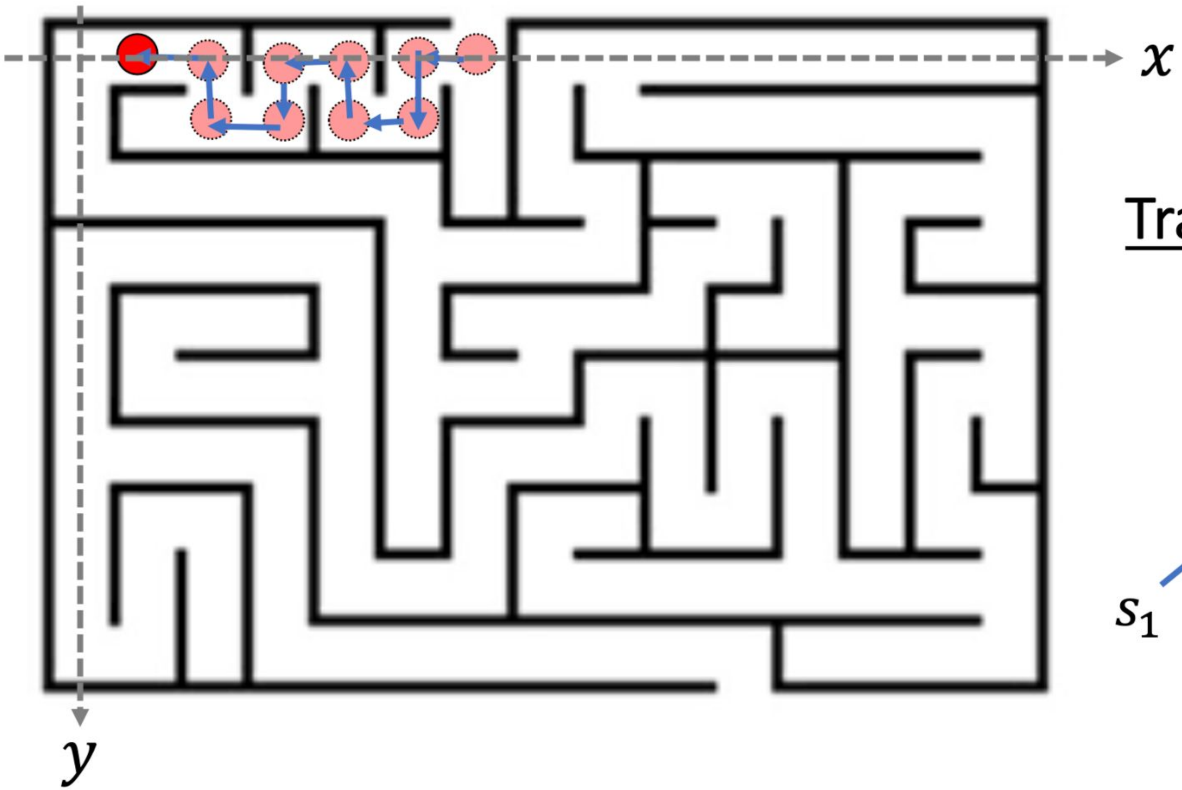
Left: x ← x - 1

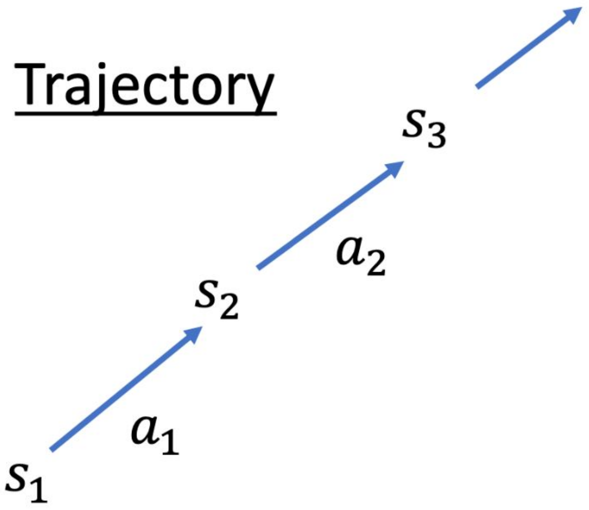
Right: x ← x + 1

Up: y ← y - 1

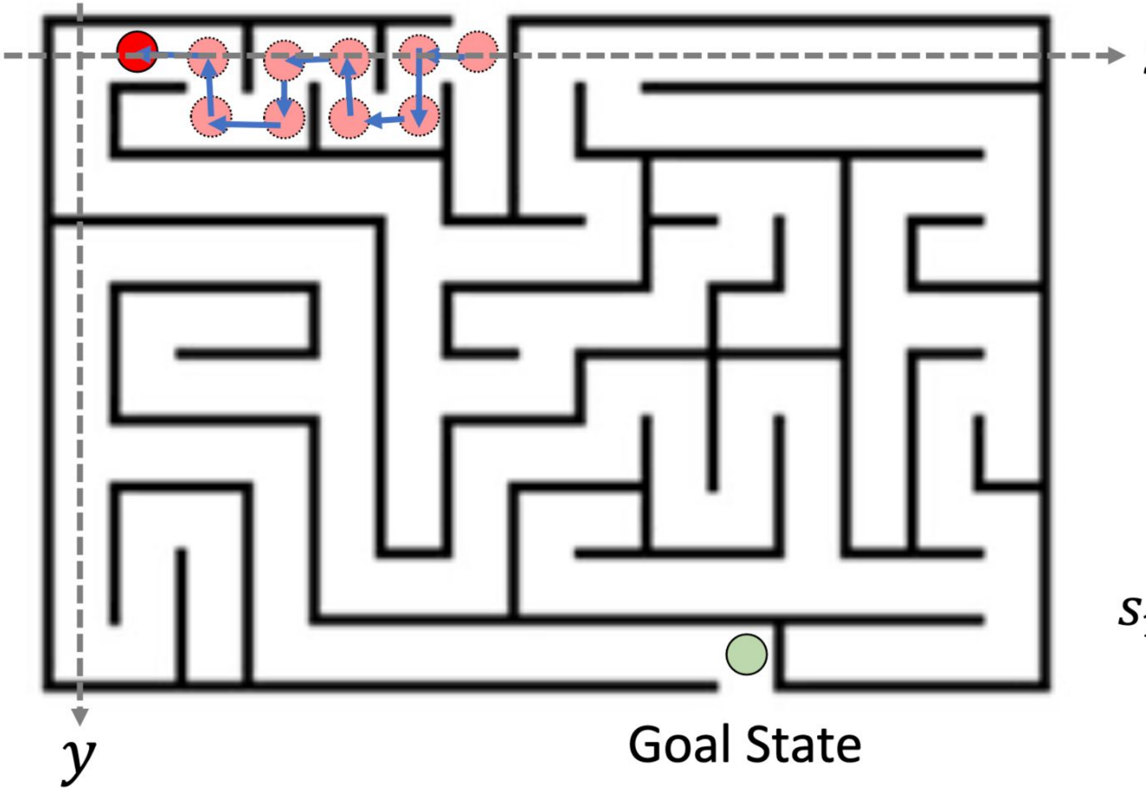
Down: y ← y + 1

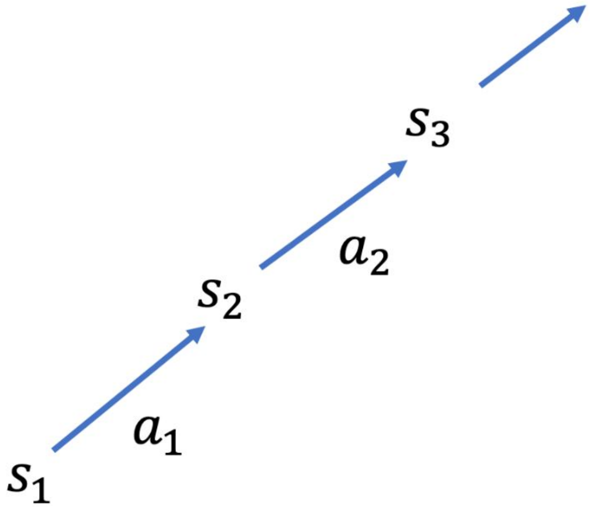
Breaking Down Reinforcement Learning



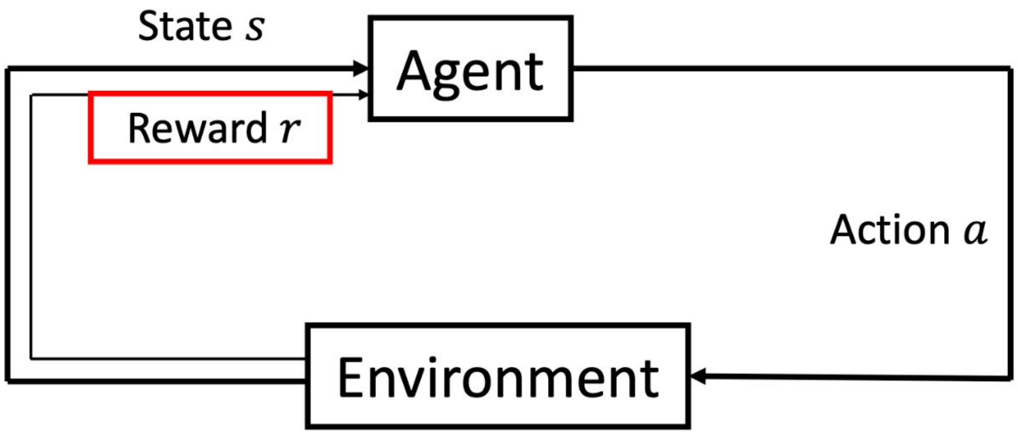


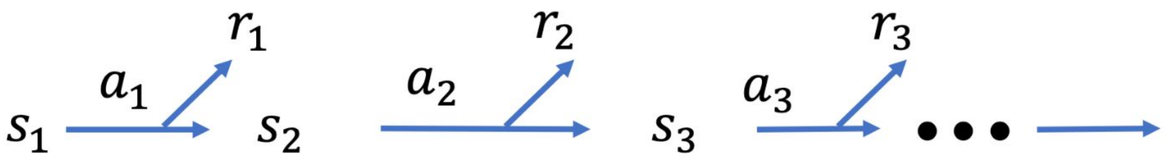
Breaking Down Reinforcement Learning





The Goal of Reinforcement Learning



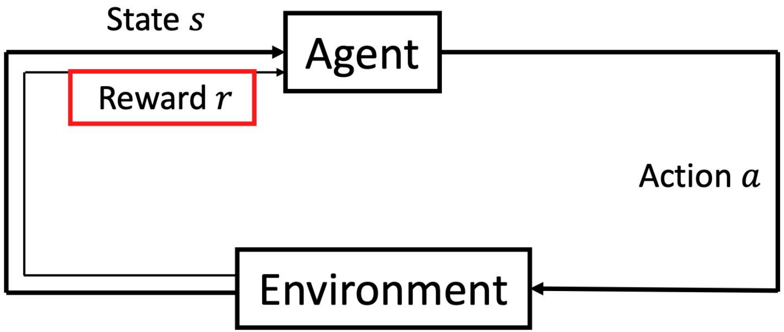


γ: Discount factor

Maximize long-term reward:

$$R = \sum_{t=1}^{+\infty} \gamma^{t-1} r_t$$

Key Quantities

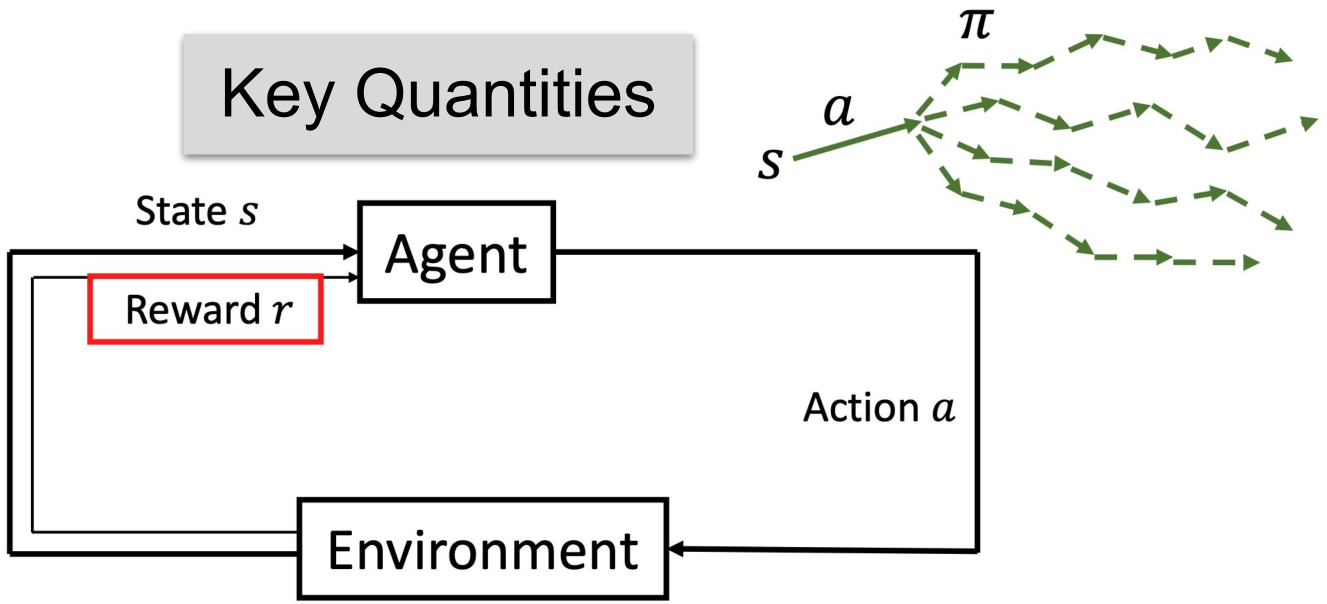


V*(s): Maximal reward you can get starting from state s

Q*(s, a): Maximal reward starting from s after taking action a

π(a|s): Probability of taking action a given state s

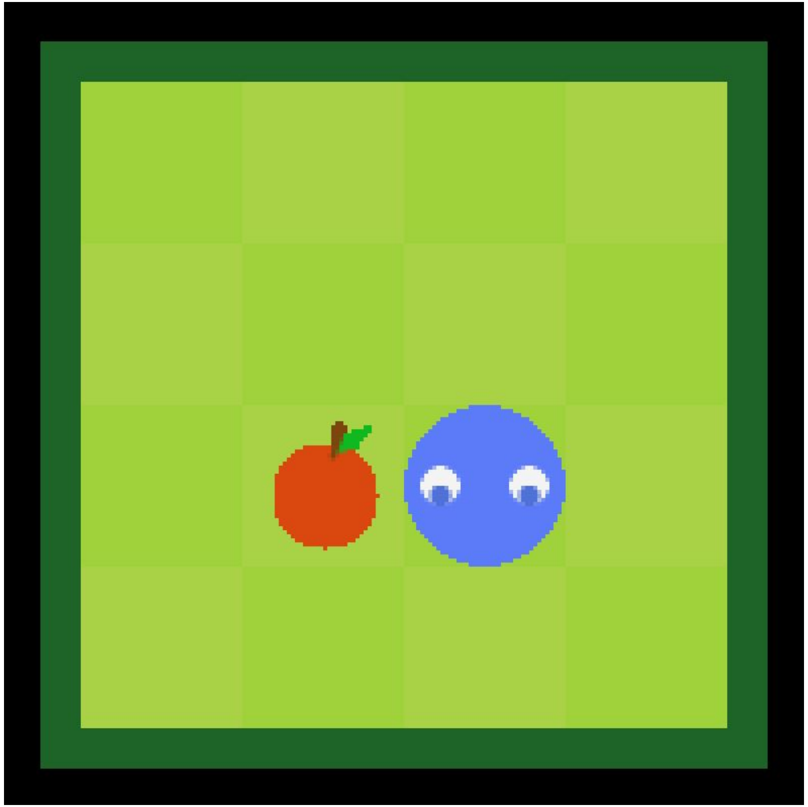
Key Quantities

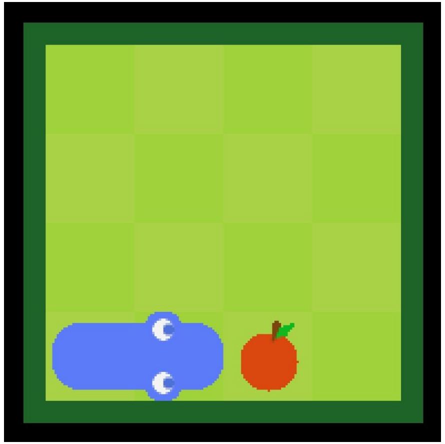


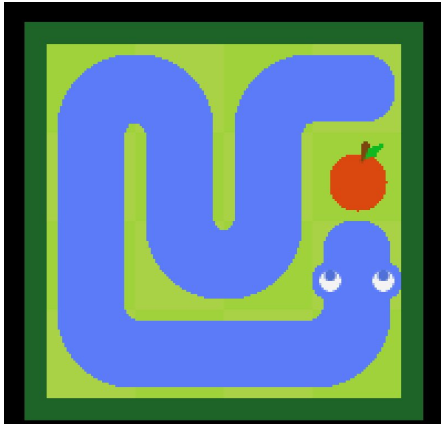
$V^{\pi}(s)$: Reward you can get, starting from s <u>following policy</u> π

$Q^{\pi}(s, a)$: Reward starting from s after taking action a <u>following</u> π

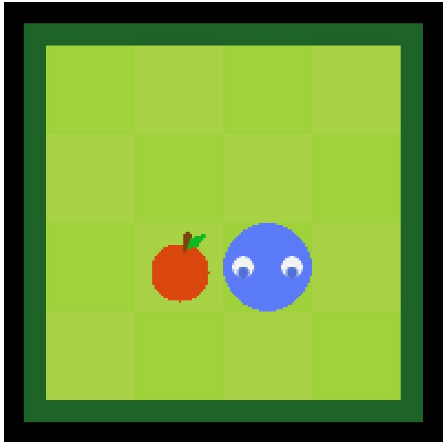
The Snake Game







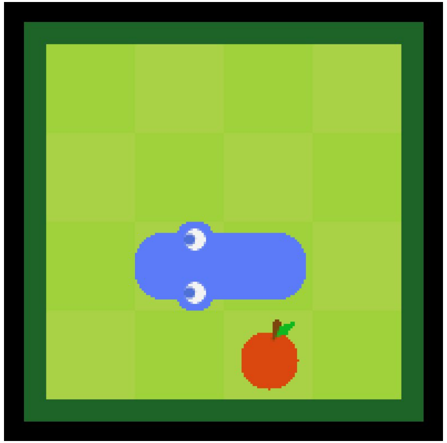
Quick Description of the environment



S_t

$$A_t = left$$

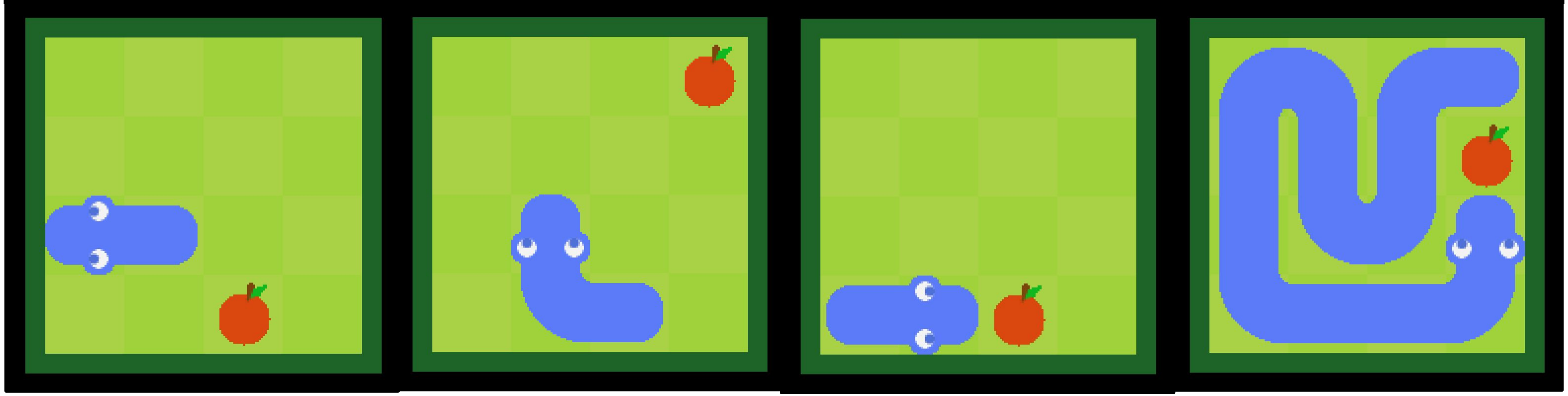
$$R_{t+1} = apple\_value$$



S_{t+1}

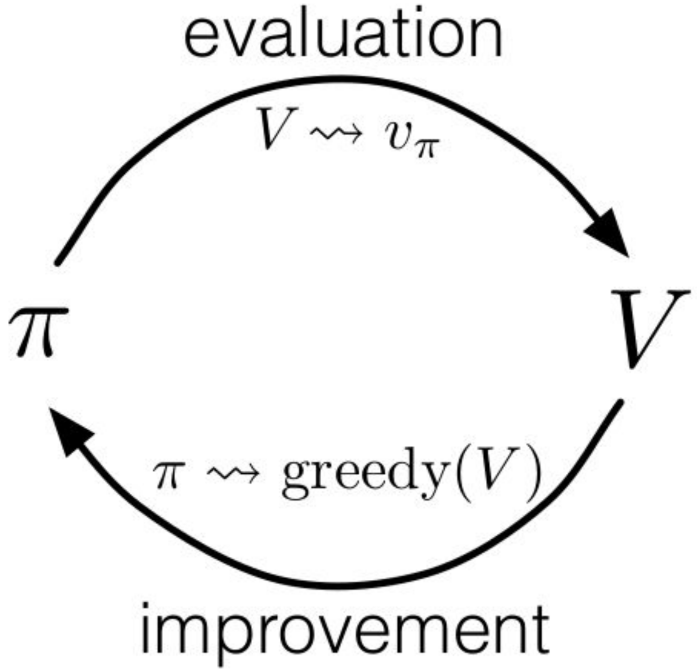
Finishing the environment : Shaping the reward function

Lose	Simple move	Get apple	Win



$A_t = left$	$A_t = left$	$A_t = right$	$A_t = up$

How unintended consequence happen ?

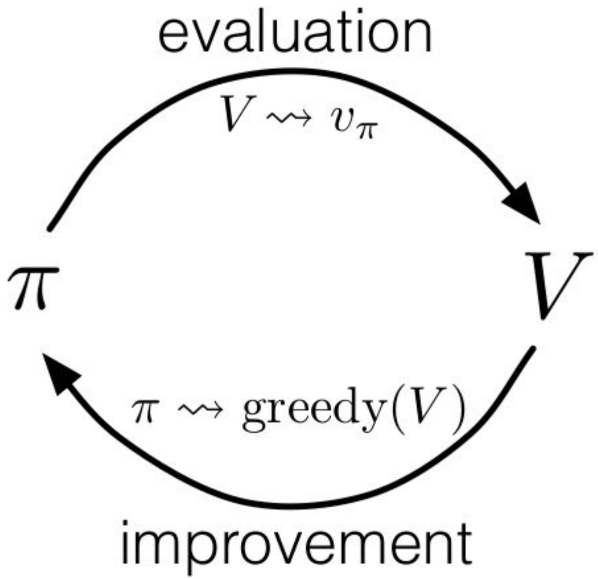


Evaluate $Q^\pi(s, a)$

Improve by taking

$\pi(s) = \text{argmax}_a\, Q^\pi(s, a)$

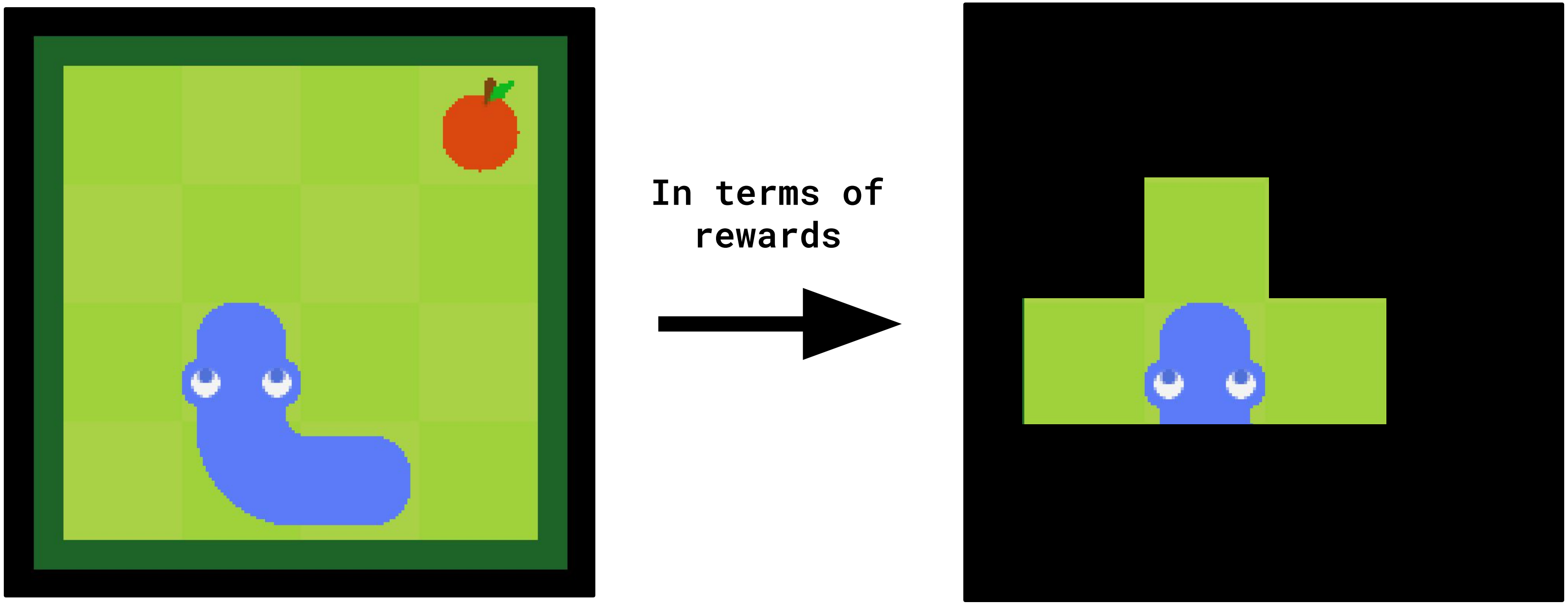
The importance of hyper-parameters: γ and ε



$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+1+k}$$

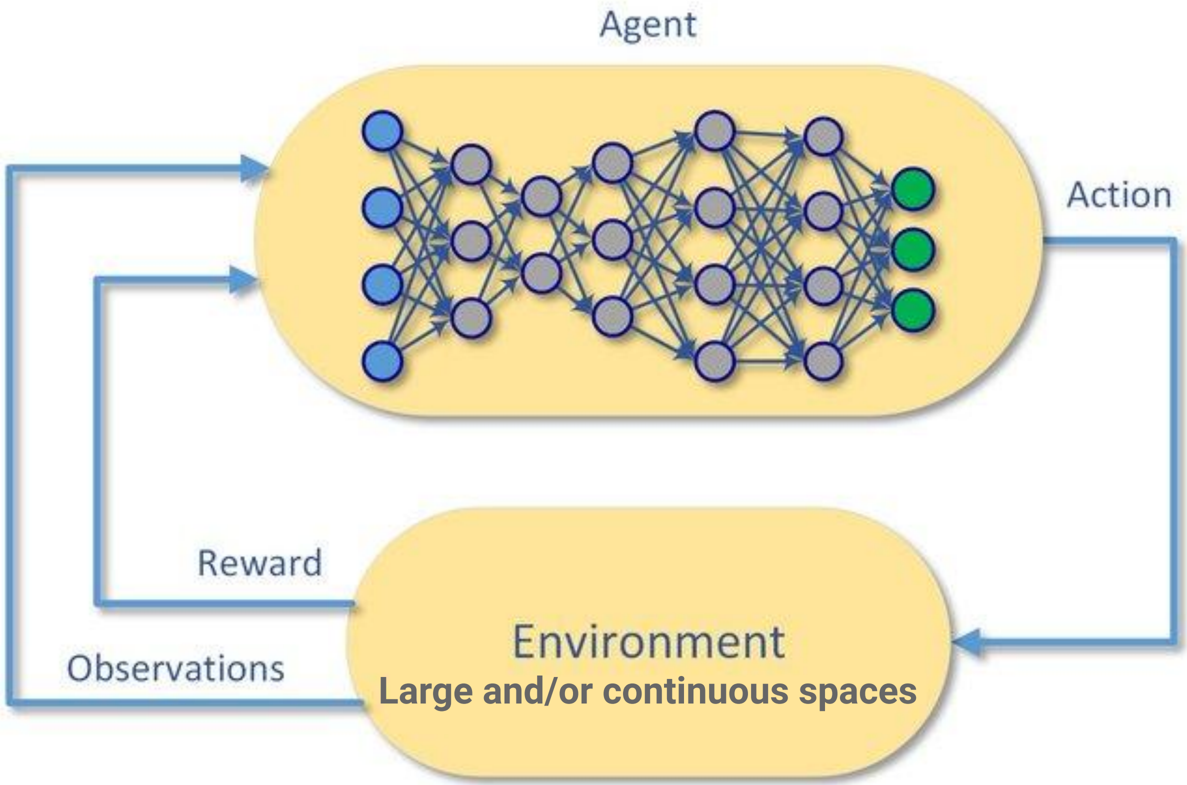
When should you explore ?
The ε-greedy setting

Visualization of γ = 0



$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+1+k} = R_{t+1}$$

What about Deep learning



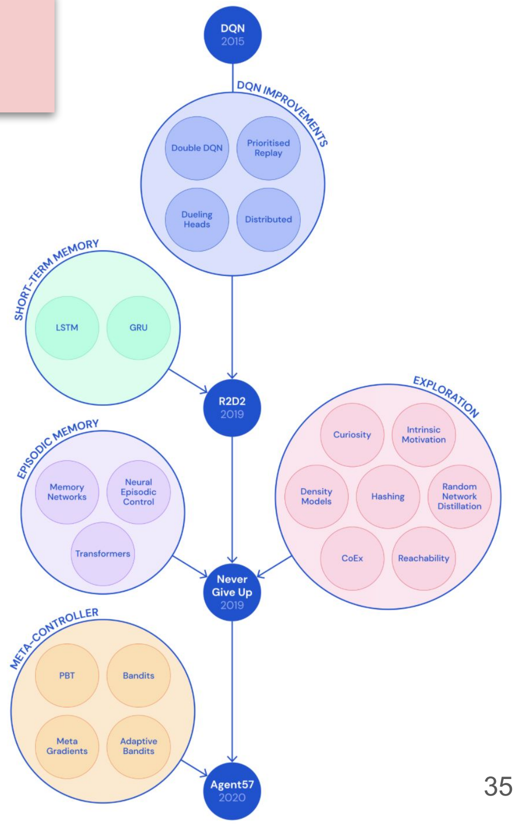
Today’s Tutorial

- High-Level Introduction to RL
- Examples
- Building RL Intuition
- **The “Sharp Edges” of DRL**
- Practical Code Walkthrough

Sharp Edges of Deep RL: Reward Shaping

Exploration vs Exploitation

- Under sparse rewards, random exploration fails
- Strategies -> Augment the reward with…
    - A bonus signal
    - A prediction model which measures curiosity of the RL agent
    - A (variational) forward model
    - A Physical properties
    - An external memory
        - Short-term memory
        - Episodic memory
    - Direct exploration



Sharp Edges of Deep RL: Inverse RL

Problem Definition:

- Inputs: (state-, action-space, sample trajectories, dynamics model)
- Outputs: **reward function**, from which we then seek to the recover **policy**

Learning rewards from goals, demos

- Practical framework for task specification
- Adversarial training can be unstable
- Requires examples of desired behavior or outcomes

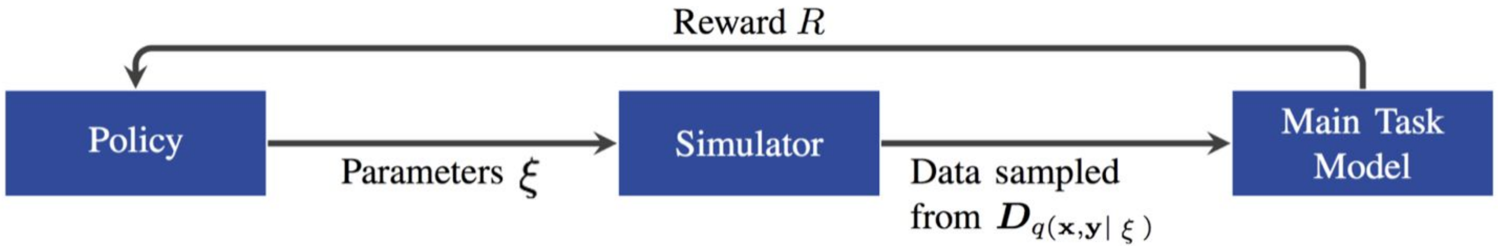
Learning rewards from human preferences

- Pairwise preferences easy to provide
- Has been deployed at scale
- May require (human) supervision in the loop of RL

Sharp Edges of Deep RL: Domain Transfer

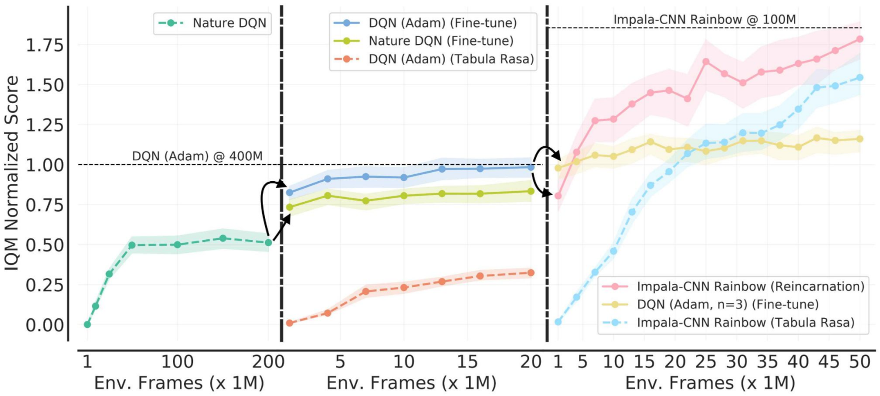
- Deep RL is sample-inefficient + data collection is expensive
- Number of available approaches
  - System identification (supervised)
  - Domain adaptation (unsupervised)
  - Domain randomization (unsupervised)

Toolkit highly dependent on simulator, and associated costs

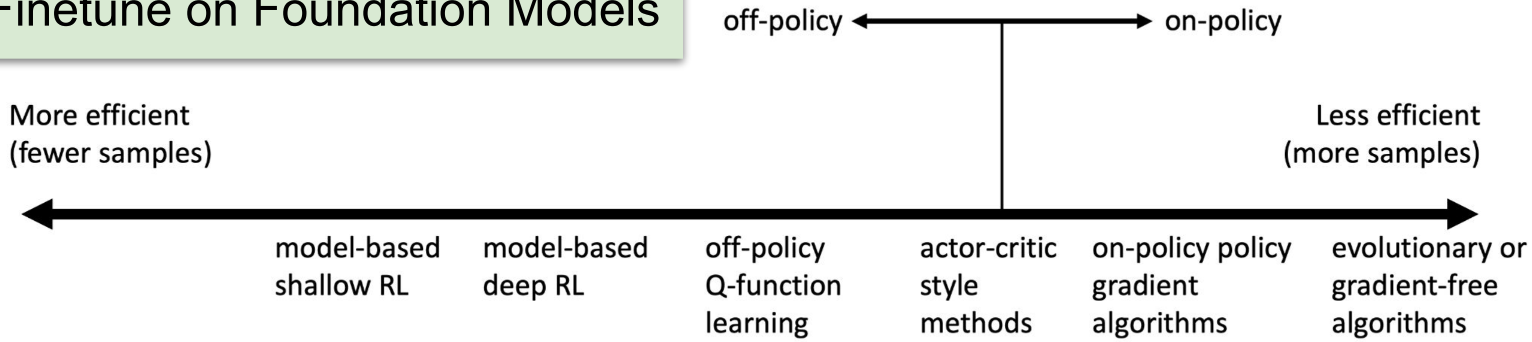


Sharp Edges of Deep RL: Cost of Data Collection

- Cost of data collection very high for scientific environments
    - >100m interactions with simulator=💣
- Ability to utilize **surrogates** such as Gaussian Processes etc.
- Highly active area of research

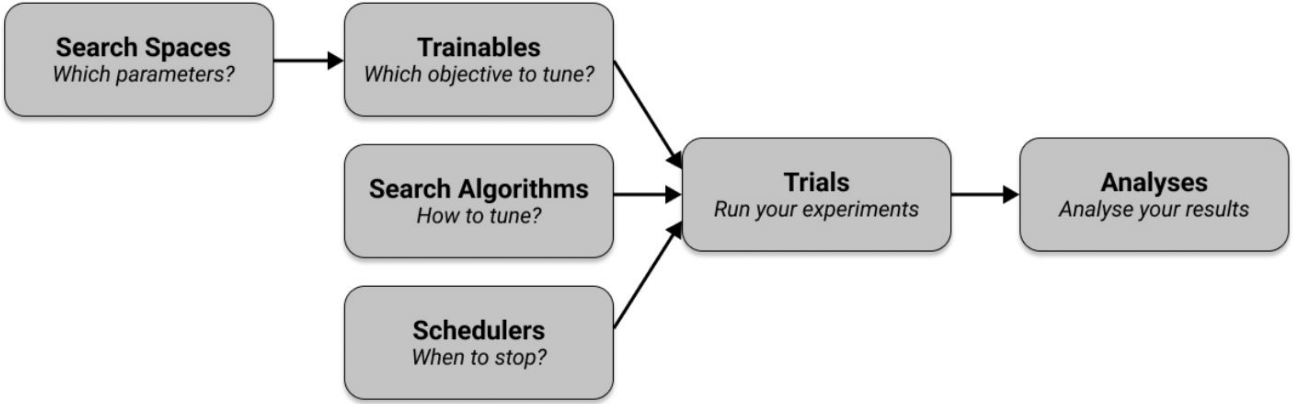


Finetune on Foundation Models



Sharp Edges of Deep RL: Hyperparameter Tuning

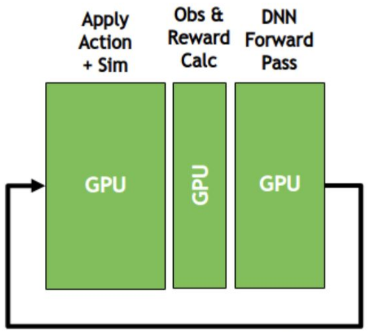
- Performance of (deep) RL highly dependent on hyperparameters
- Number of Hyperparameter tuning approaches available
  - Bayesian optimizationm
  - Population-based training
- Most stable with a Hyperparameter Tuning Toolkit like **Ray Tune**

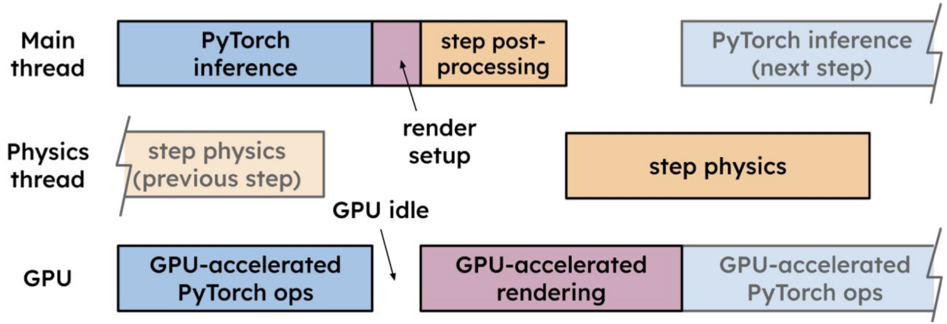


Sharp Edges of Deep RL: Environment Construction

- We desire: Efficient, low-latency implementation of the environment
- Techniques:
  - Separation of compute between CPUs & GPUs with latency-masking
  - Agent + environment compiled into one executable and run on the GPU/TPU
  - Vectorized gymnasium environments
  - Operation of own Threadpool with own Queue

Ray + Ray Runtime give us most of these





Practical Code Walkthrough

https://tinyurl.com/dynamicsai-2023