SUMMIT
ONLINE

INT08

Unlabelled data and the rise of reinforcement learning

Ben Thurgood
Senior Manager, Solutions Architect
Amazon Web Services

Agenda

What's the big deal about RL?

How / why RL works

How to build an RL model (with minimum pain)

Demo

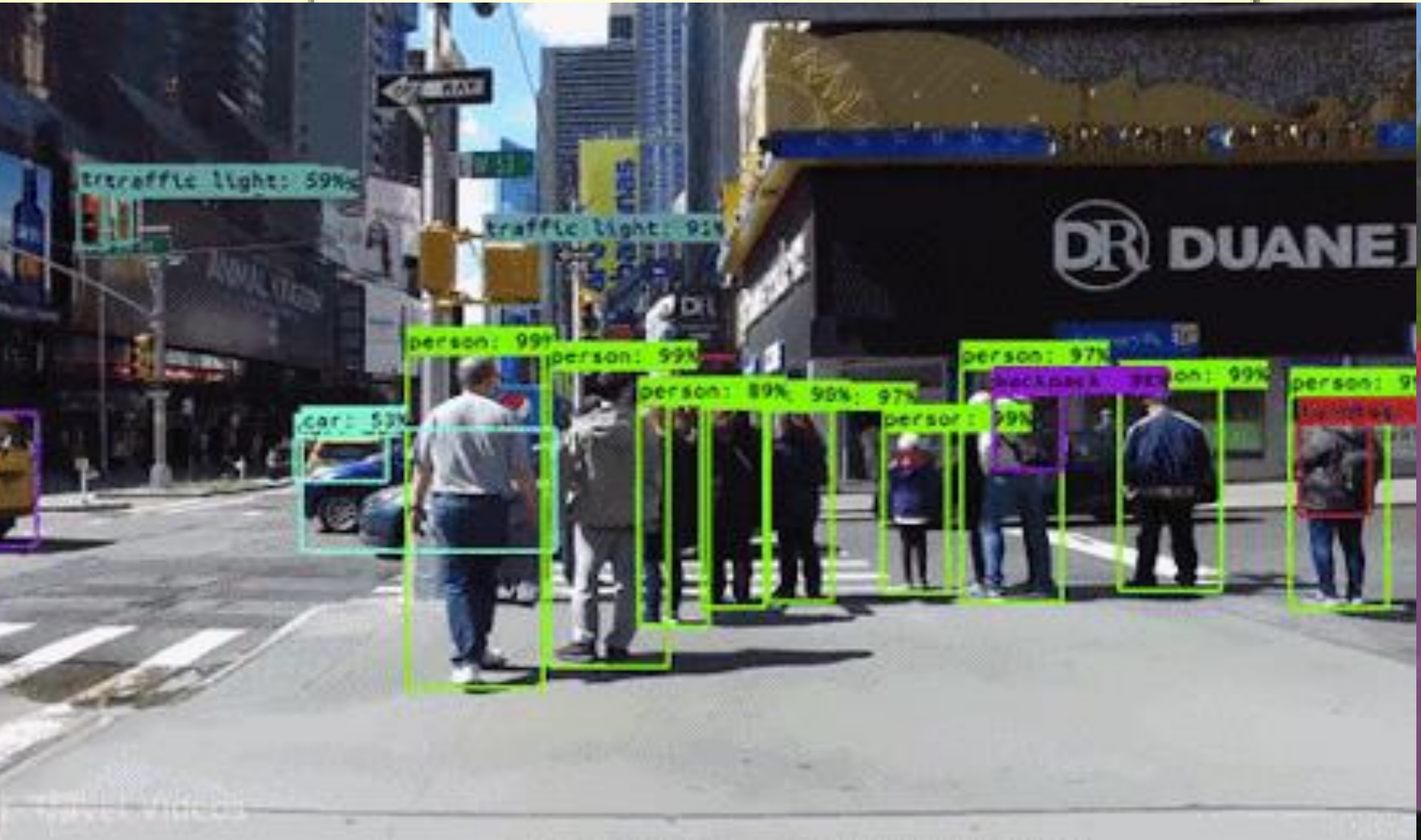
When to use RL? (and when not to)

Tips for success

What's next

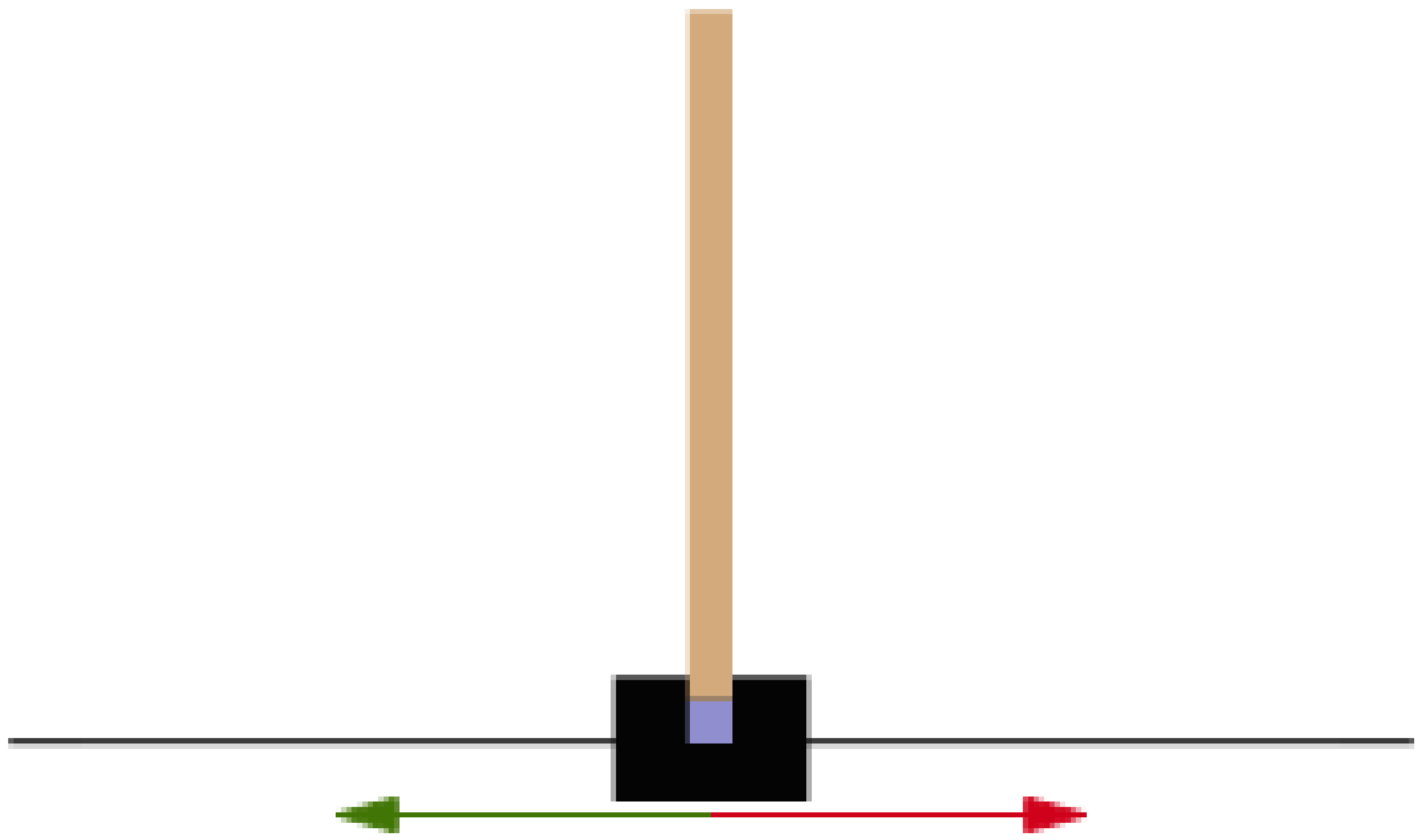
RL: What’s the big deal?

This is one of Crichton's best books. The characters of Karen Ross, Peter Elliot, Munro, and Amy are beautifully developed and their interactions are exciting, complex, and fast-paced throughout this impressive novel. And about 99.8 percent of that got lost in the film. Seriously, the screenplay AND the directing were horrendous and clearly done by people who could not fathom what was good about the novel. I can't fault the actors because frankly, they never had a chance to make this turkey live up to Crichton's original work. I know good novels, especially those with a science fiction edge, are hard to bring to the screen in a way that lives up to the original. But this may be the absolute worst disparity in quality between novel and screen adaptation ever. The book is really, really good. The movie is just dreadful.

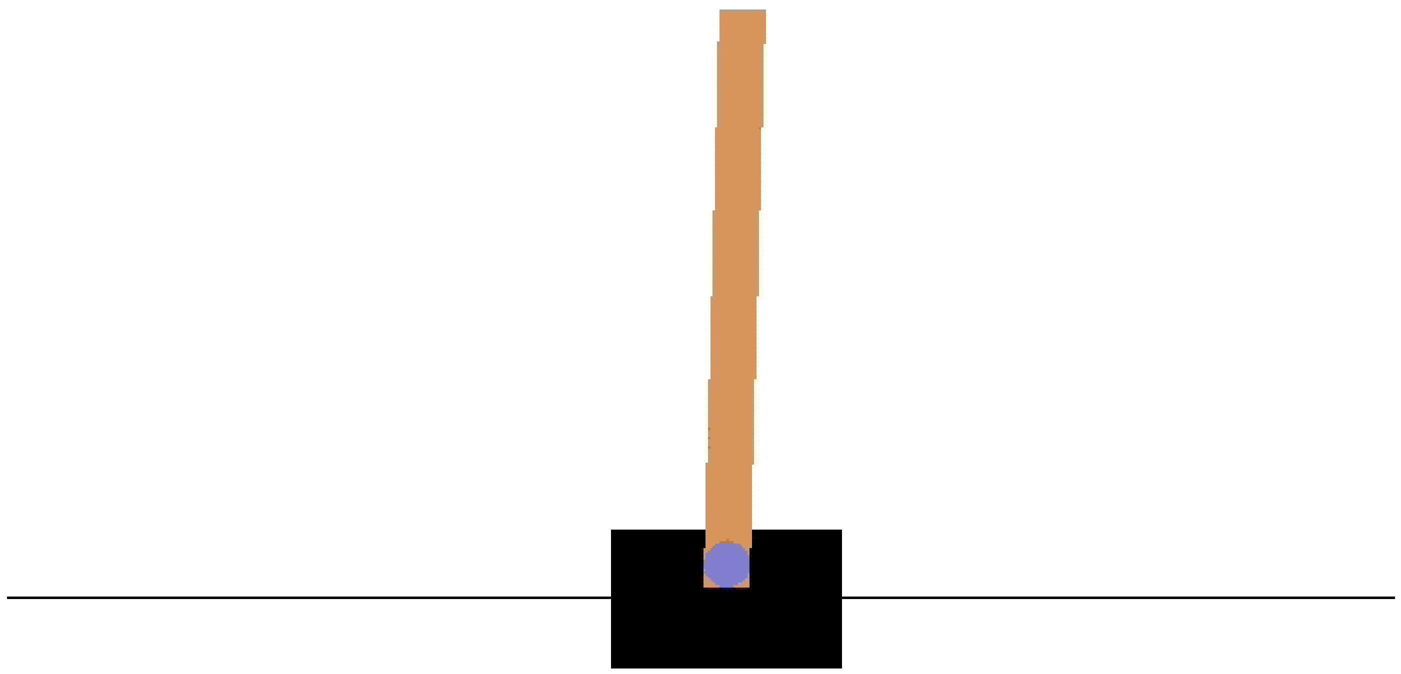


Successful models require high-quality data

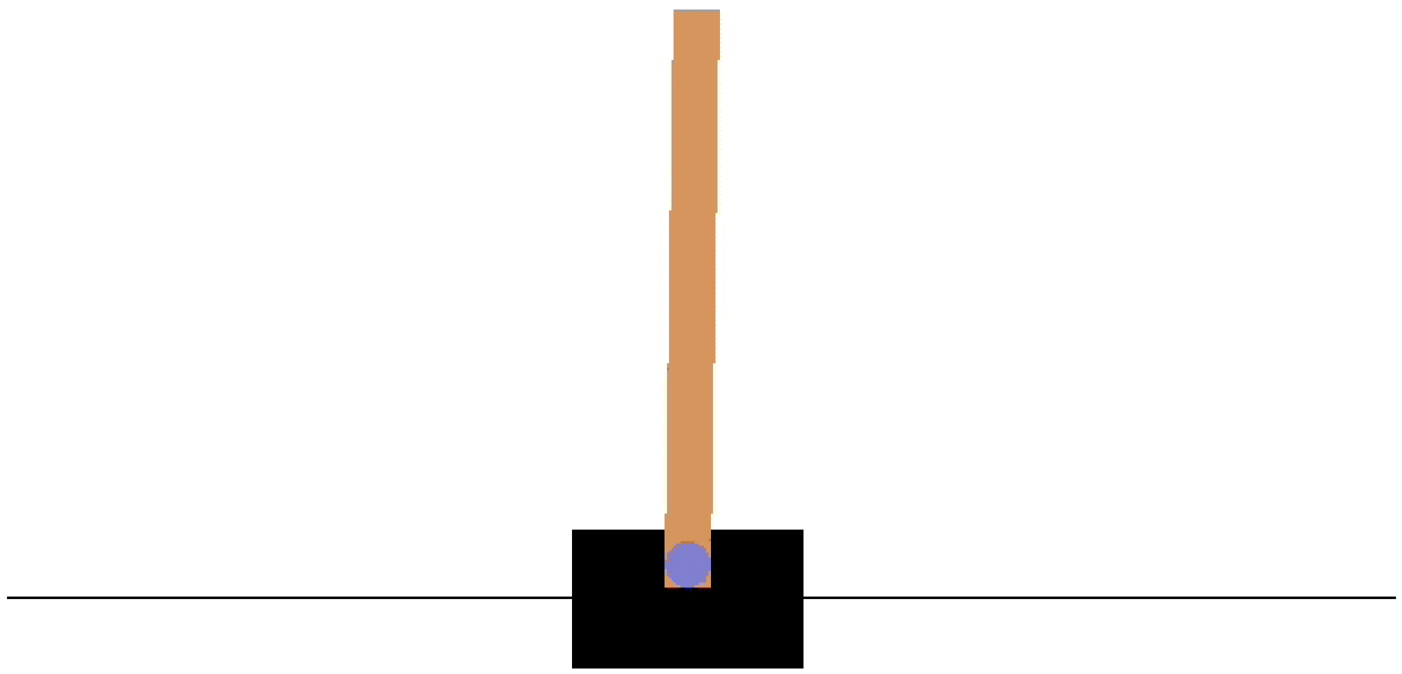
Balance a pole on a cart (Cartpole)



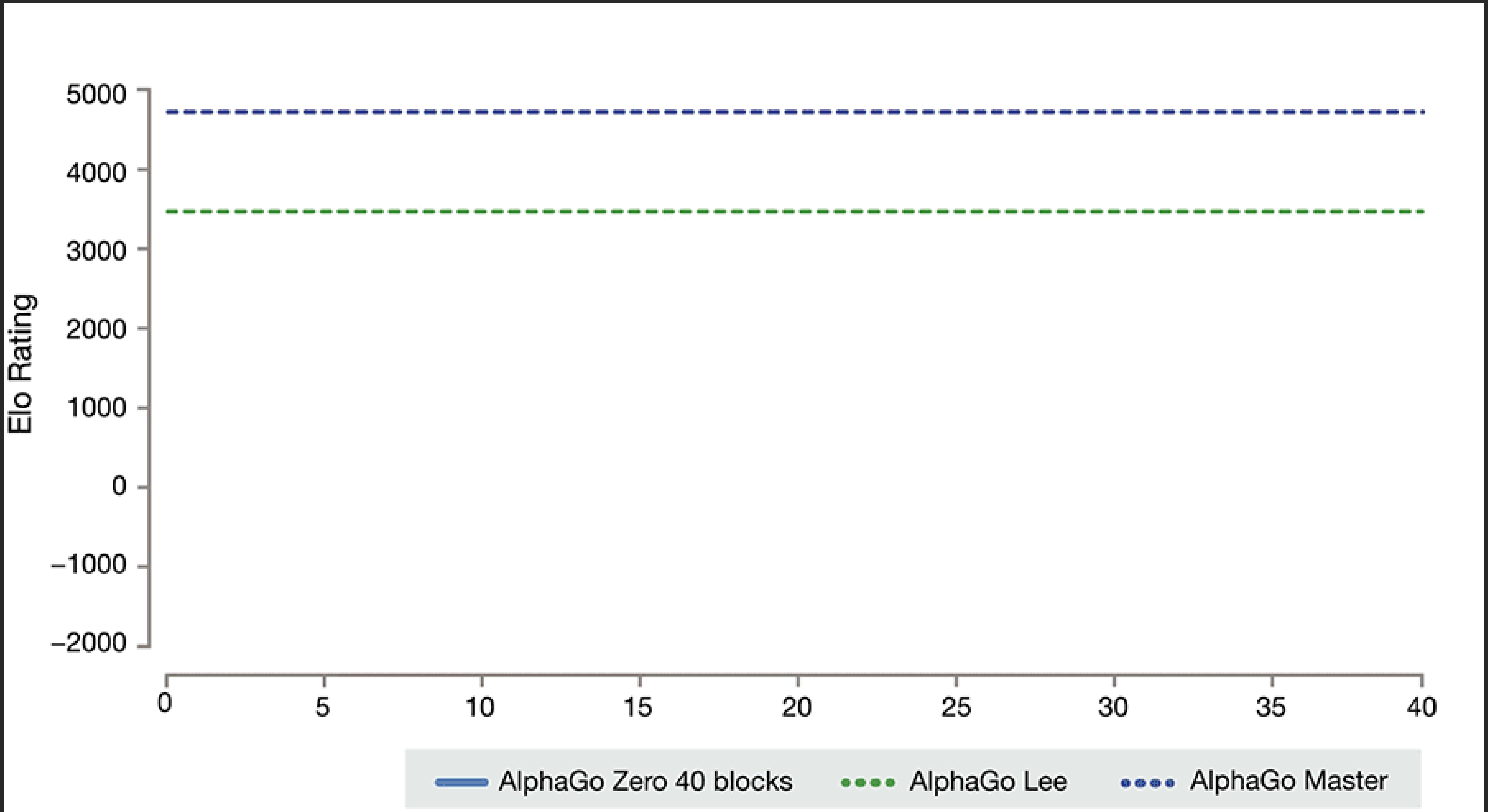
10 tries



200 tries

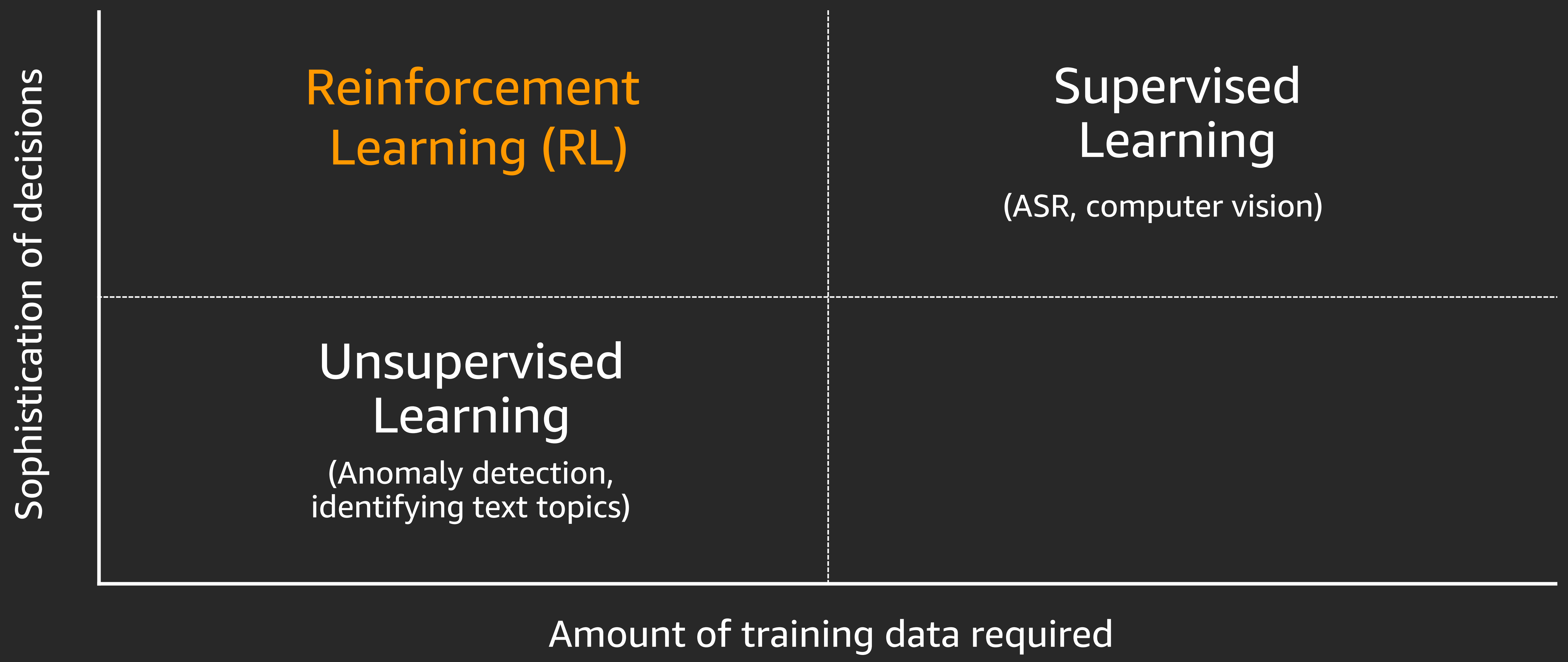


Going beyond mimicry



https://deepmind.com/blog/article/alphago-zero-starting-scratch

Machine learning approaches

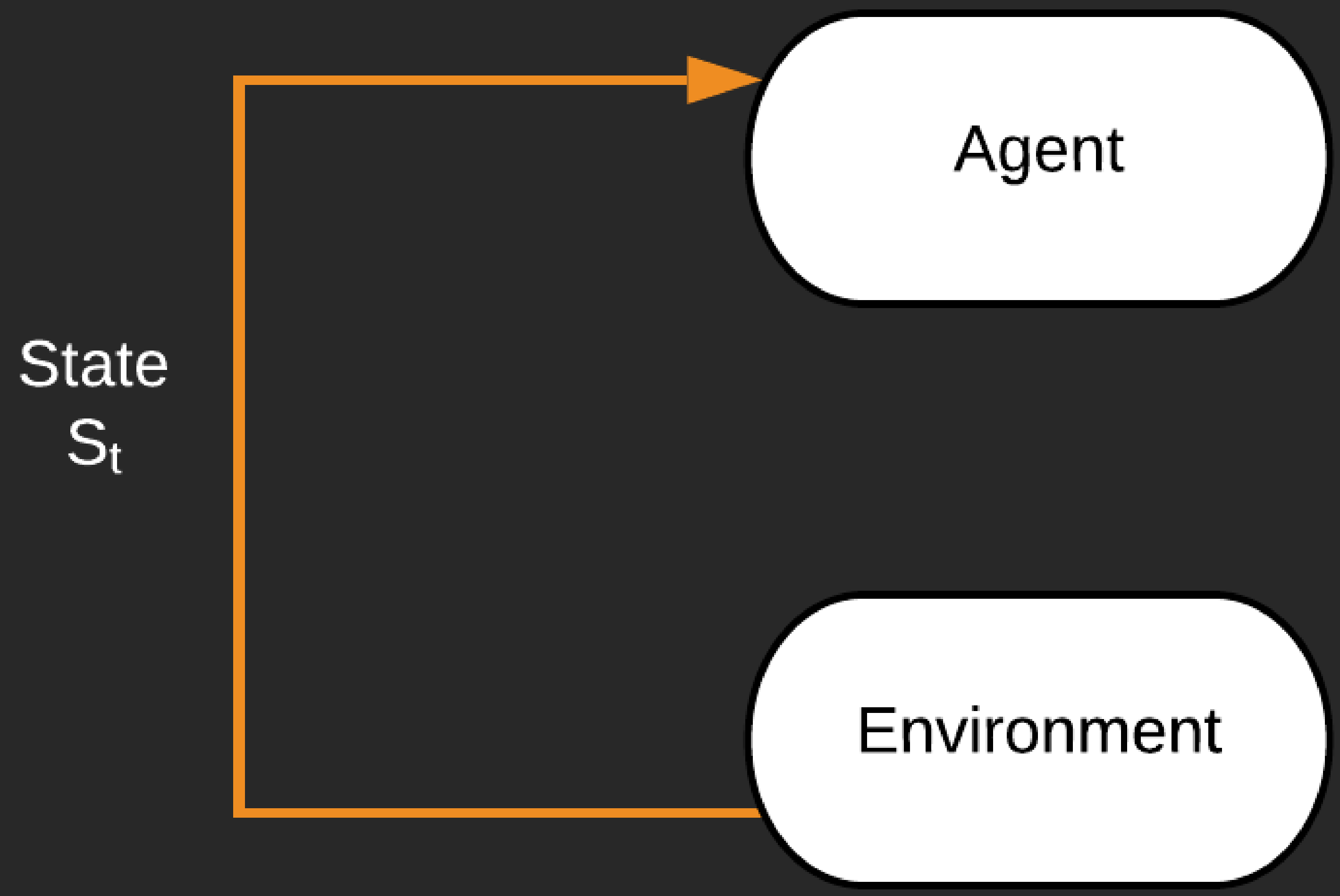


RL: How / why it works

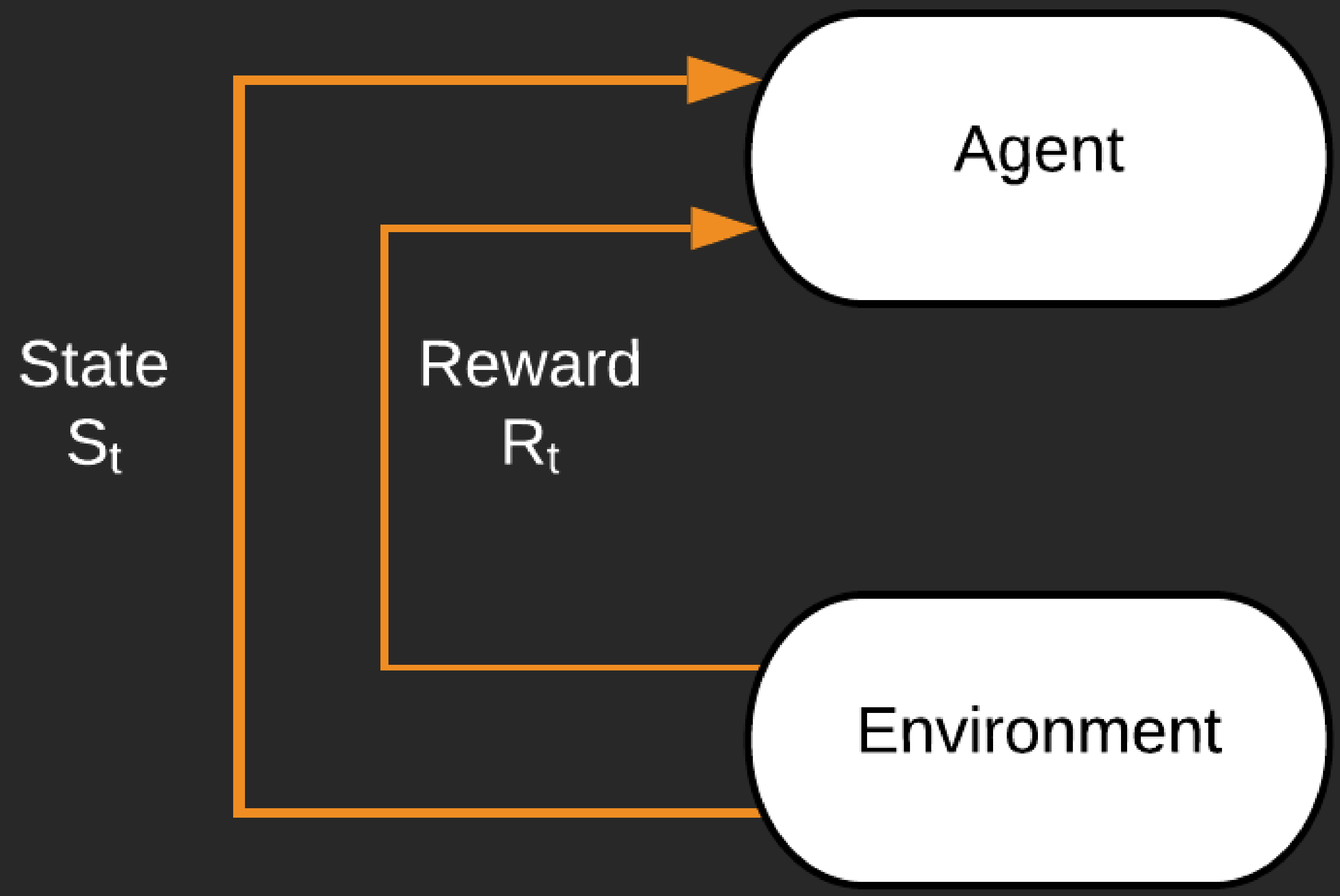
Markov Decision Process (MDP)

Markov Decision Process (MDP)

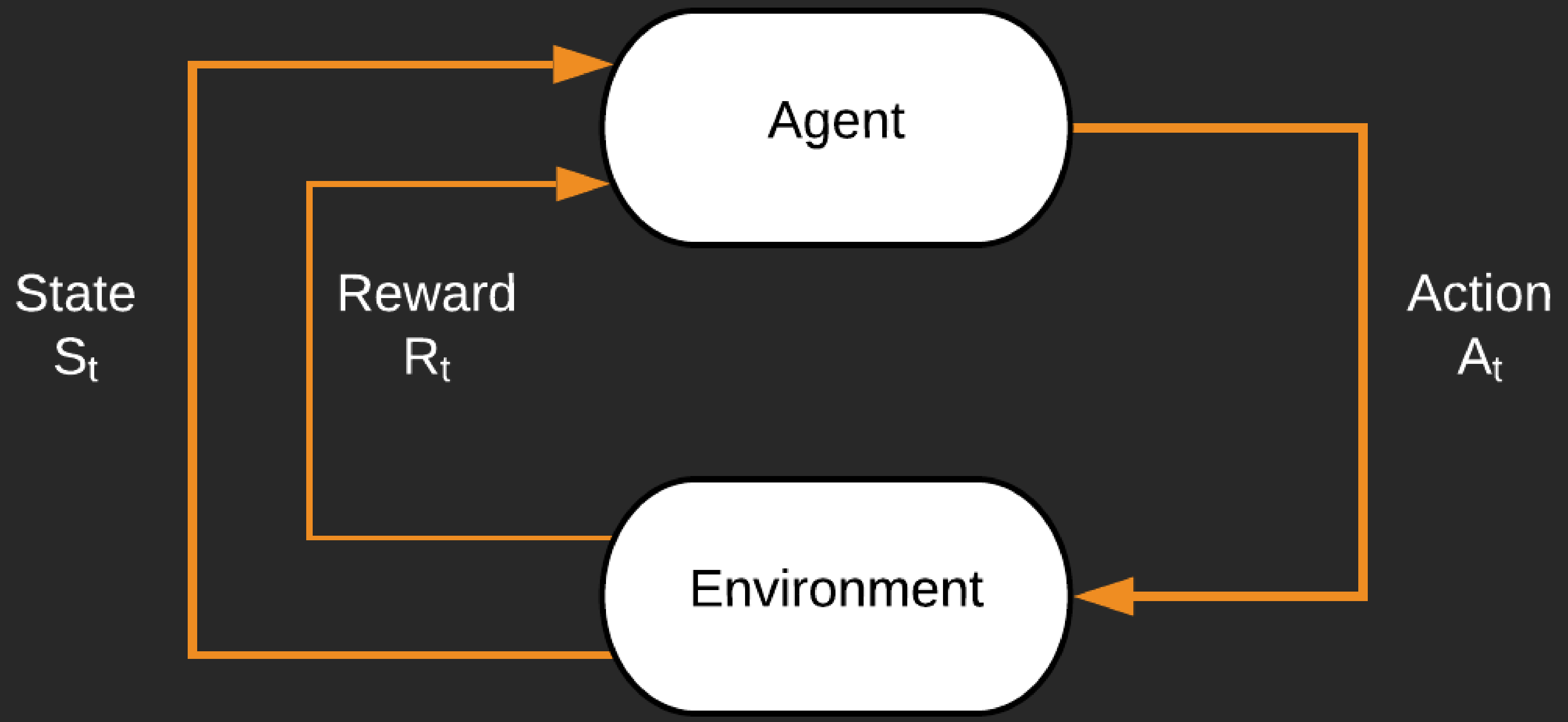
Markov Decision Process (MDP)



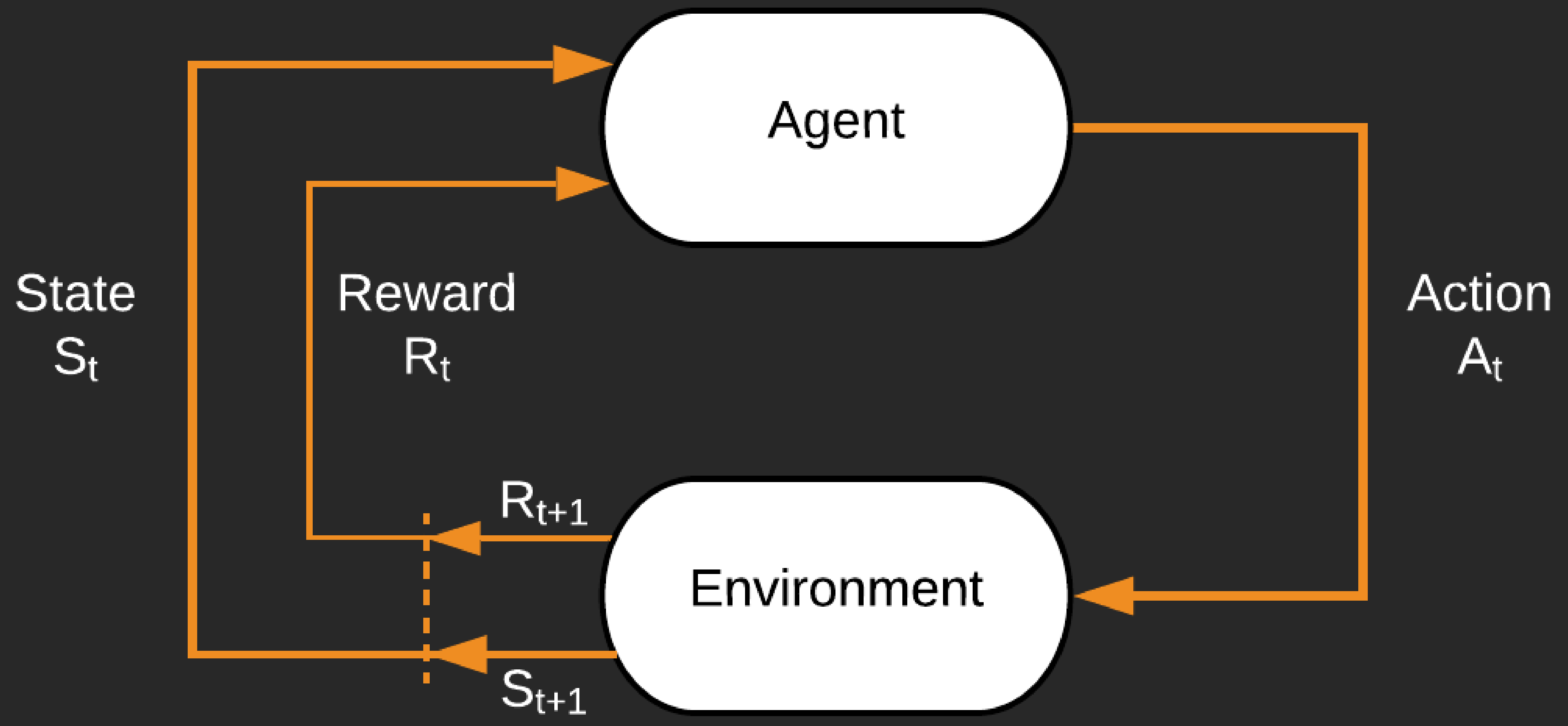
Markov Decision Process (MDP)



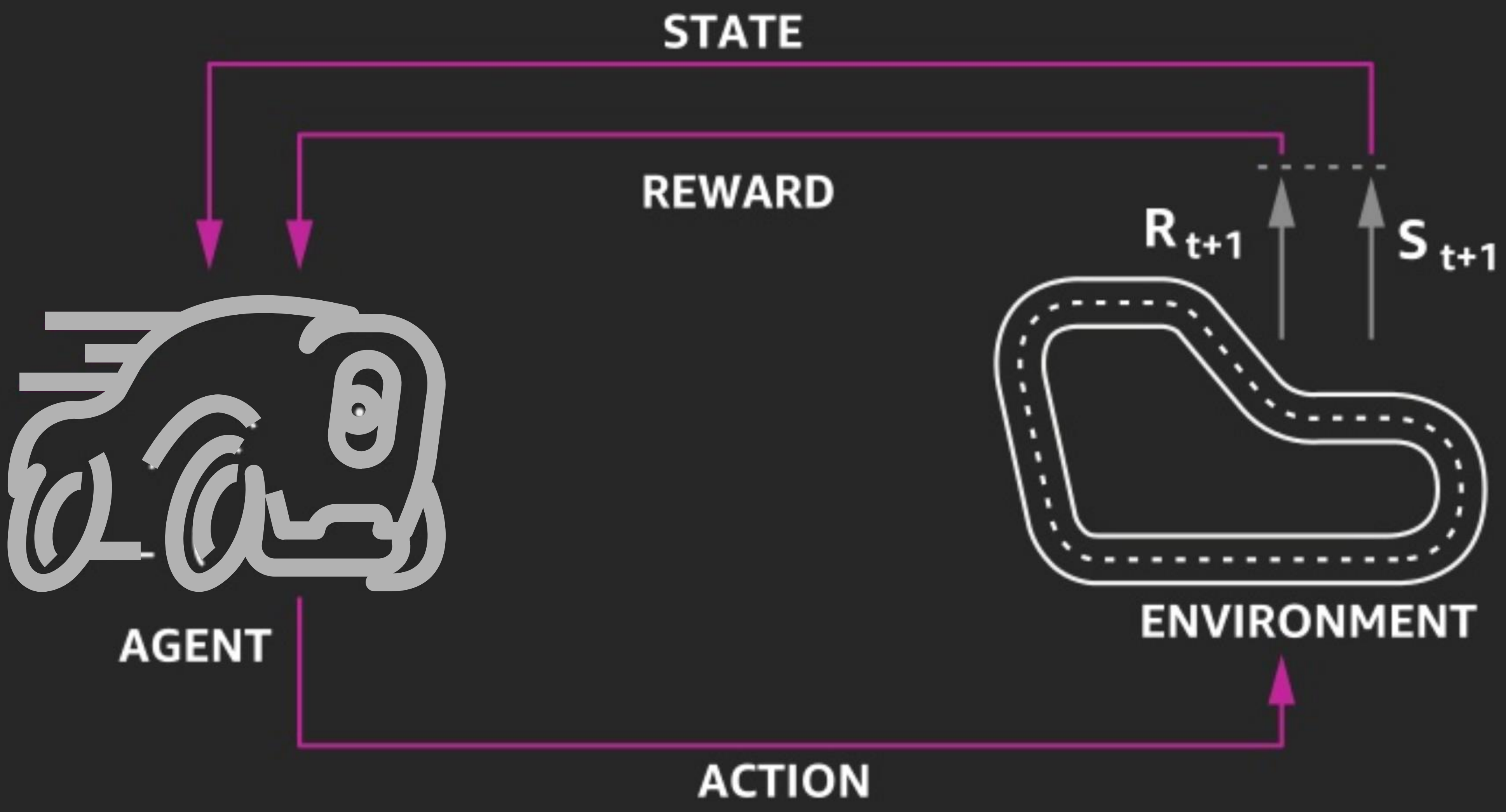
Markov Decision Process (MDP)



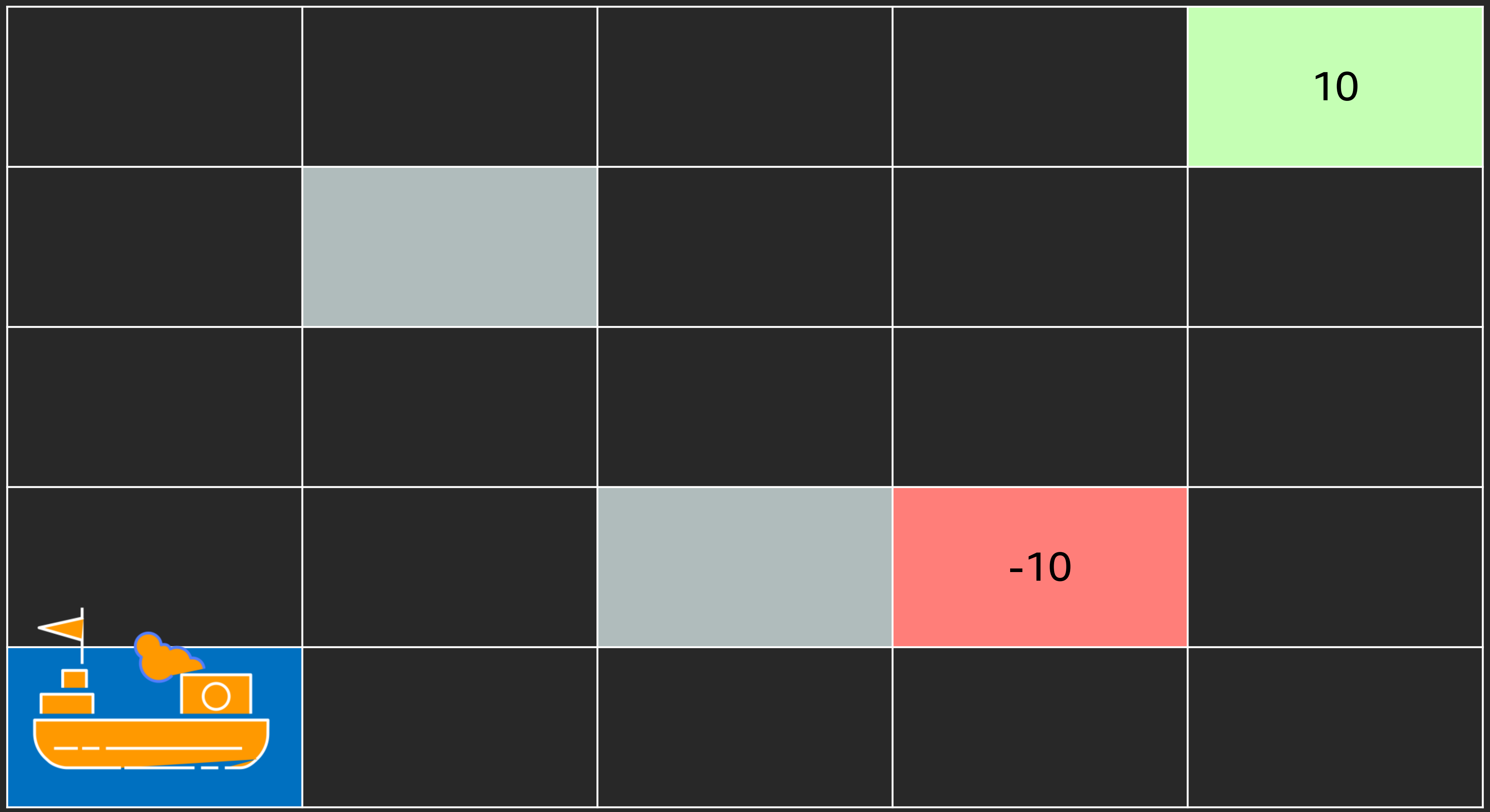
Markov Decision Process (MDP)



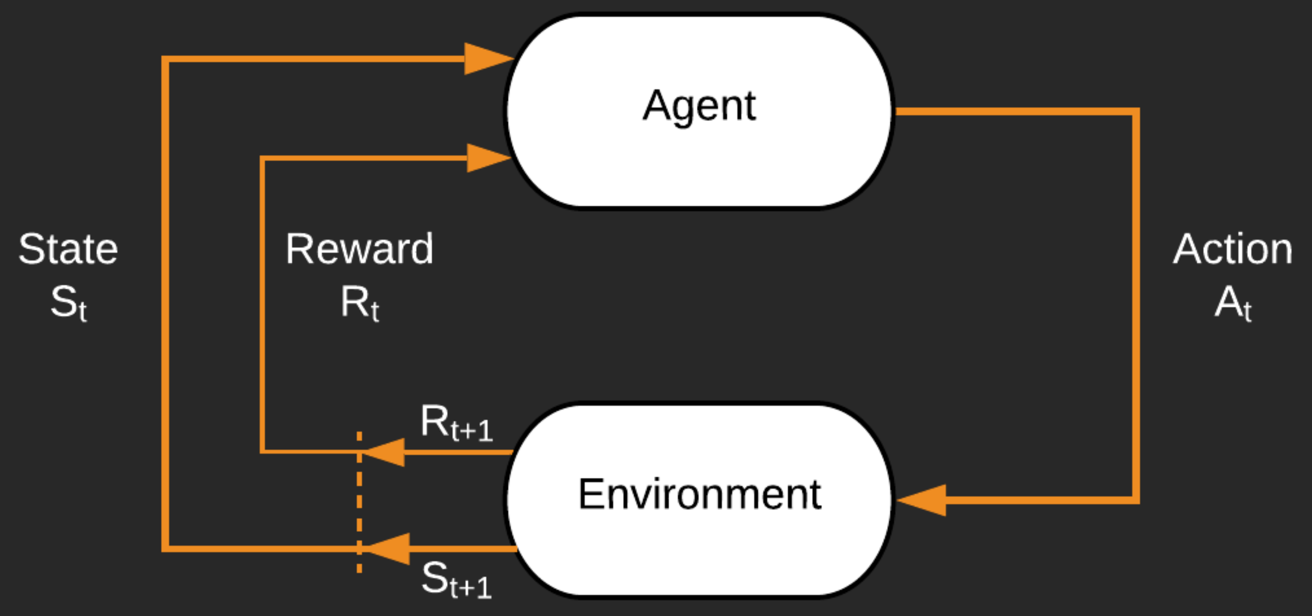
AWS DeepRacer

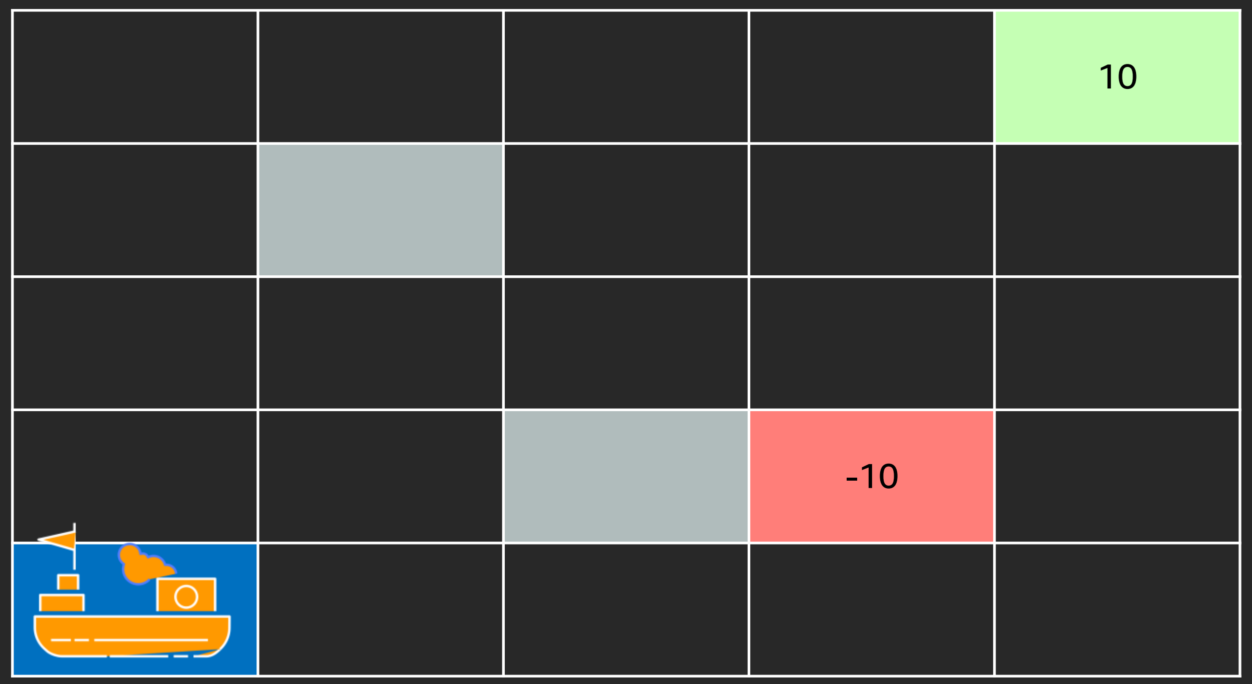


Using RL to solve the puzzle

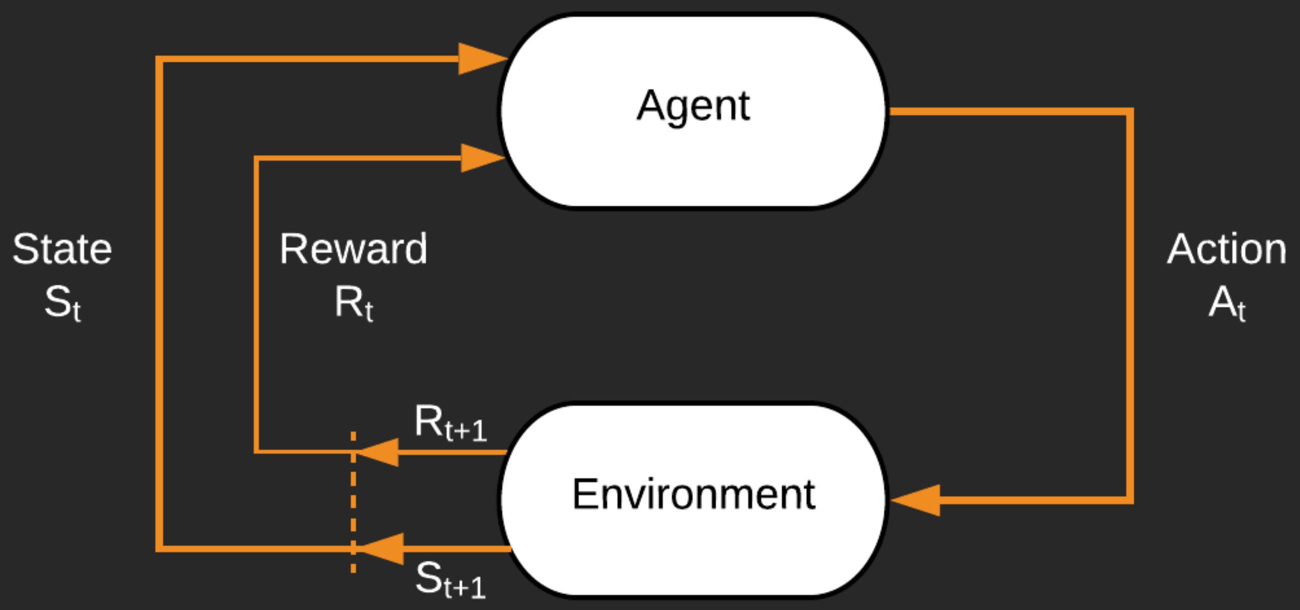


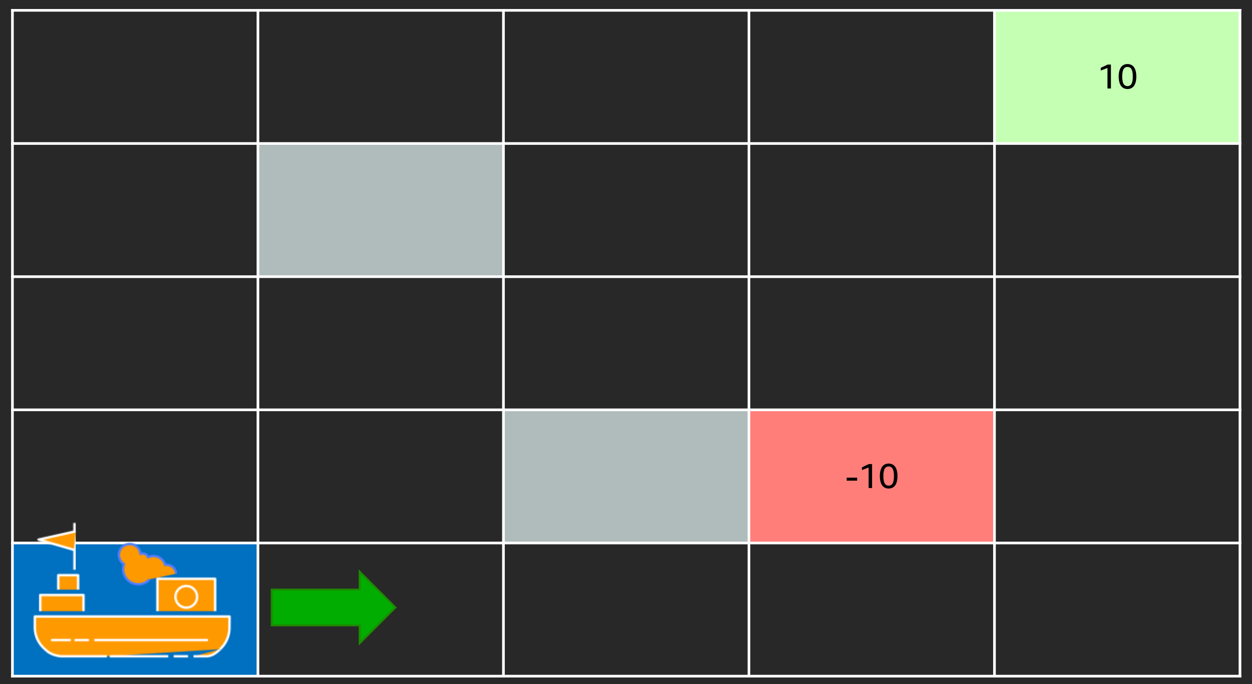
Action Space = Up, Right, or Terminate



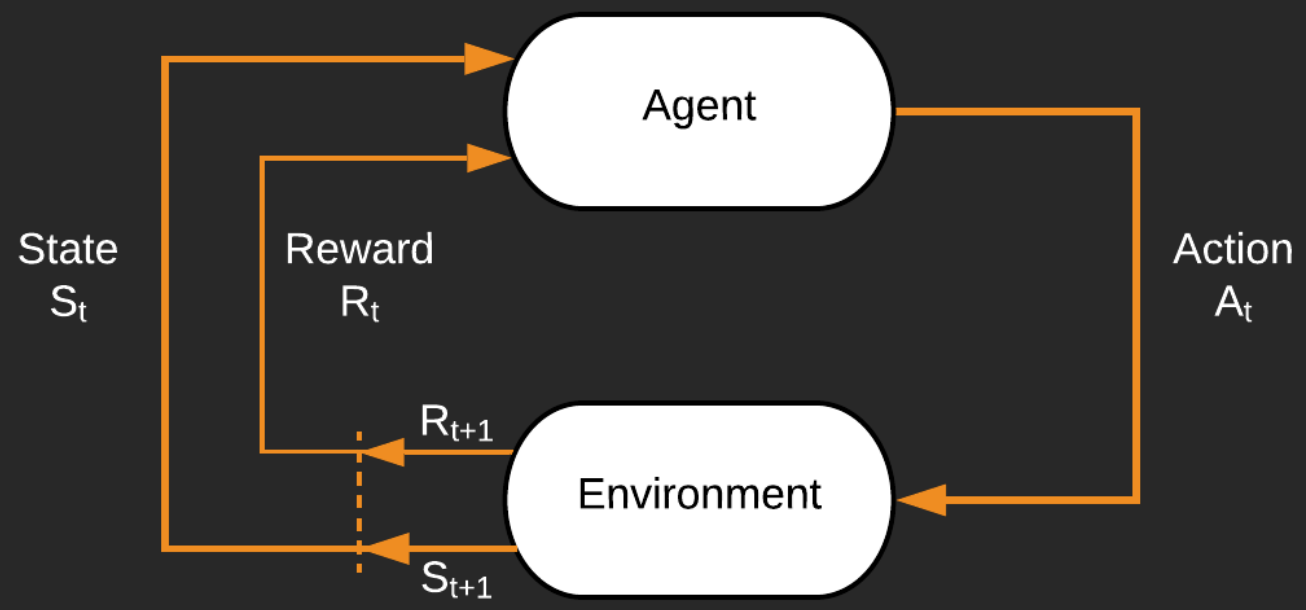


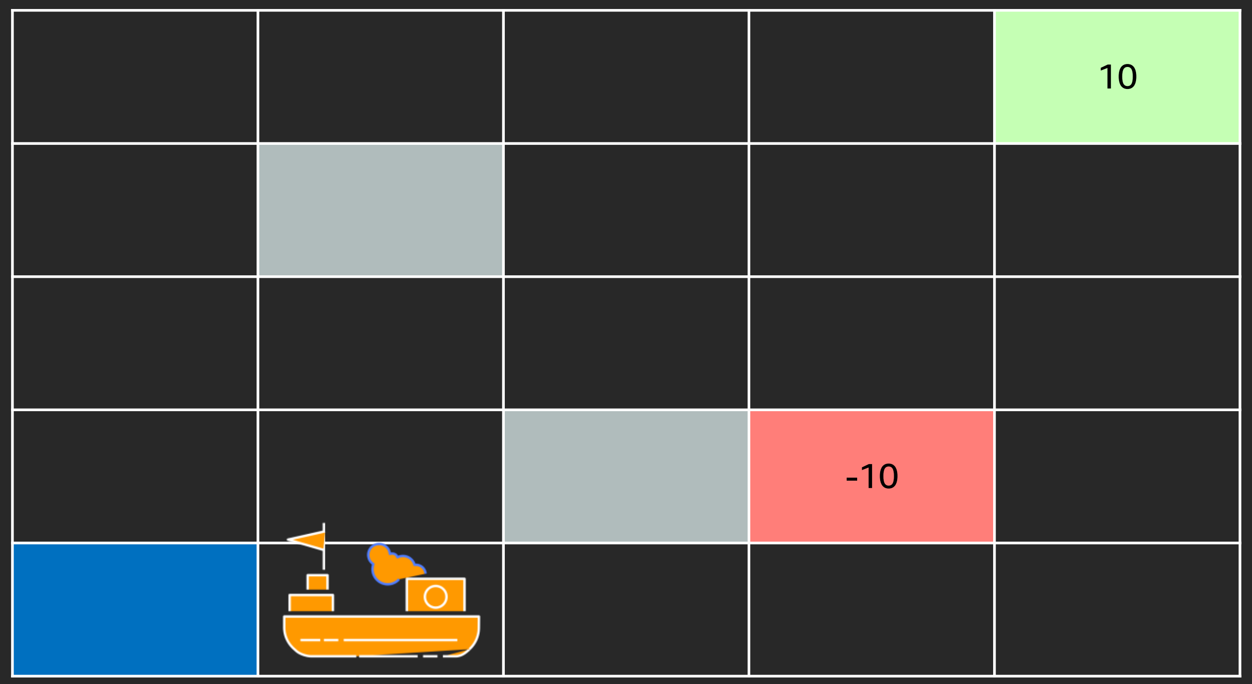
$\text{State}_0 = \{1,1\}, \text{Reward}_0 = \{0\}$





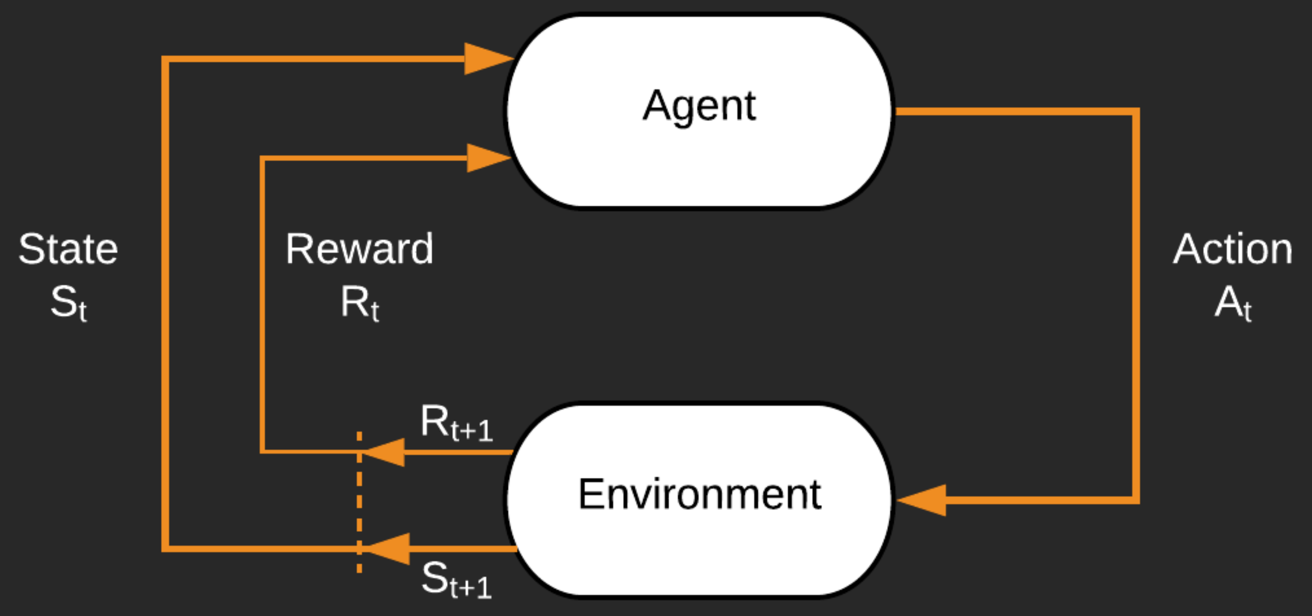
$State_0$ = {1,1}, $Reward_0$ = {0}, $Action_0$ = {R}

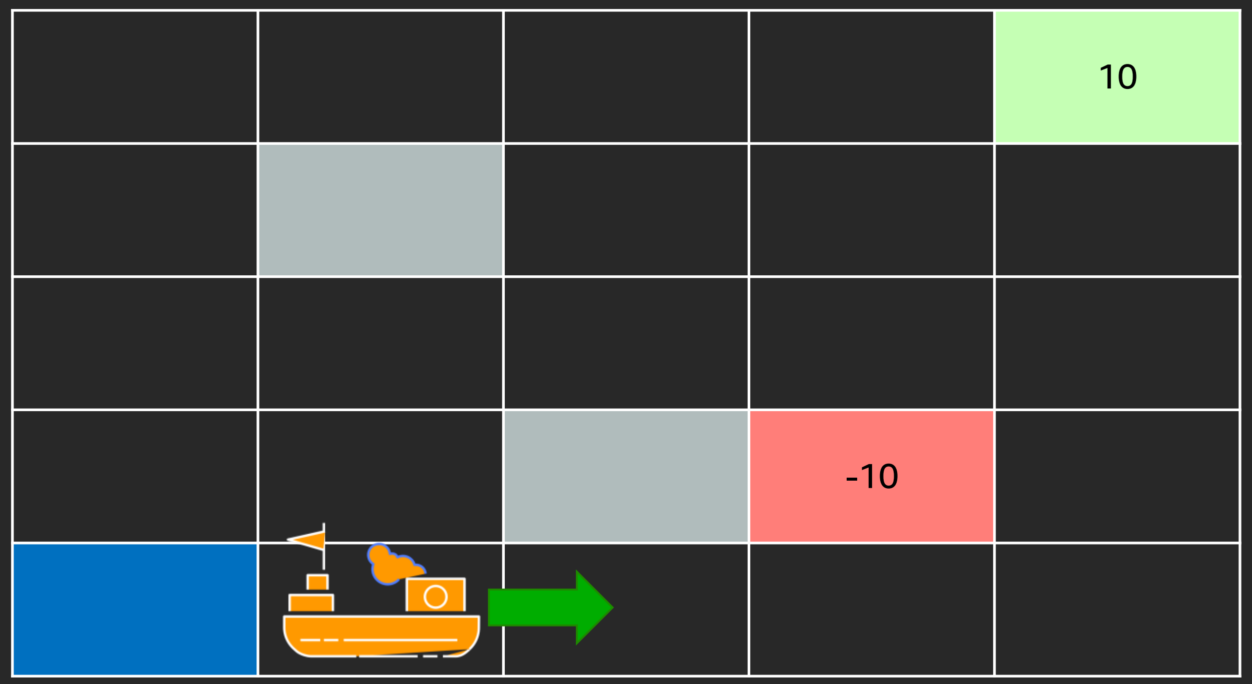




$\text{State}_0 = \{1,1\}$, $\text{Reward}_0 = \{0\}$, $\text{Action}_0 = \{R\}$

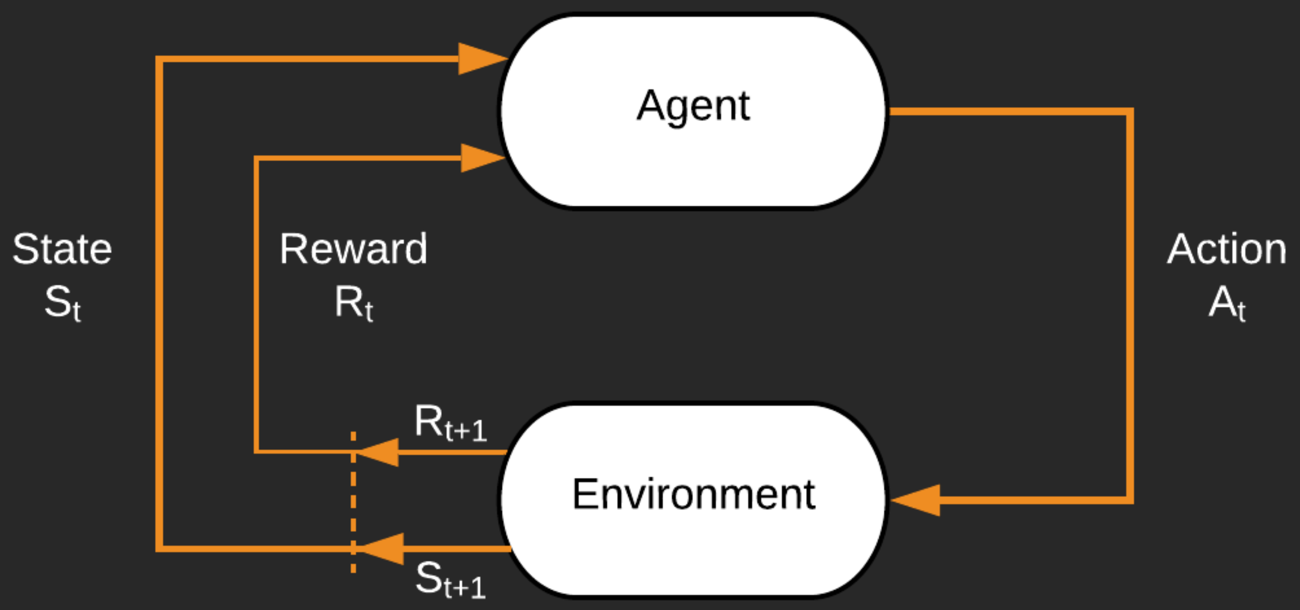
$\text{State}_1 = \{2,1\}$, $\text{Reward}_1 = \{0\}$

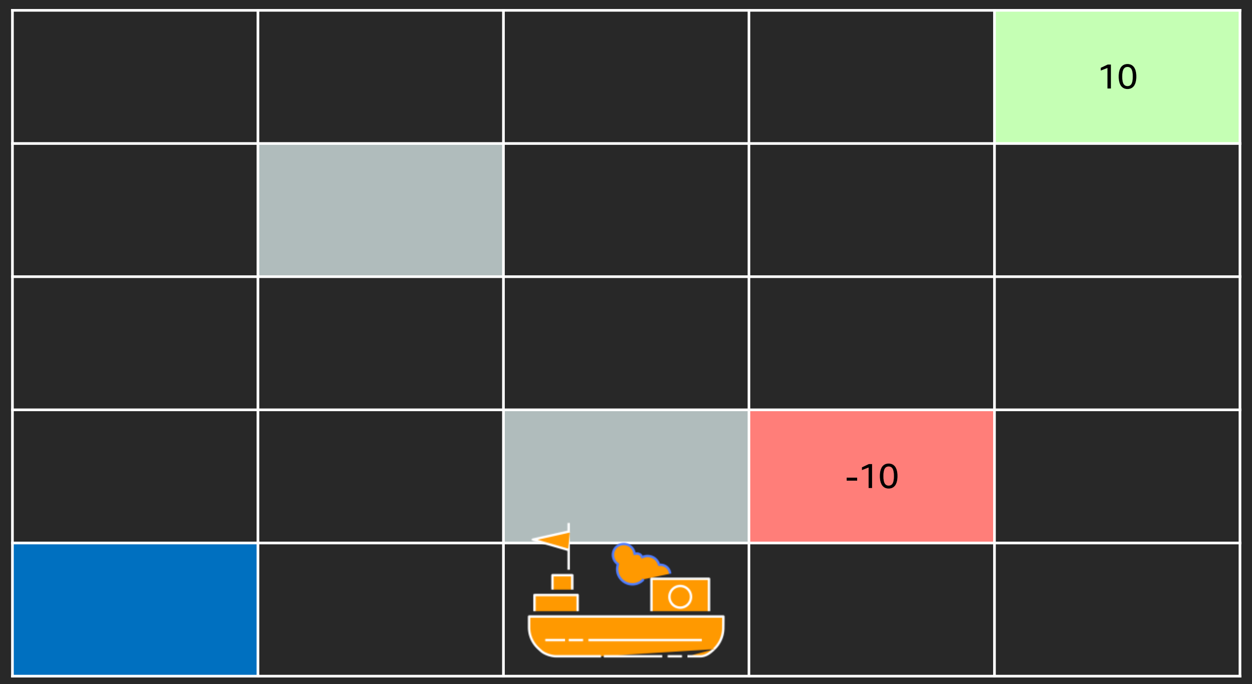




$\text{State}_0 = \{1,1\}$, $\text{Reward}_0 = \{0\}$, $\text{Action}_0 = \{R\}$

$\text{State}_1 = \{2,1\}$, $\text{Reward}_1 = \{0\}$, $\text{Action}_1 = \{R\}$

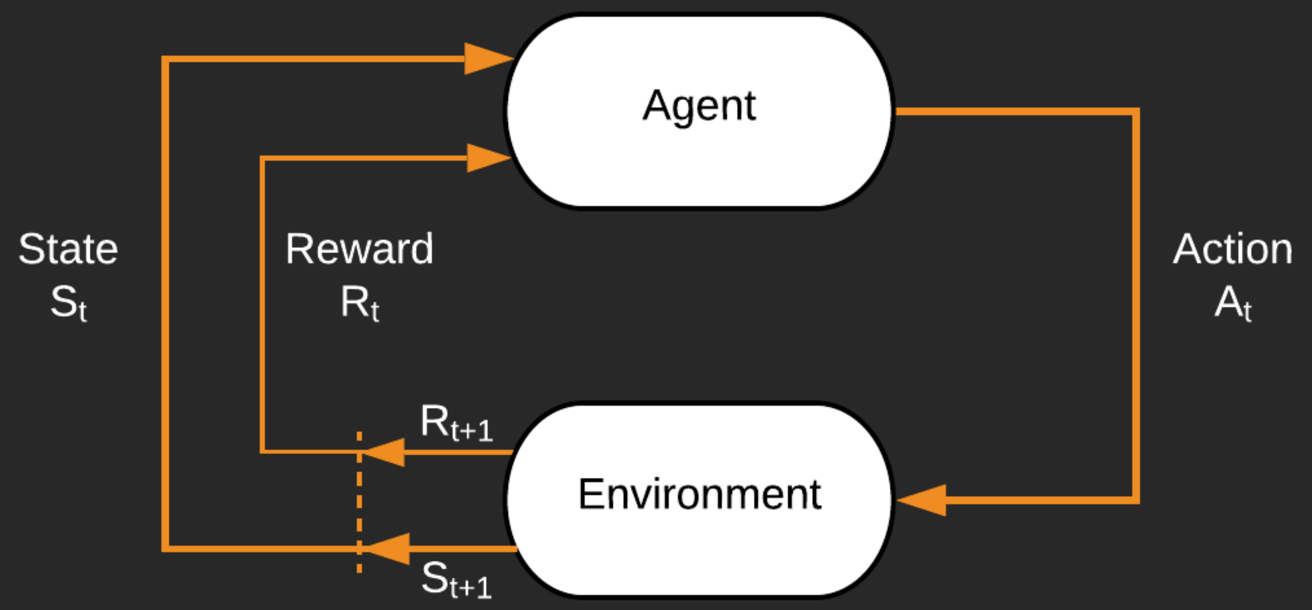


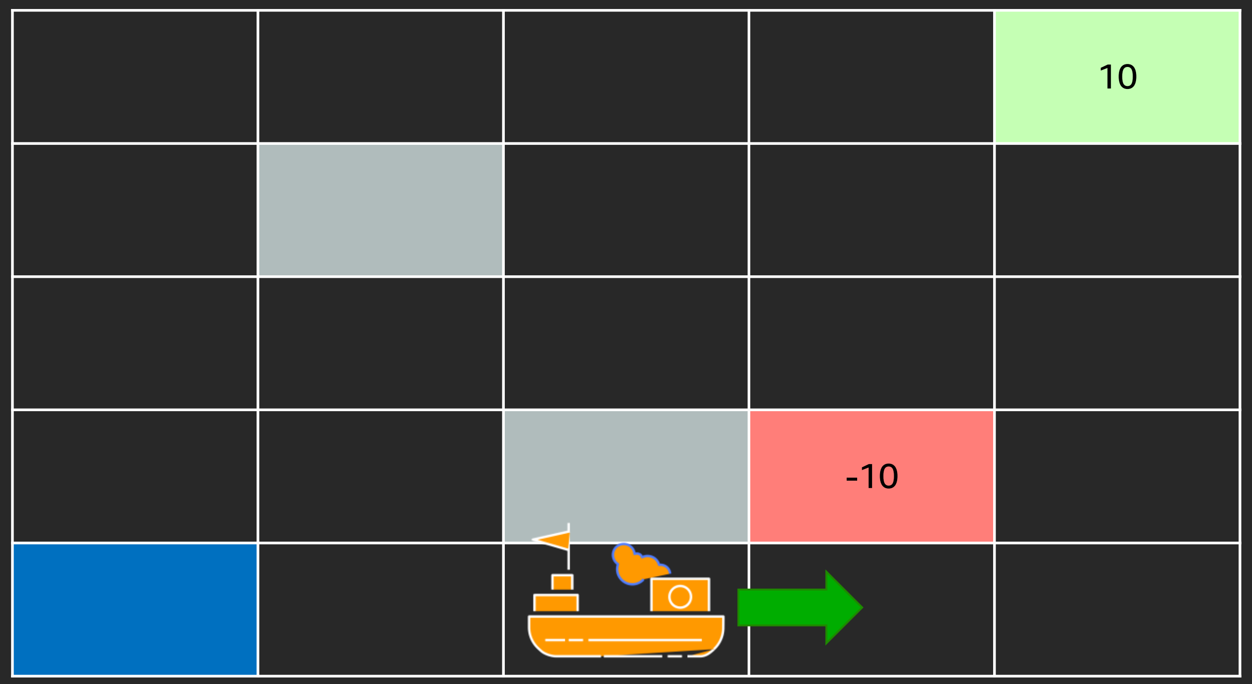


$$State_0 = \{1,1\},\ Reward_0 = \{0\},\ Action_0 = \{R\}$$

$$State_1 = \{2,1\},\ Reward_1 = \{0\},\ Action_1 = \{R\}$$

$$State_2 = \{3,1\},\ Reward_2 = \{0\}$$

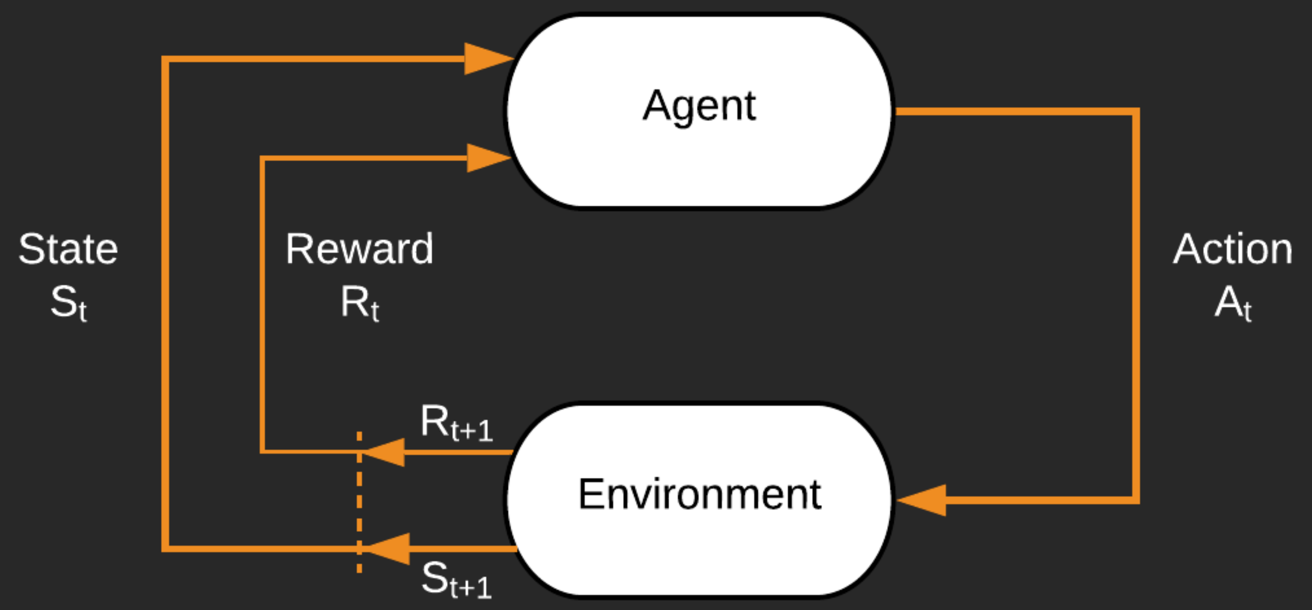


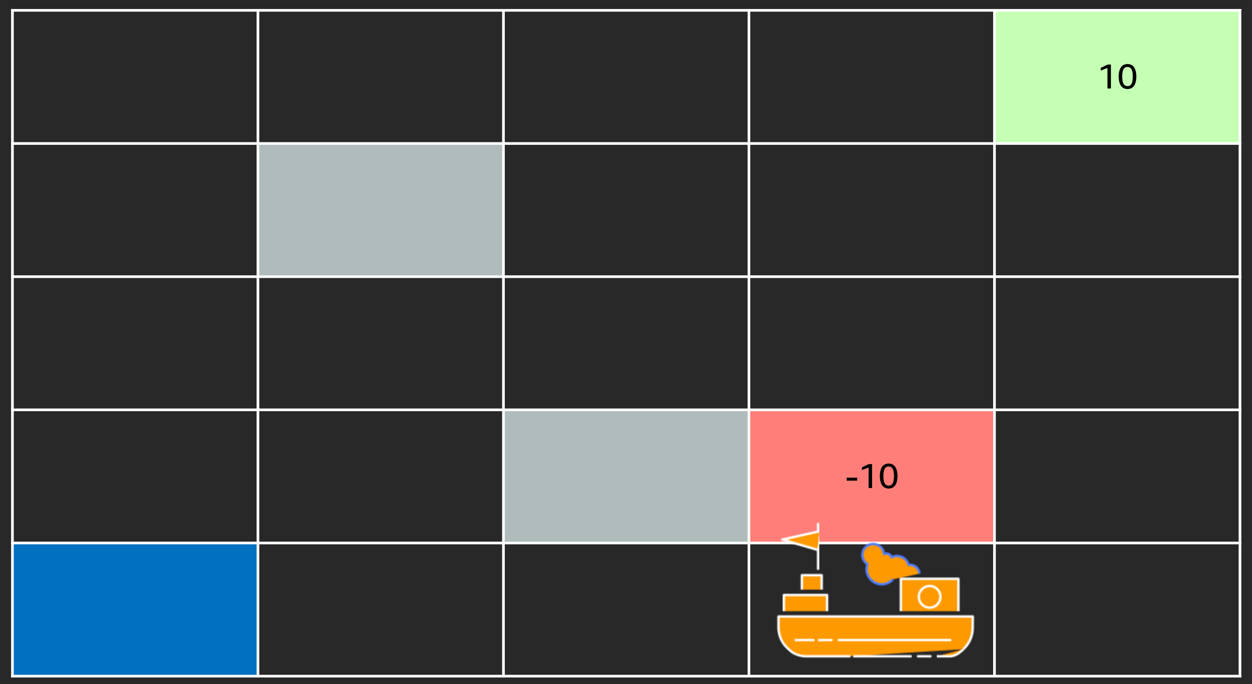


$$State_0 = \{1,1\}, Reward_0 = \{0\}, Action_0 = \{R\}$$

$$State_1 = \{2,1\}, Reward_1 = \{0\}, Action_1 = \{R\}$$

$$State_2 = \{3,1\}, Reward_2 = \{0\}, Action_2 = \{R\}$$



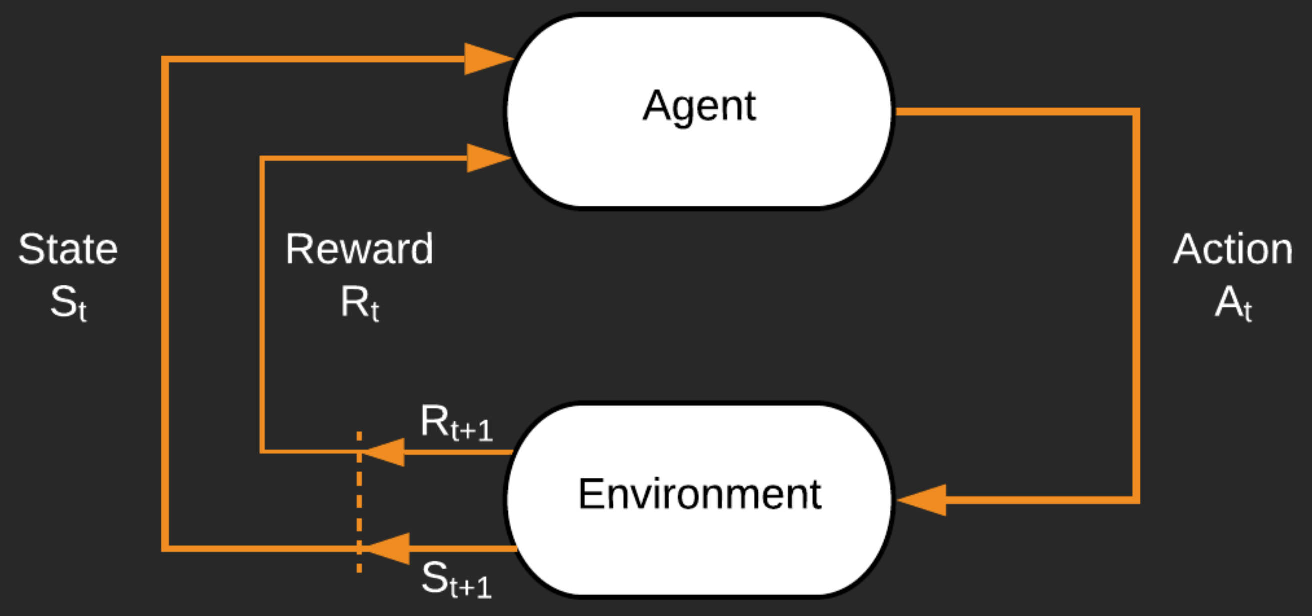


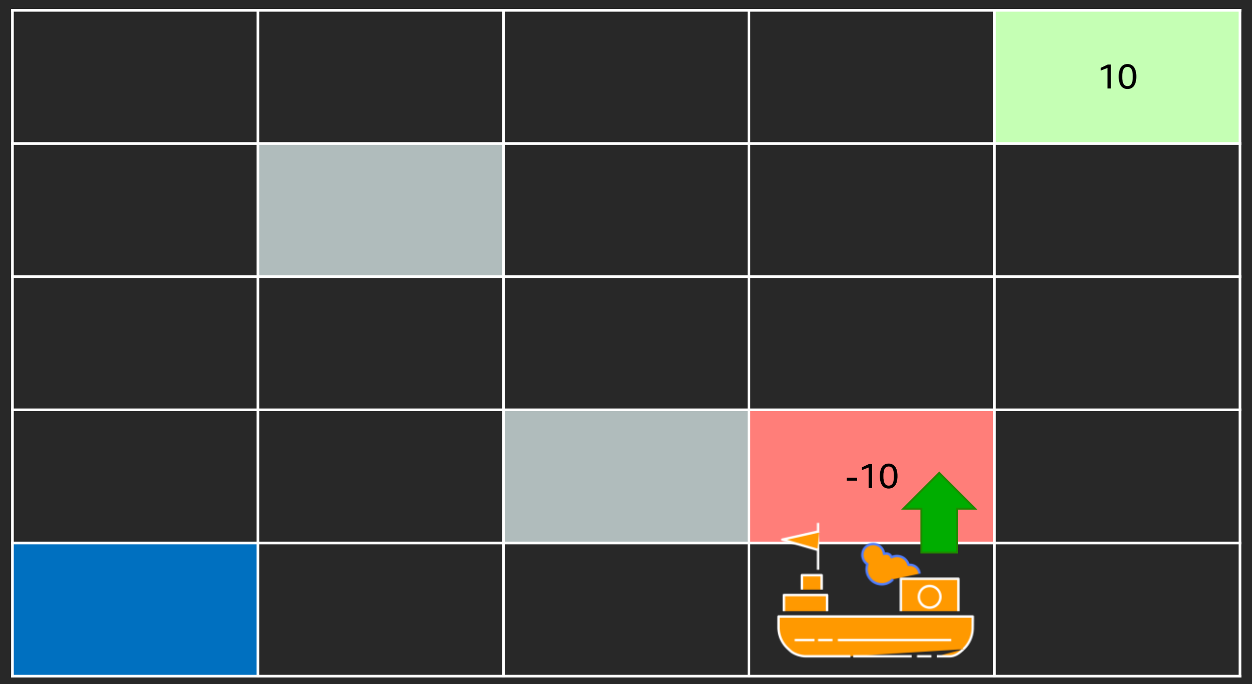
$\text{State}_0 = \{1,1\}$, $\text{Reward}_0 = \{0\}$, $\text{Action}_0 = \{R\}$

$\text{State}_1 = \{2,1\}$, $\text{Reward}_1 = \{0\}$, $\text{Action}_1 = \{R\}$

$\text{State}_2 = \{3,1\}$, $\text{Reward}_2 = \{0\}$, $\text{Action}_2 = \{R\}$

$\text{State}_3 = \{4,1\}$, $\text{Reward}_3 = \{0\}$



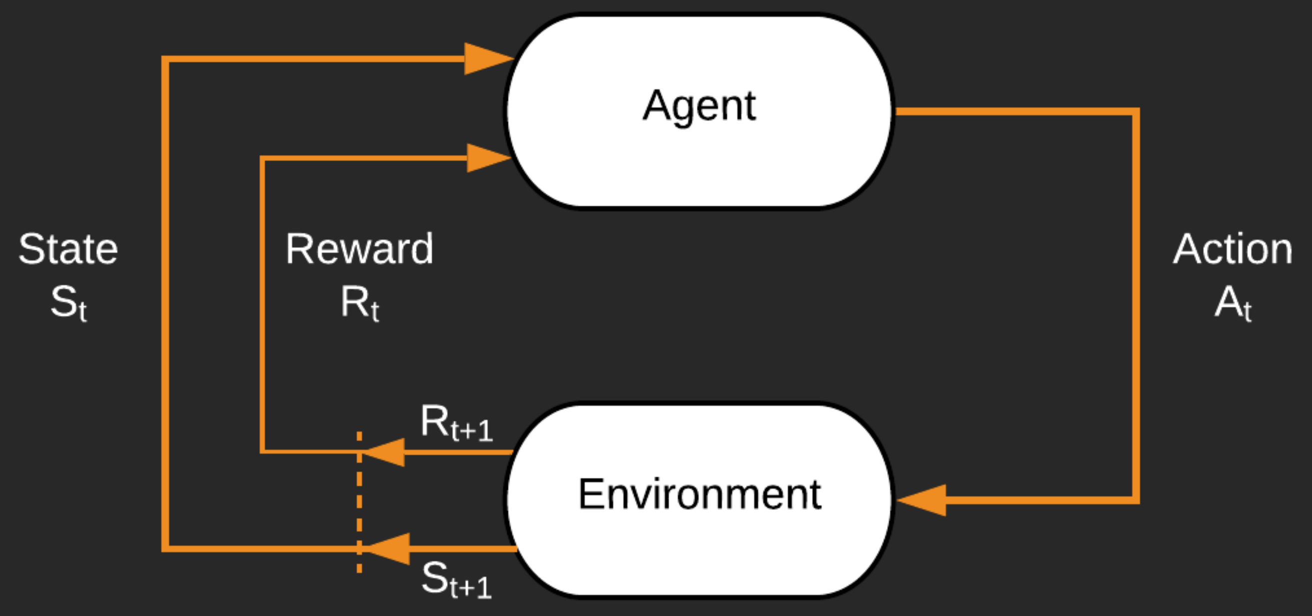


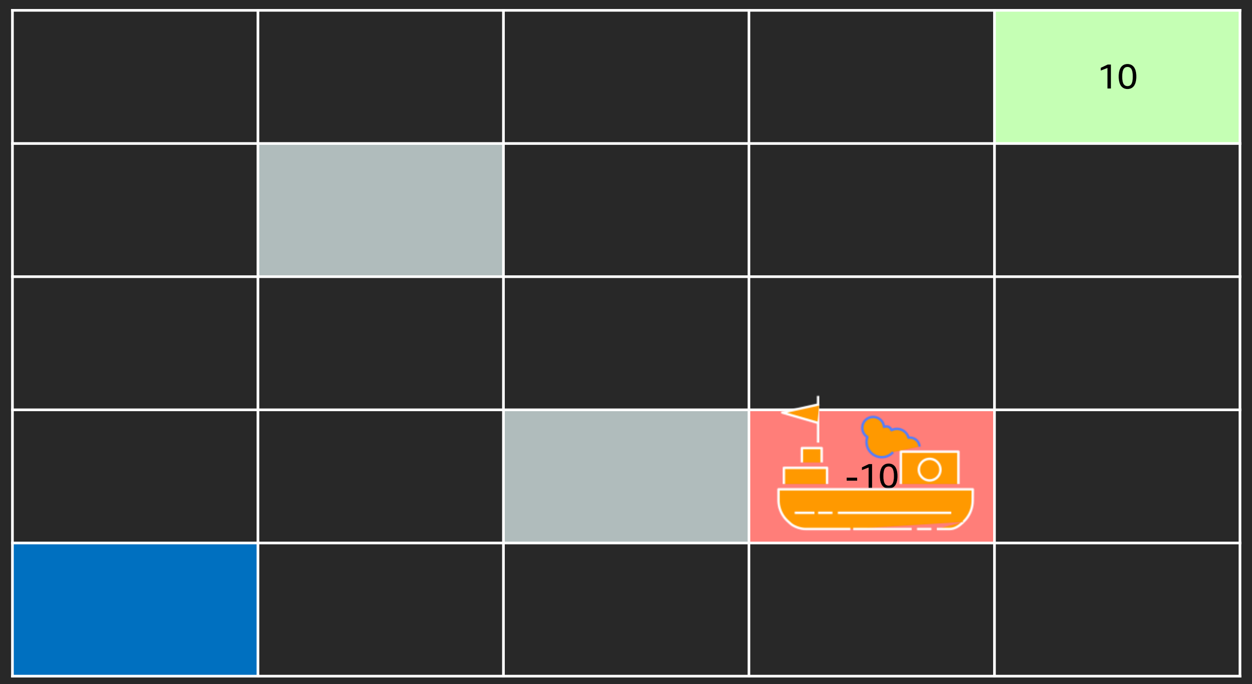
$\text{State}_0 = \{1,1\}$, $\text{Reward}_0 = \{0\}$, $\text{Action}_0 = \{R\}$

$\text{State}_1 = \{2,1\}$, $\text{Reward}_1 = \{0\}$, $\text{Action}_1 = \{R\}$

$\text{State}_2 = \{3,1\}$, $\text{Reward}_2 = \{0\}$, $\text{Action}_2 = \{R\}$

$\text{State}_3 = \{4,1\}$, $\text{Reward}_3 = \{0\}$, $\text{Action}_3 = \{U\}$





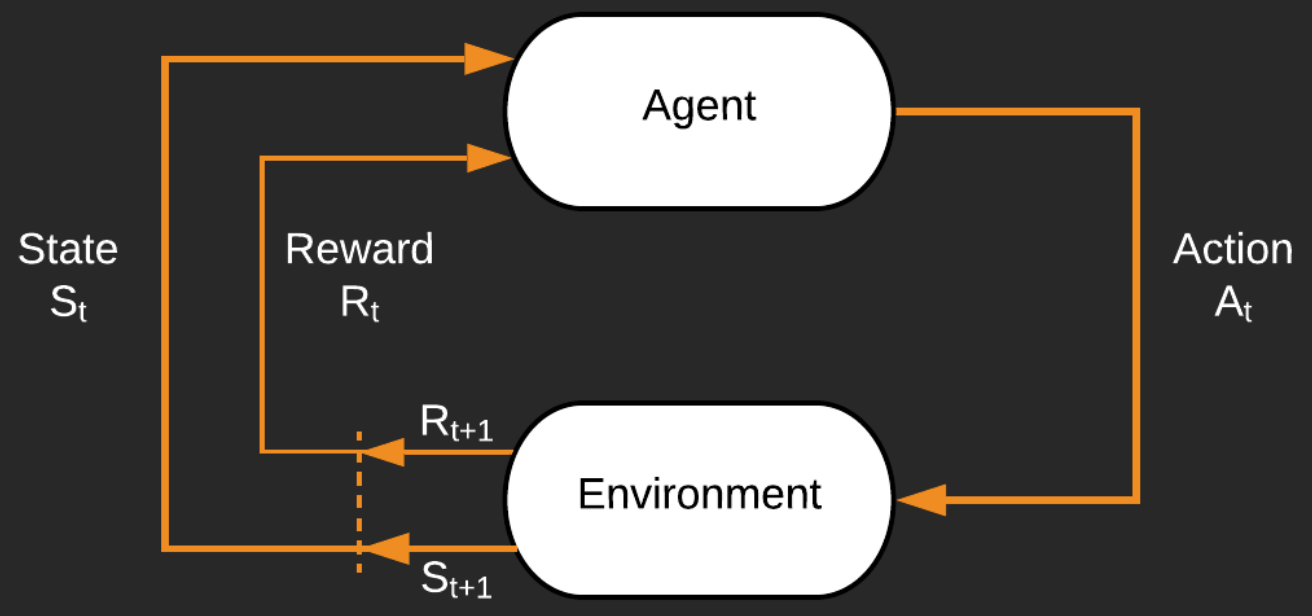
$State_0 = \{1,1\}$, $Reward_0 = \{0\}$, $Action_0 = \{R\}$

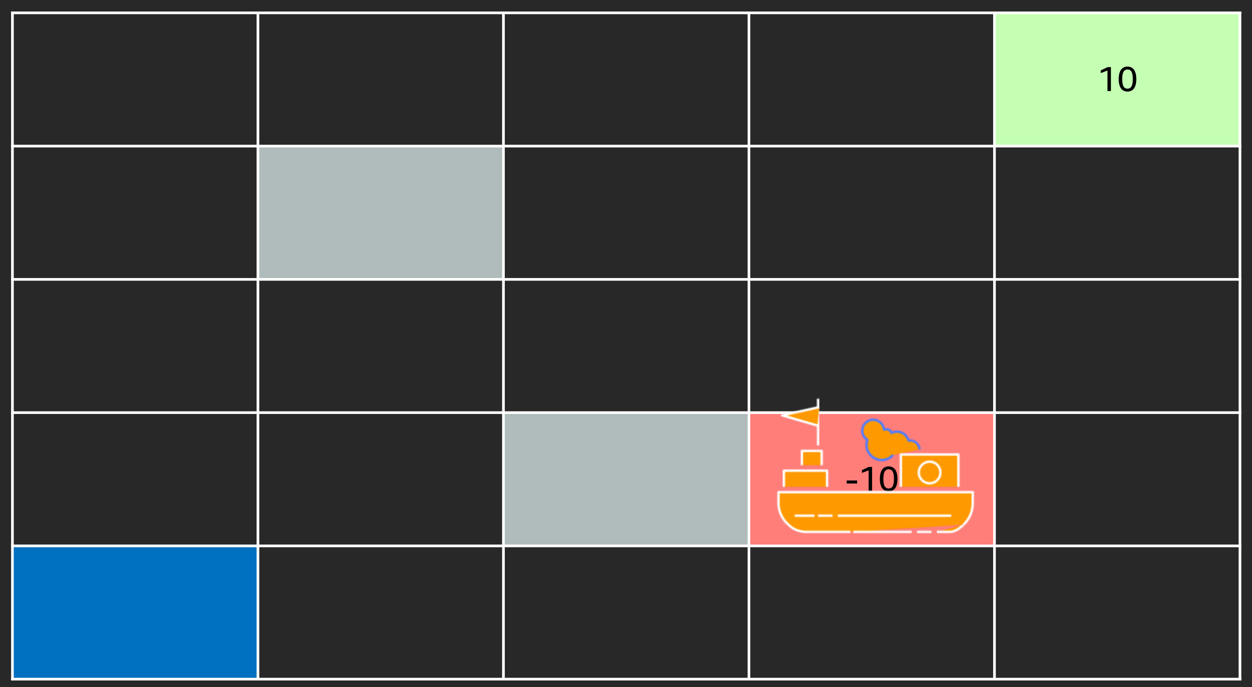
$State_1 = \{2,1\}$, $Reward_1 = \{0\}$, $Action_1 = \{R\}$

$State_2 = \{3,1\}$, $Reward_2 = \{0\}$, $Action_2 = \{R\}$

$State_3 = \{4,1\}$, $Reward_3 = \{0\}$, $Action_3 = \{U\}$

$State_4 = \{4,2\}$, $Reward_4 = \{-10\}$





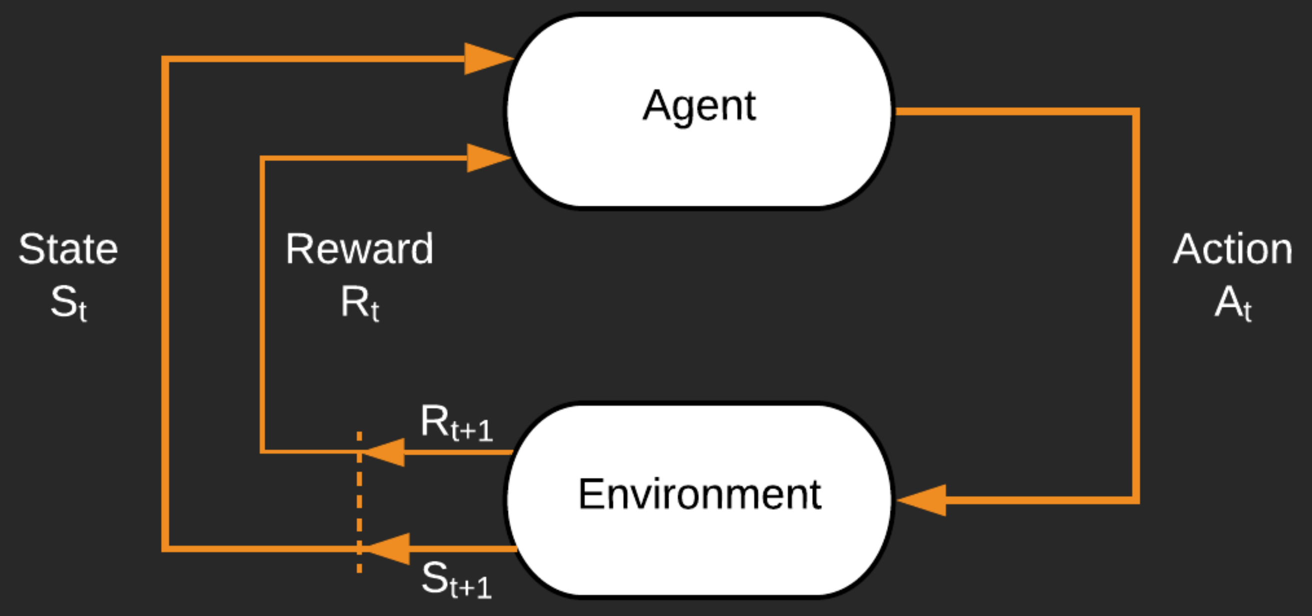
$\text{State}_0 = \{1,1\}$, $\text{Reward}_0 = \{0\}$, $\text{Action}_0 = \{R\}$

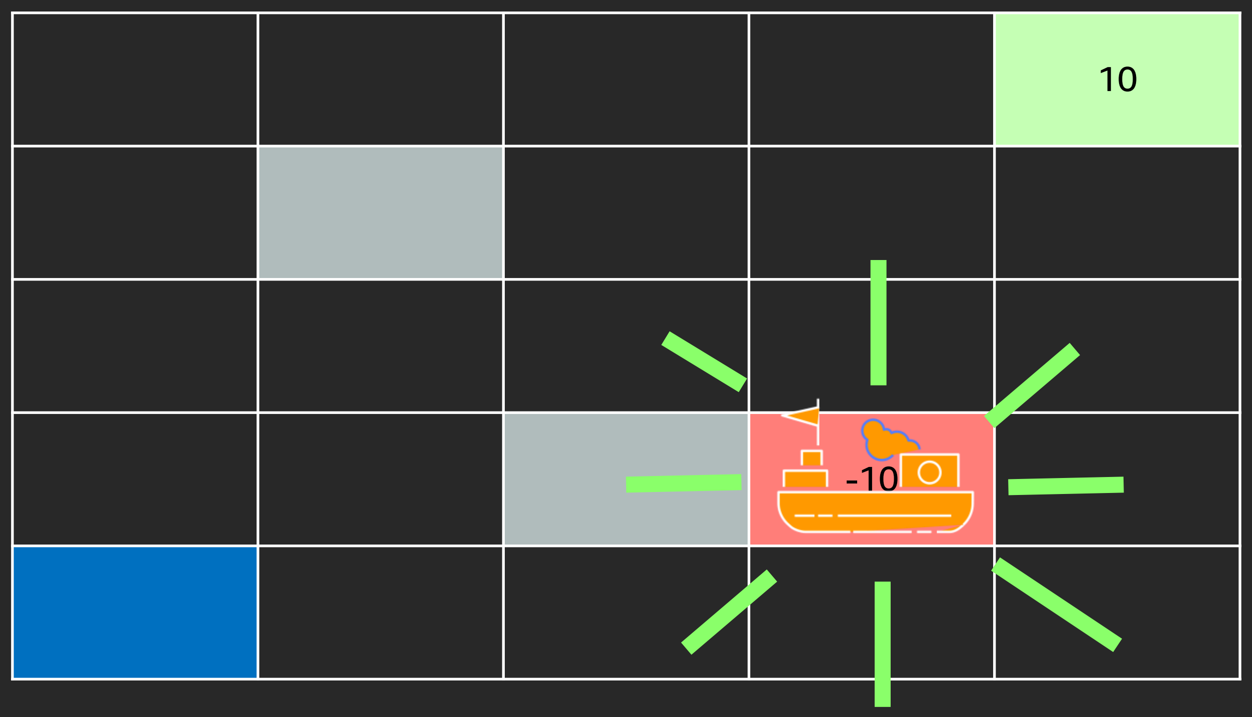
$\text{State}_1 = \{2,1\}$, $\text{Reward}_1 = \{0\}$, $\text{Action}_1 = \{R\}$

$\text{State}_2 = \{3,1\}$, $\text{Reward}_2 = \{0\}$, $\text{Action}_2 = \{R\}$

$\text{State}_3 = \{4,1\}$, $\text{Reward}_3 = \{0\}$, $\text{Action}_3 = \{U\}$

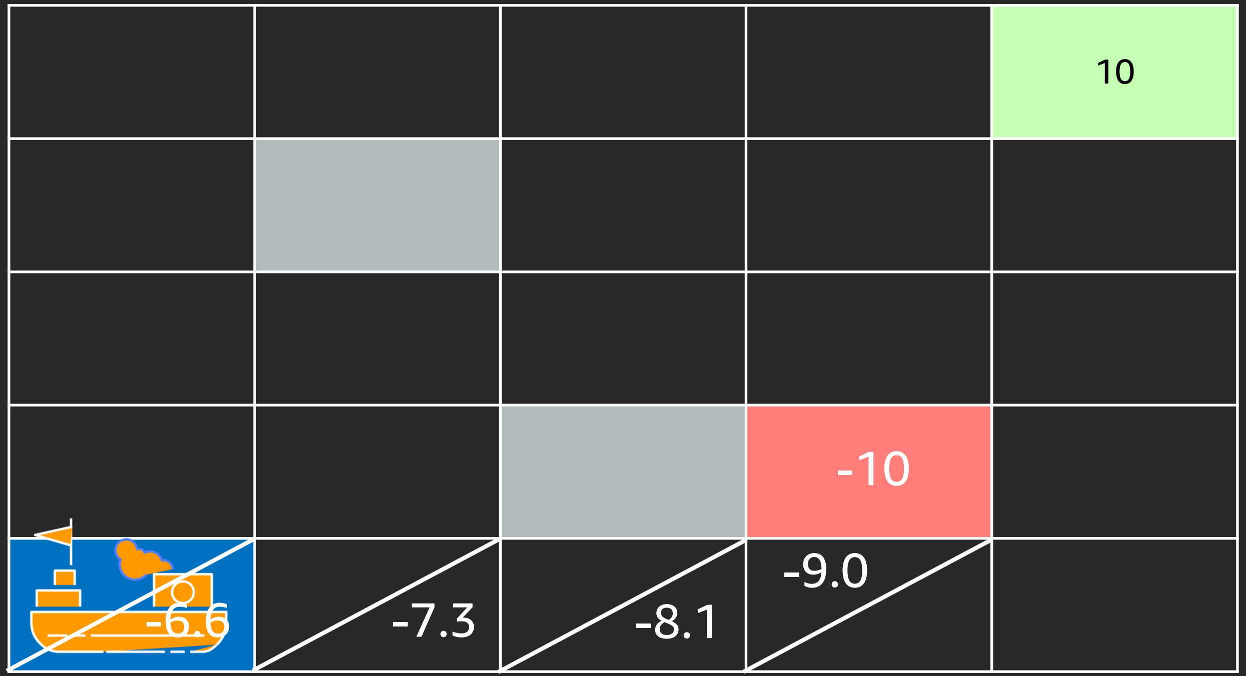
$\text{State}_4 = \{4,2\}$, $\text{Reward}_4 = \{-10\}$, $\text{Action}_4 = \{T\}$





$$\left.\begin{array}{l}
\text{State}_0 = \{1,1\},\ \text{Reward}_0 = \{0\},\ \text{Action}_0 = \{R\} \\
\text{State}_1 = \{2,1\},\ \text{Reward}_1 = \{0\},\ \text{Action}_1 = \{R\} \\
\text{State}_2 = \{3,1\},\ \text{Reward}_2 = \{0\},\ \text{Action}_2 = \{R\} \\
\text{State}_3 = \{4,1\},\ \text{Reward}_3 = \{0\},\ \text{Action}_3 = \{U\} \\
\text{State}_4 = \{4,2\},\ \text{Reward}_4 = \{-10\},\ \text{Action}_4 = \{T\}
\end{array}\right\}\ \text{Episode (or trajectory)}$$

State	Up	Right	Terminate	Distance from start
1, 1		$\boxed{-6.6}$		0
2, 1		$\boxed{-7.3}$		1
3, 1		$\boxed{-8.1}$		2
4, 1	$\boxed{-9.0}$			3
4,2			$\boxed{-10}$	4



[1,1,0,R], [2,1,0,R],[3,1,0,R],[4,1,0,U],[4,2,-10,T]

$$\text{Quality}_t = \text{Discount} * \text{Quality}_{t+1}$$

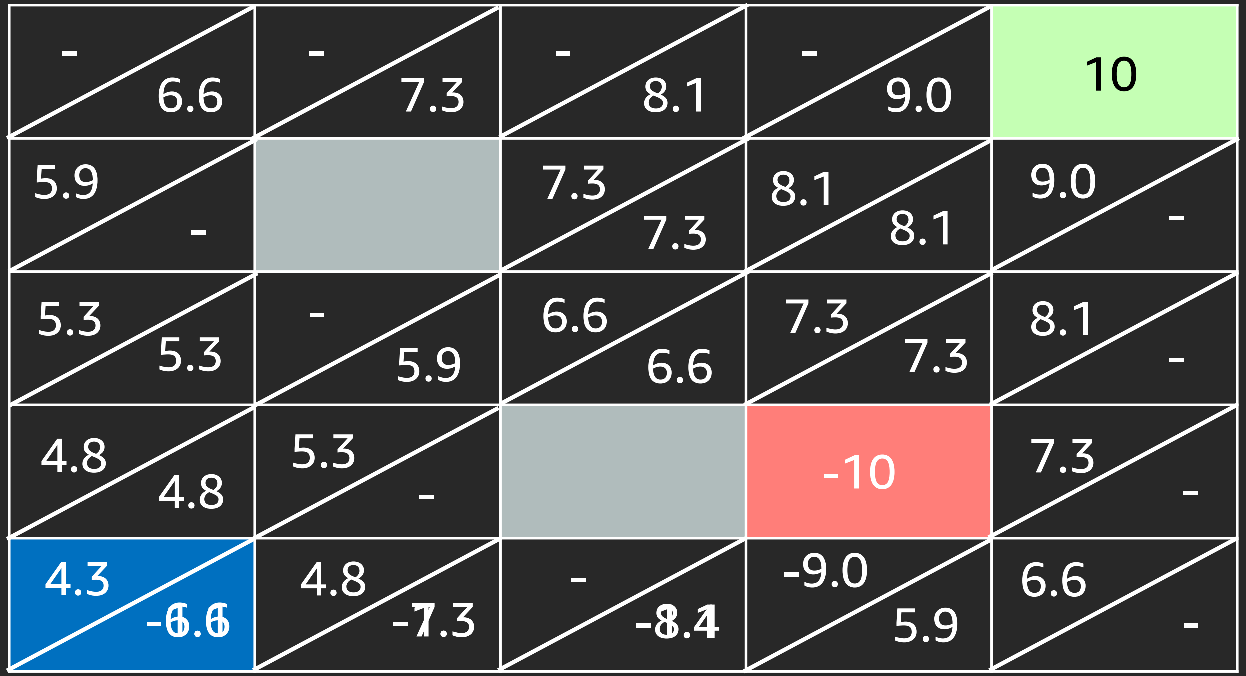
$$-9.0 = 0.9 * -10$$

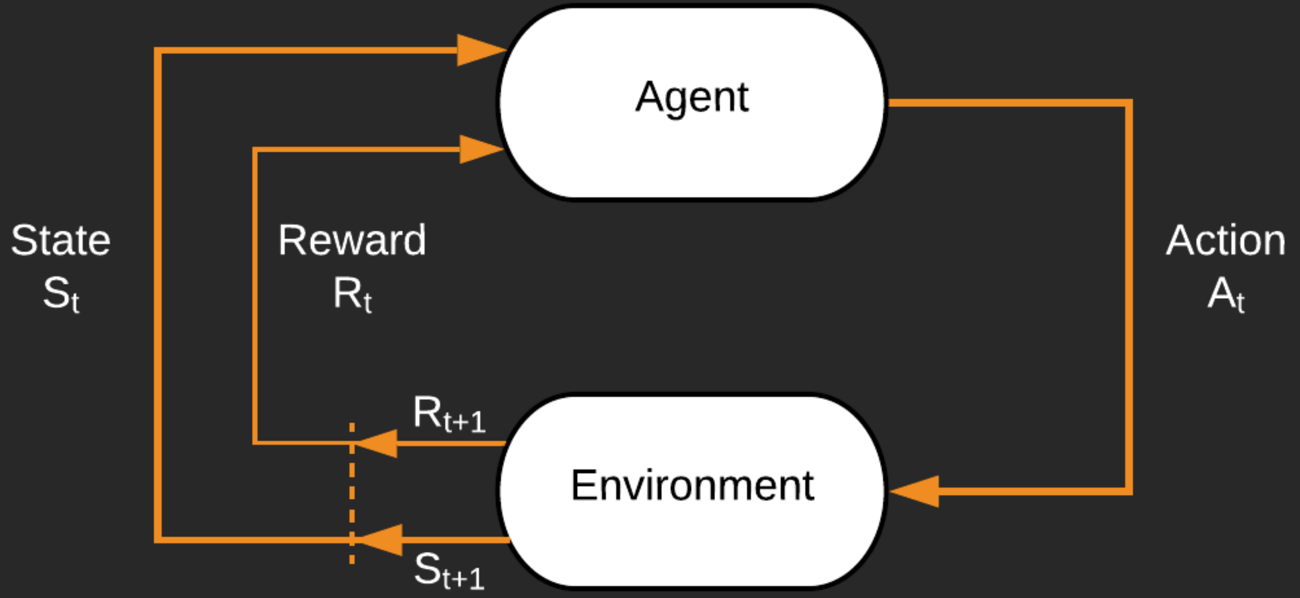
$$-8.1 = 0.9 * -9.0$$

$$-7.3 = 0.9 * -8.1$$

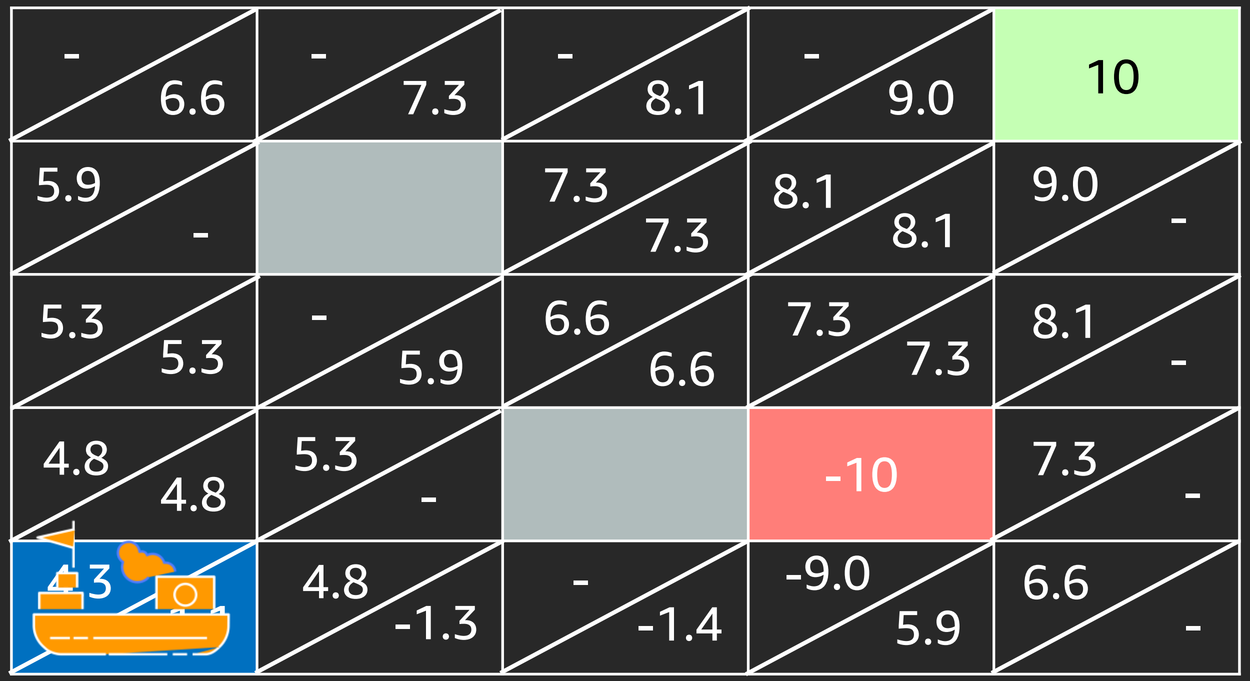
$$-6.6 = 0.9 * -7.3$$

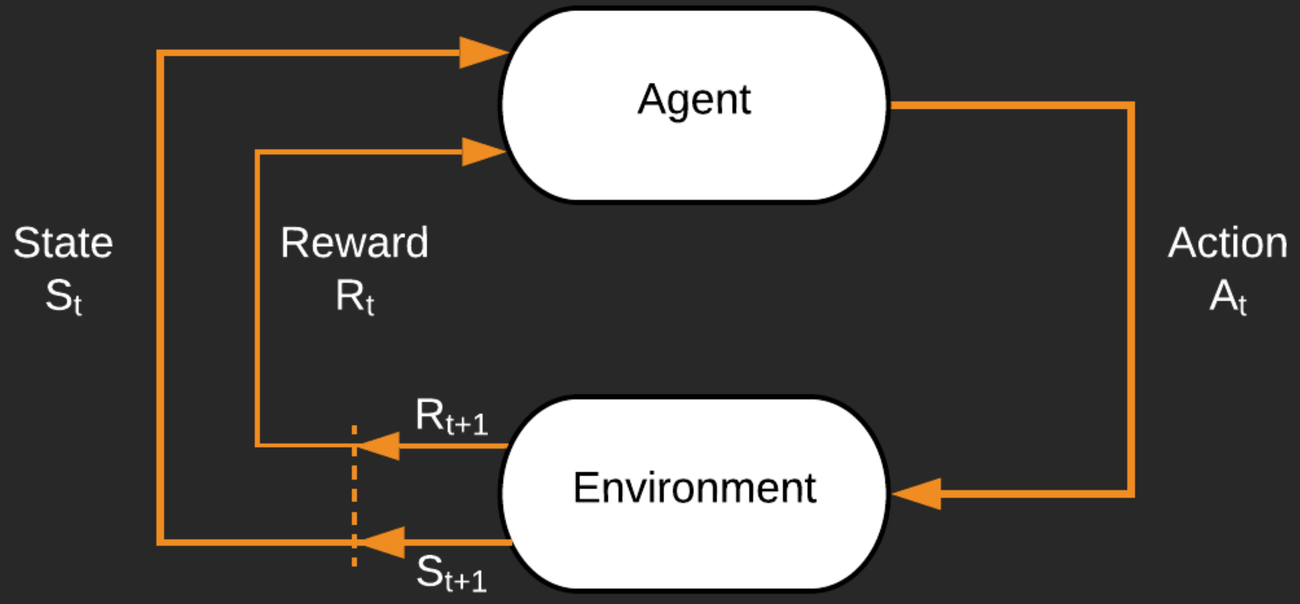
State	Up	Right	Terminate	Distance from start
1,1	4.3	-1.1		0
1,2	4.8	4.8		1
1,3	5.3	5.3		2
1,4	5.9			3
1,5		6.6		4
2,1	4.8	-1.3		1
2,2	5.3			2
2,3		5.9		3
2,4				
2,5		7.3		5
3,1		-1.4		2
3,2				
3,3	6.6	6.6		4
3,4	7.3	7.3		5
3,5		8.1		6
4,1	-9.0	5.9		3
4,2			-10	4
4,3	7.3	7.3		5
4,4	8.1	8.1		6
4,5		9.0		7
5,1	6.6			4
5,2	7.3			5
5,3	8.1			6
5,4	9.0			7
5,5			10	8





State	Up	Right	Terminate	Distance from start
1,1	4.3	-1.1		0
1,2	4.8	4.8		1
1,3	5.3	5.3		2
1,4	5.9			3
1,5		6.6		4
2,1	4.8	-1.3		1
2,2	5.3			2
2,3		5.9		3
2,4				
2,5		7.3		5
3,1		-1.4		2
3,2				
3,3	6.6	6.6		4
3,4	7.3	7.3		5
3,5		8.1		6
4,1	-9.0	5.9		3
4,2			-10	4
4,3	7.3	7.3		5
4,4	8.1	8.1		6
4,5		9.0		7
5,1	6.6			4
5,2	7.3			5
5,3	8.1			6
5,4	9.0			7
5,5			10	8

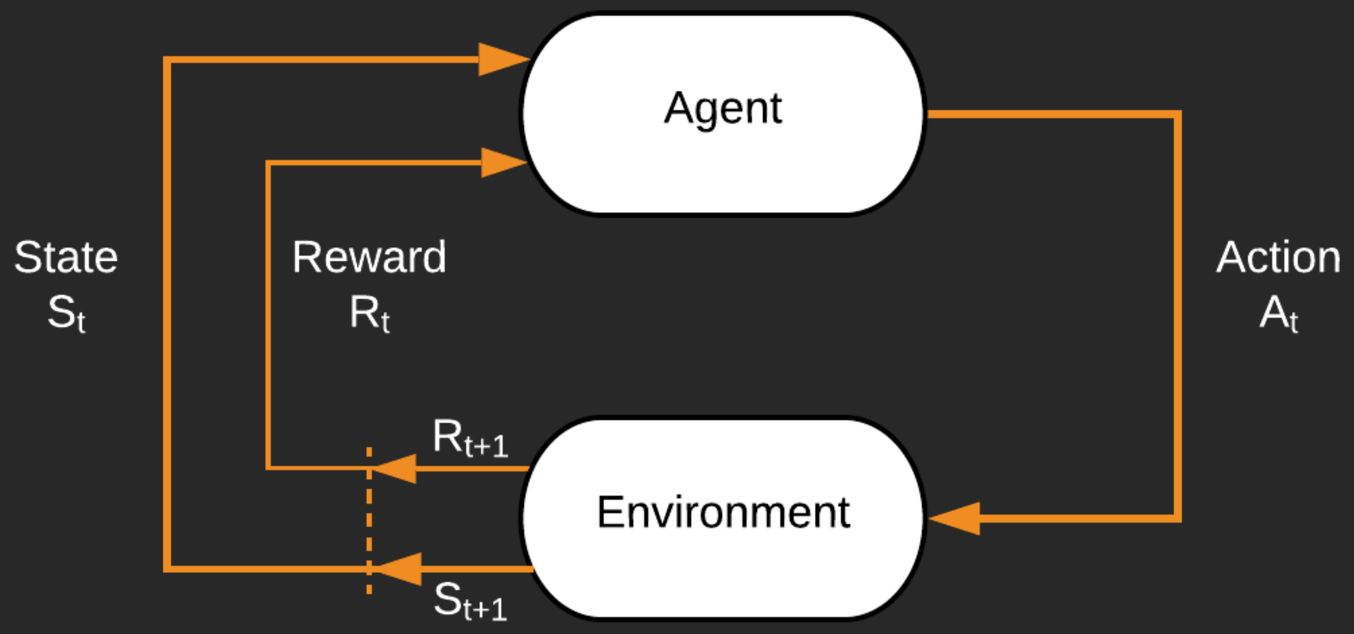




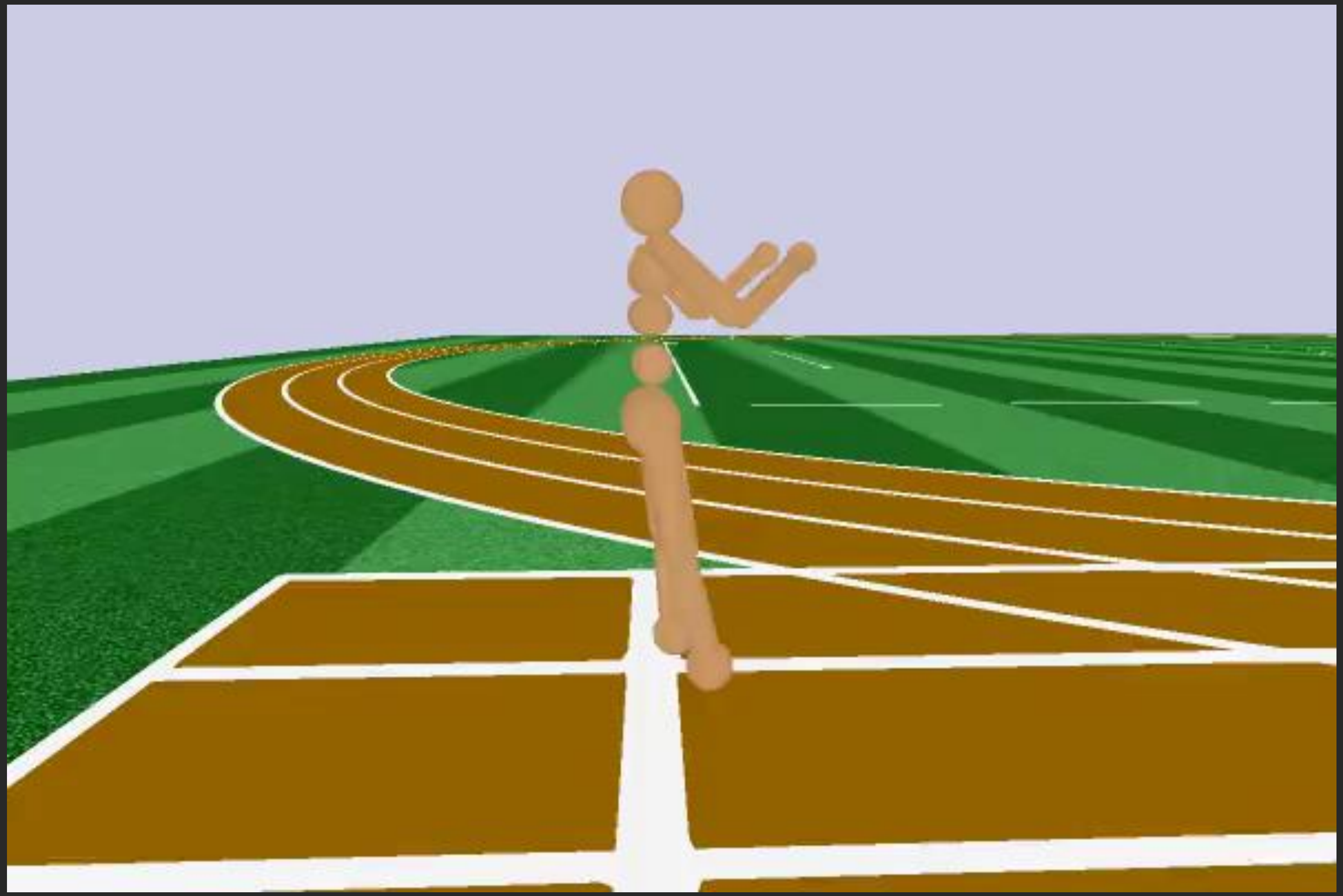
Q-Learning

Q-table

	A1	A2	A3
state1	..	..	..
state2	..	..	..
state3	..	..	..
state4	..	..	..

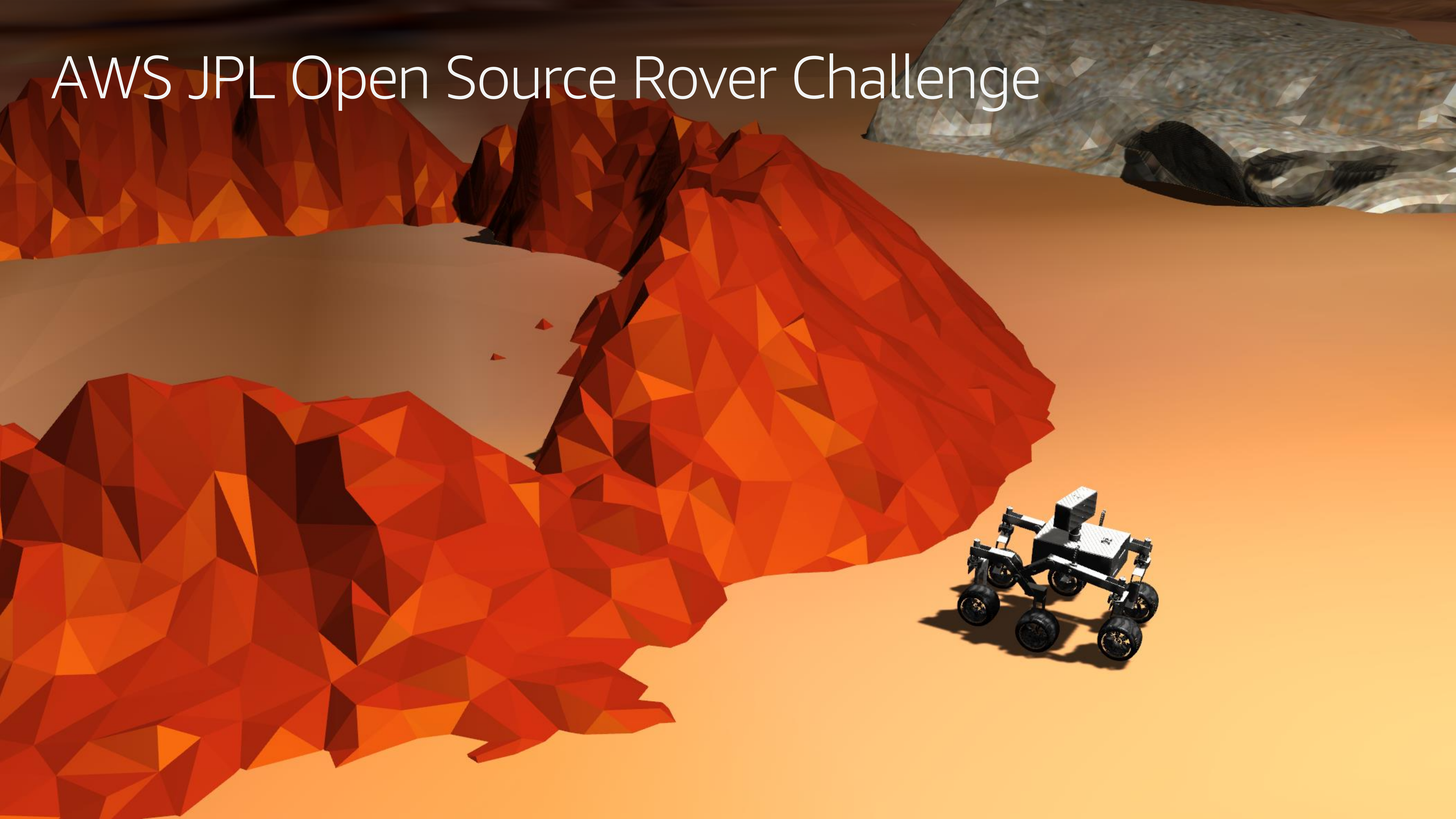


What about more complex environments?

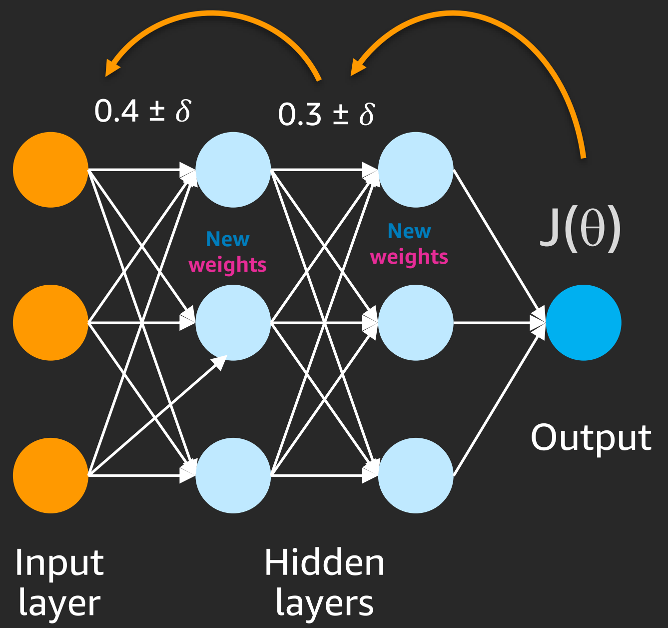


With complex or continuous state action spaces…

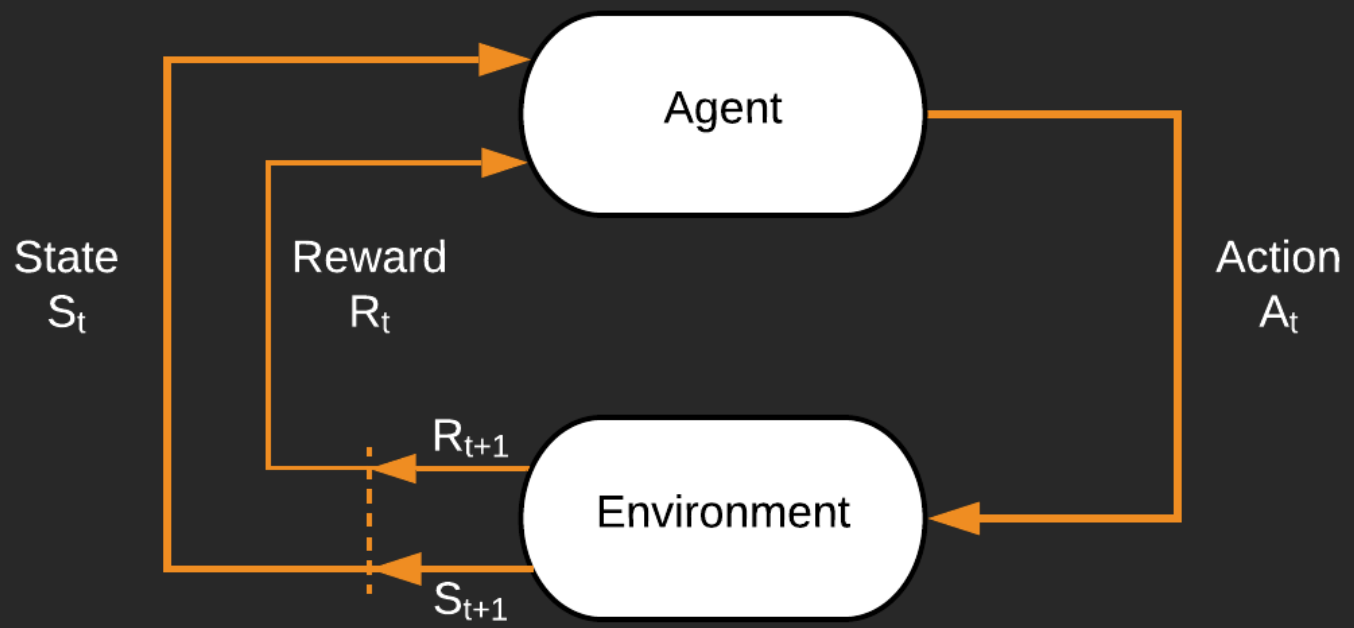
AWS JPL Open Source Rover Challenge



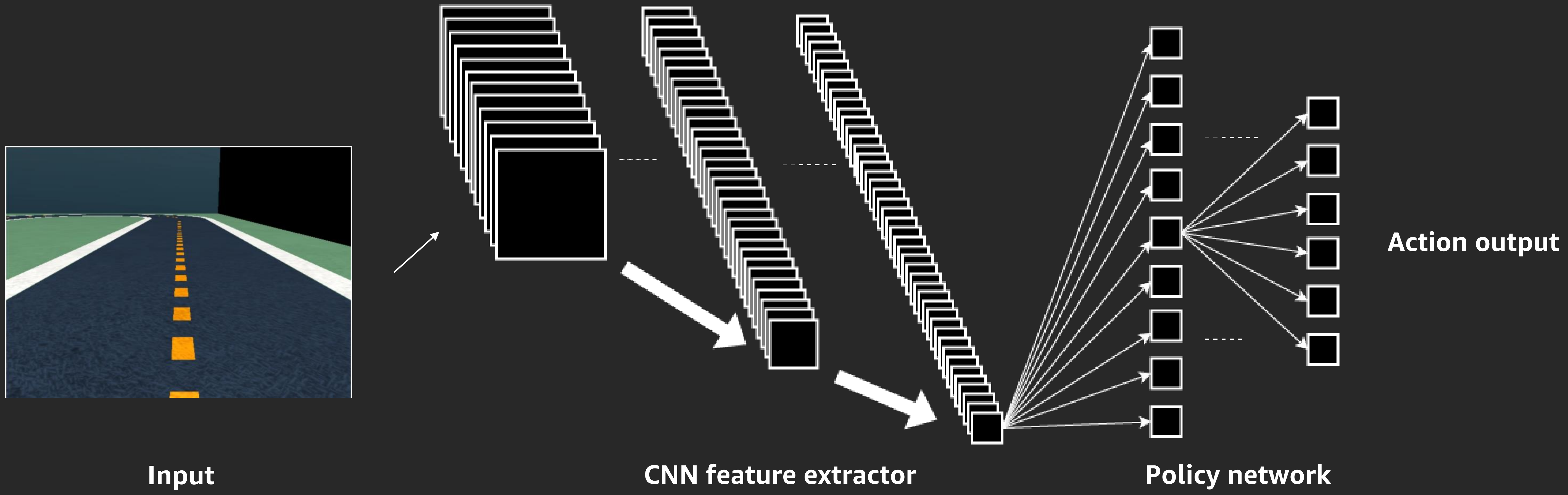
Deep RL



Use DNN(s) to approximate policy and value



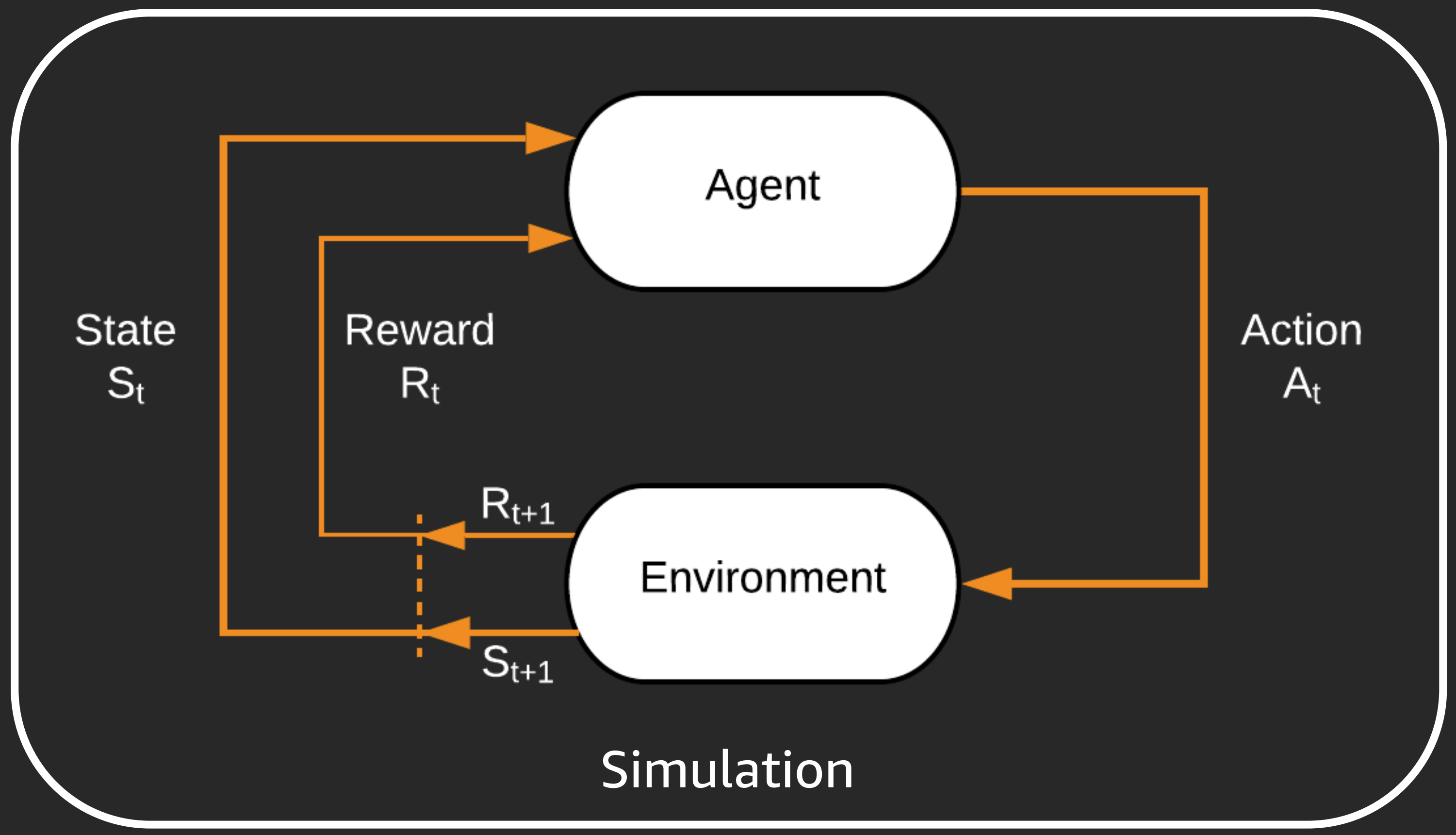
Deep RL - example



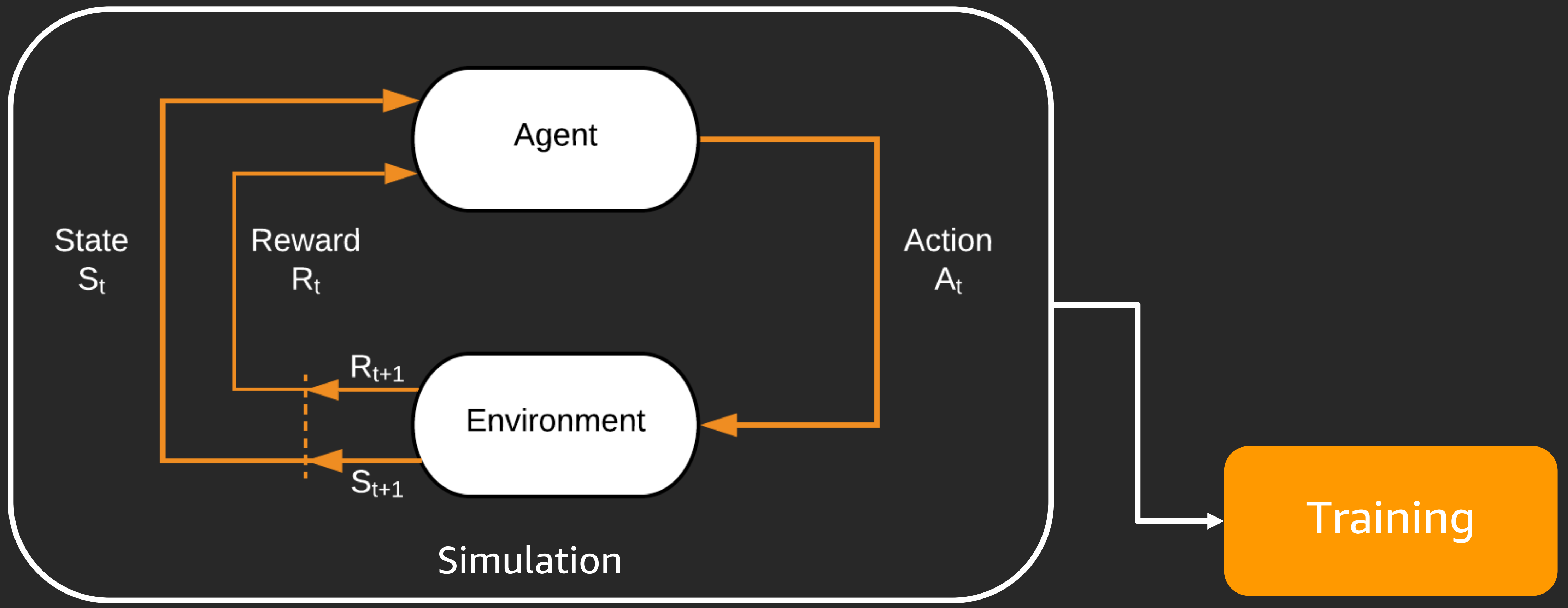
How to build an RL model

(with minimum pain)

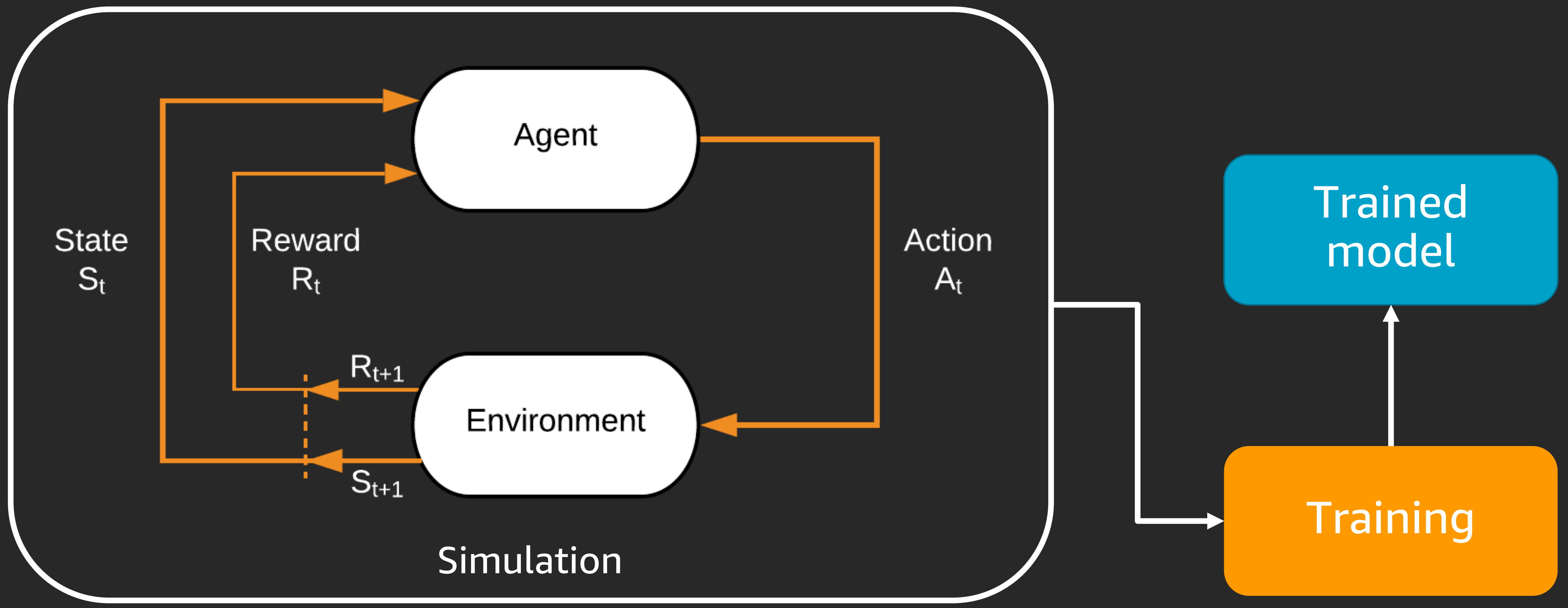
Explore / Capture / Train



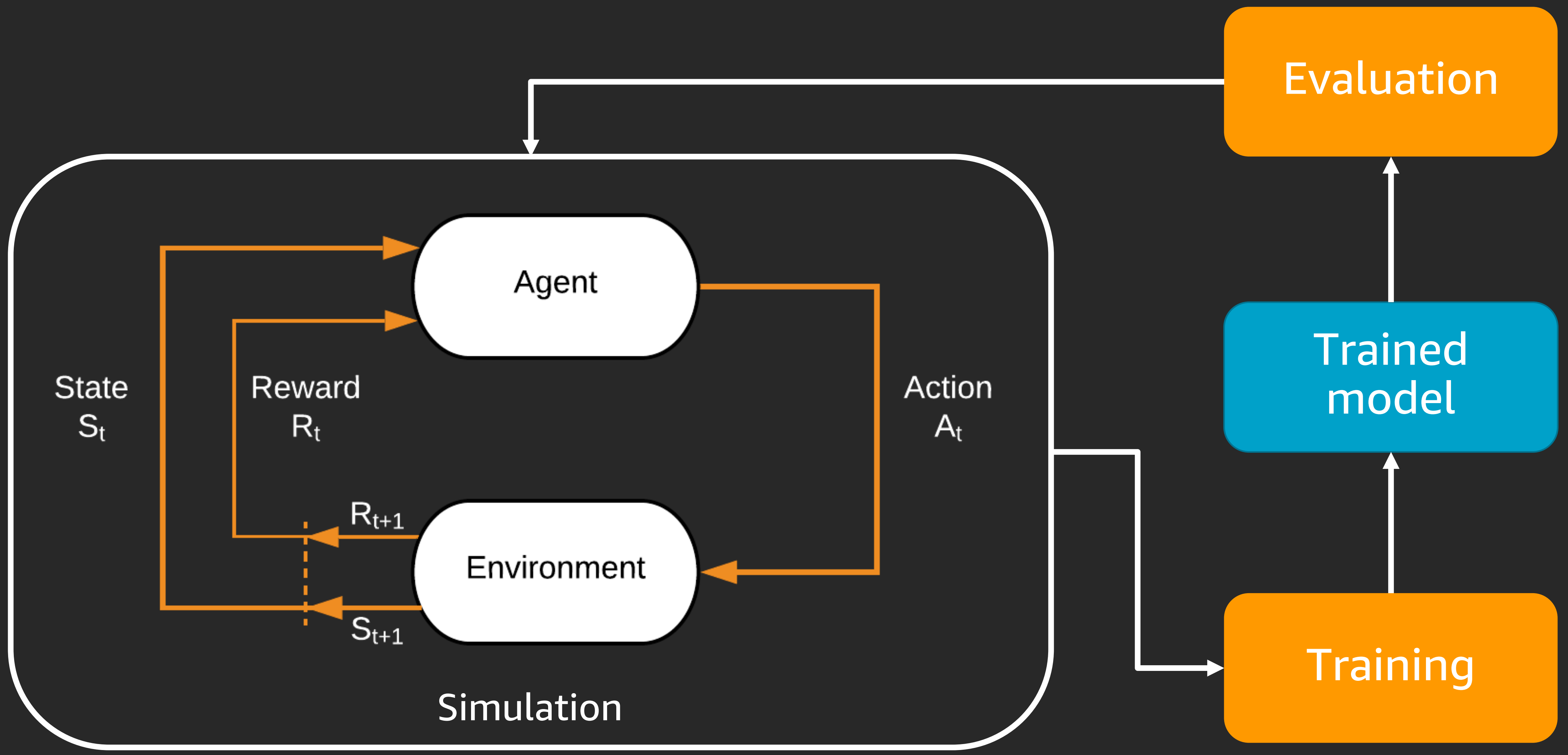
Explore / Capture / Train



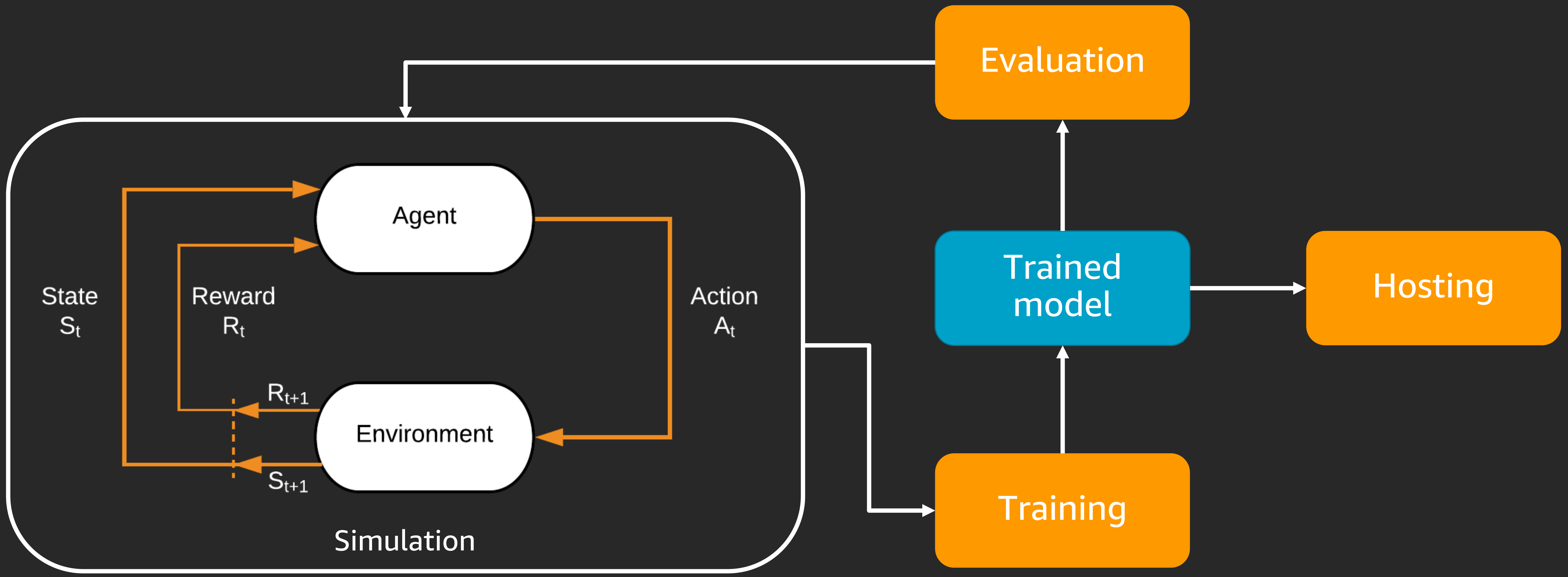
Explore / Capture / Train



Explore / Capture / Train



Explore / Capture / Train



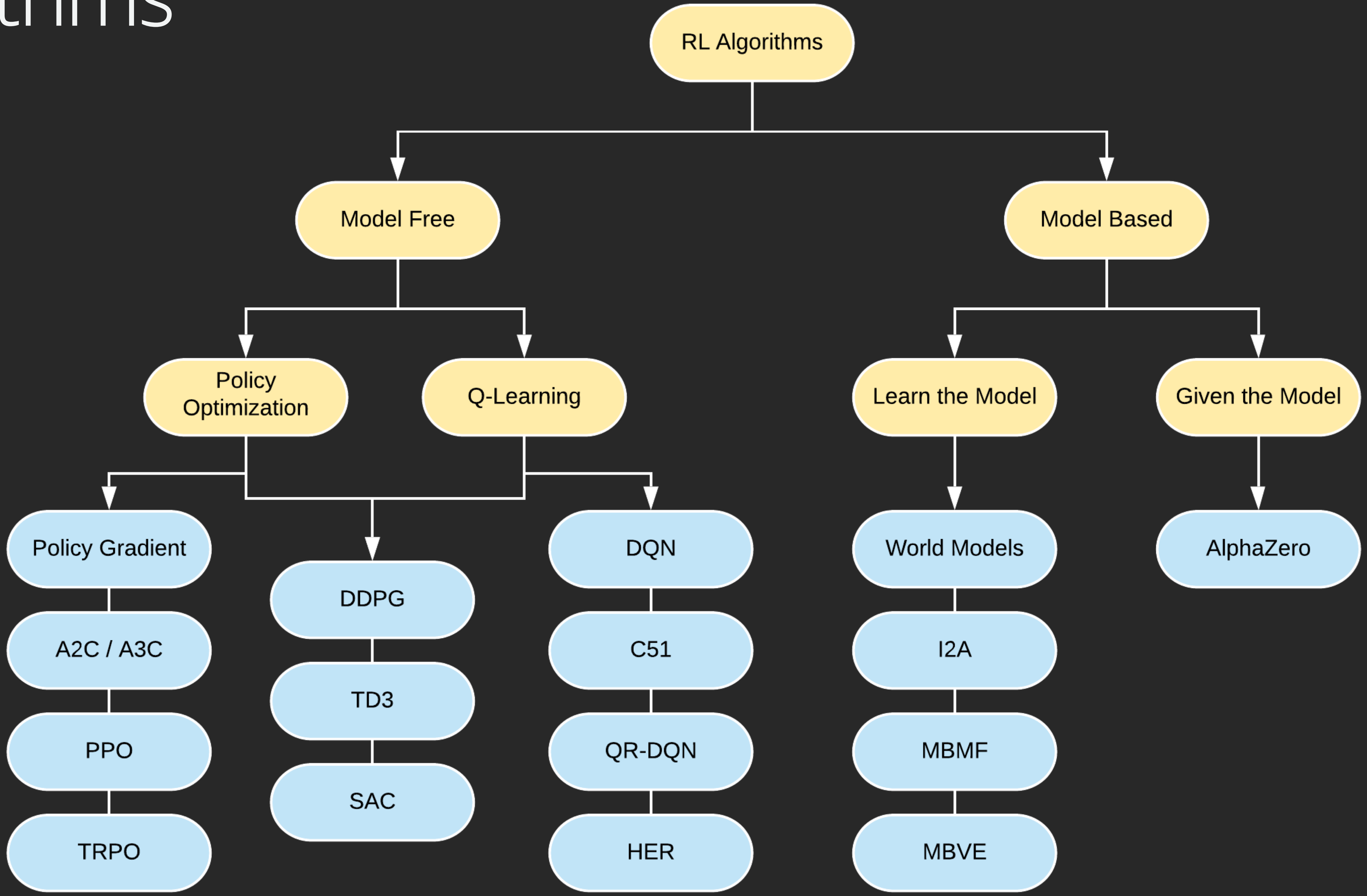
Steps

1. Select the algorithm

2. Create and setup the environment / simulation
   a) Define the goal(s) / reward(s)

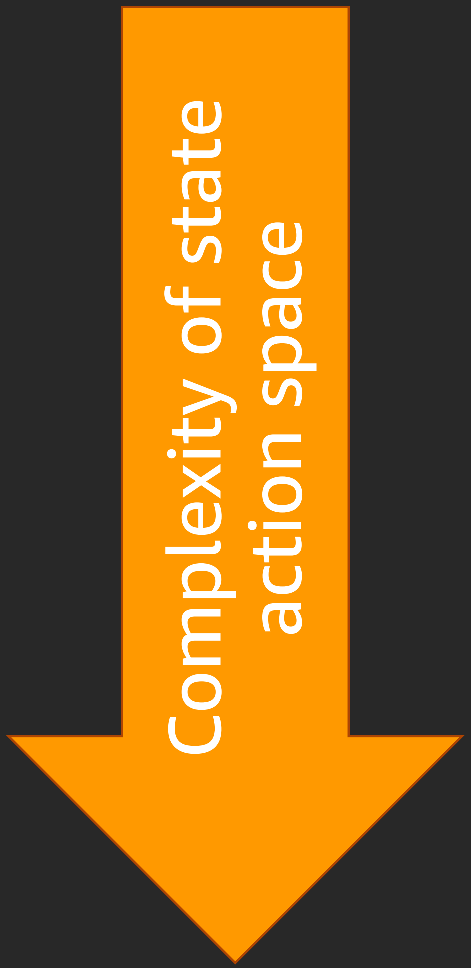
3. Loop: Train the model
   a) Collect trajectories / episodes (data, explore / exploit) and calculate rewards
   b) Train the functions

RL Algorithms



https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

RL Algorithms



Model based – learn a model of the environment

Quality based – learn the value of an action from a state

Policy based – learn the best actions to take

Combo – policy and value / advantage e.g. A2C / PPO

Roll your own

Step 1: Create an Instance—Amazon Deep Learning AMI

https://aws.amazon.com/amazon-ai/amis/

Step 2: Install OpenAI Gym

https://openai.com/

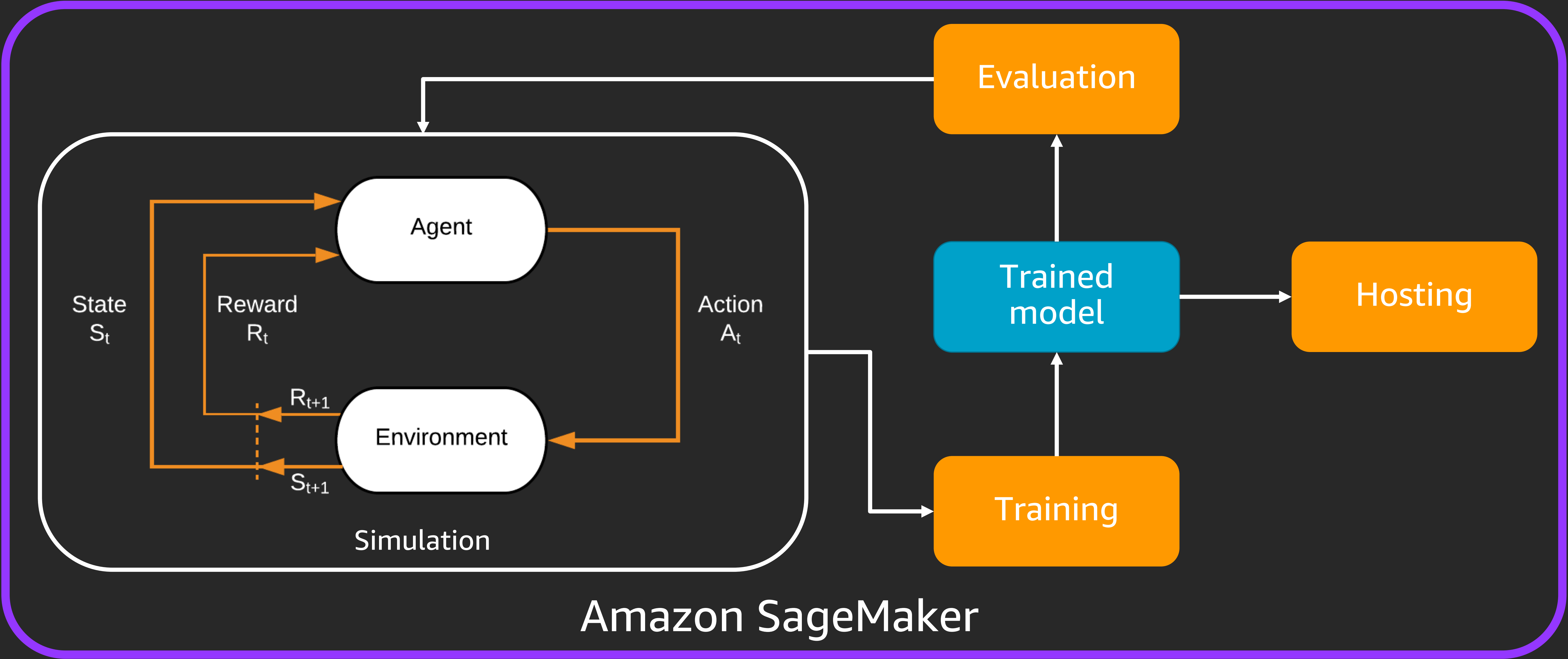
Step 3: Copy MXNet DQN Notebook from Github

https://github.com/zackchase/mxnet-the-straight-dope/

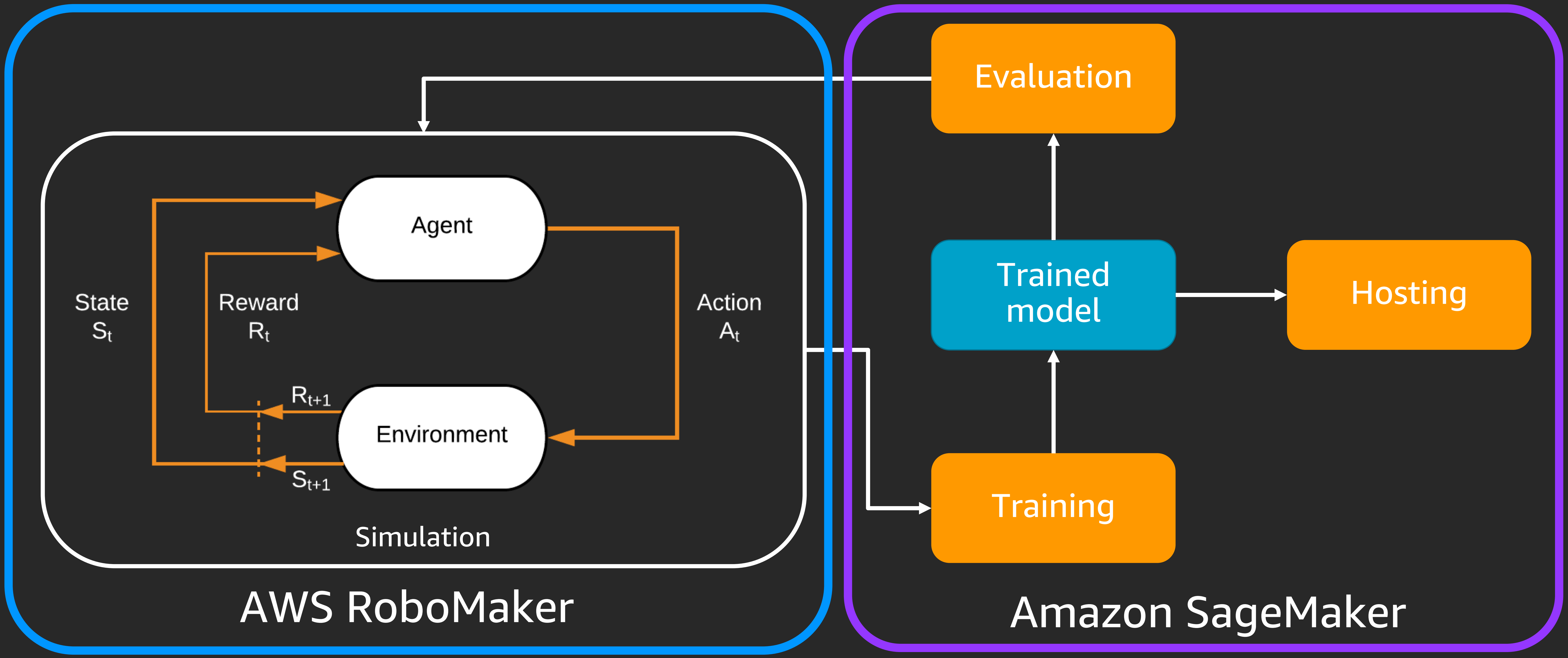
Step 4: Create a SSH tunnel and start Jupyter Notebook

https://www.youtube.com/watch?v=R6yex9kbt50

Amazon SageMaker: Training with custom and open source simulators

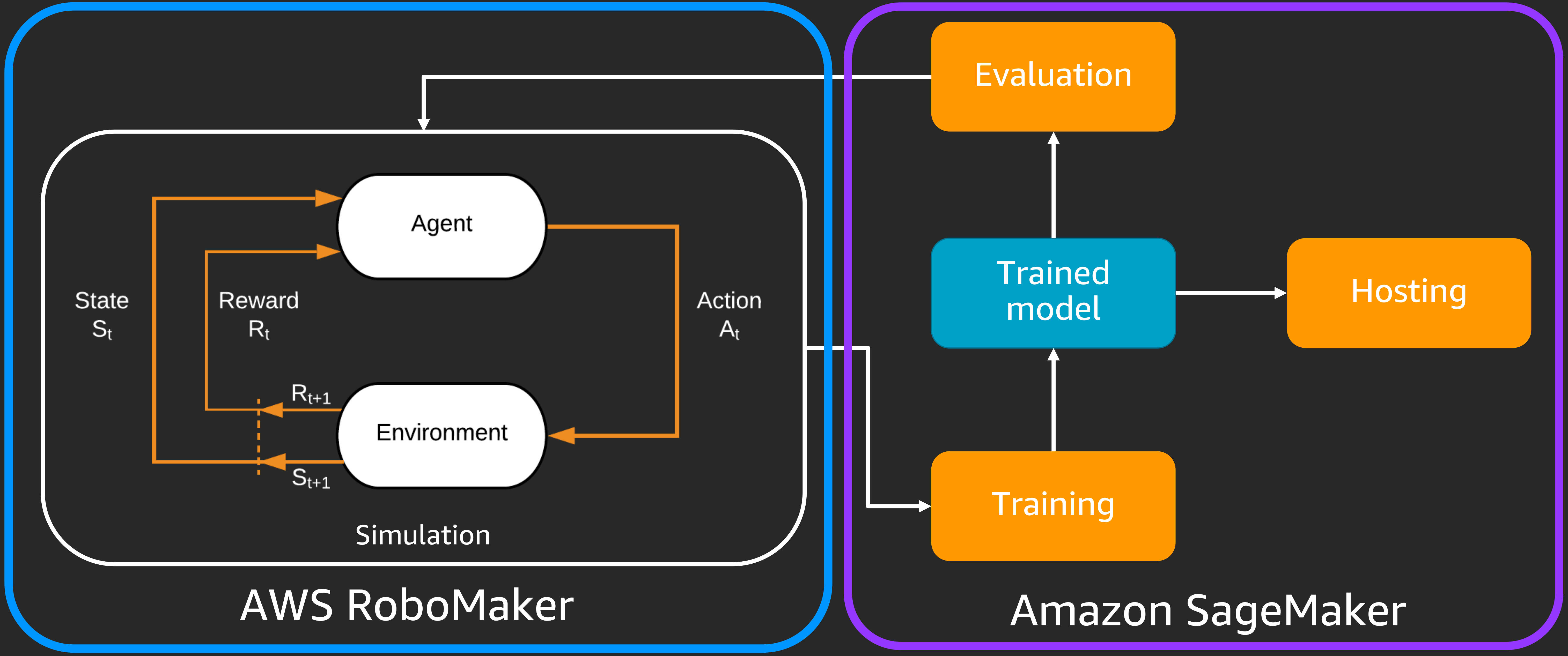


Amazon SageMaker: Training with remote simulation



Amazon SageMaker: Training with remote simulation

AWS DeepRacer



What you’ll need

1. RL Environment
2. RL Toolkit
3. Deep learning framework

Amazon SageMaker Provides

RL Environments

- AI Gym
- Open Source
- Commercial
- Custom
- Remote Simulation

RL Toolkits

- Ray RL Lib
- Intel Coach
- BYO

Deep Learning Frameworks

- TensorFlow
- MXNet
- BYO

Amazon SageMaker Provides

End to end examples

- Robotics
- Industrial control
- HVAC
- Autonomous vehicles
- Remote simulation
- Operations
- Finance
- Games
- NLP
- Recommendations

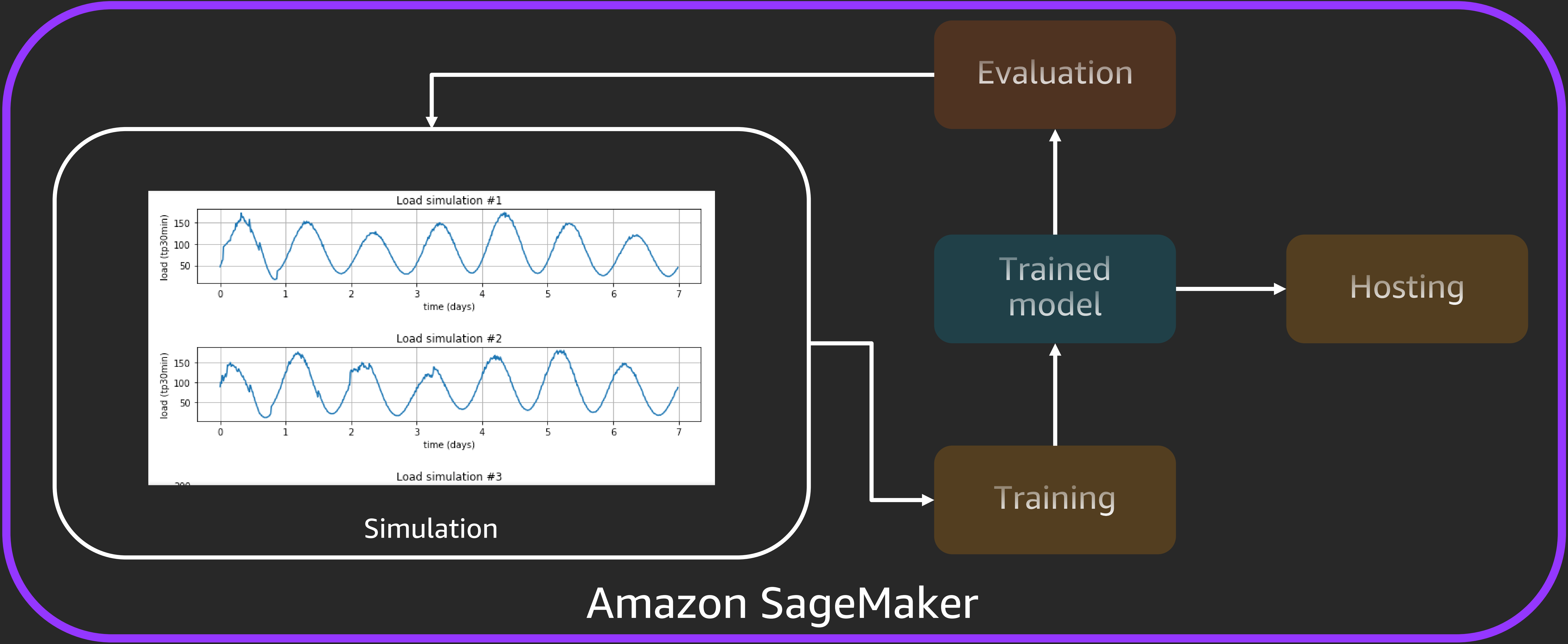
Setup and kick off your RL with one line of code*

```
estimator = RLEstimator (source_dir='src’,
        entry_point="train-coach.py", dependencies=["common/sagemaker_rl"],
        toolkit=RLToolkit.COACH, toolkit_version='0.11.0’,
        framework=RLFramework.MXNET, role=role,
        train_instance_count=1, train_instance_type=instance_type,
        output_path=s3_output_path, base_job_name=job_name_prefix,
        hyperparameters = {
                "RLCOACH_PRESET" : "preset-portfolio-management-clippedppo",
                "rl.agent_params.algorithm.discount": 0.9,
                "rl.evaluation_steps:EnvironmentEpisodes": 5
                }
        )
```

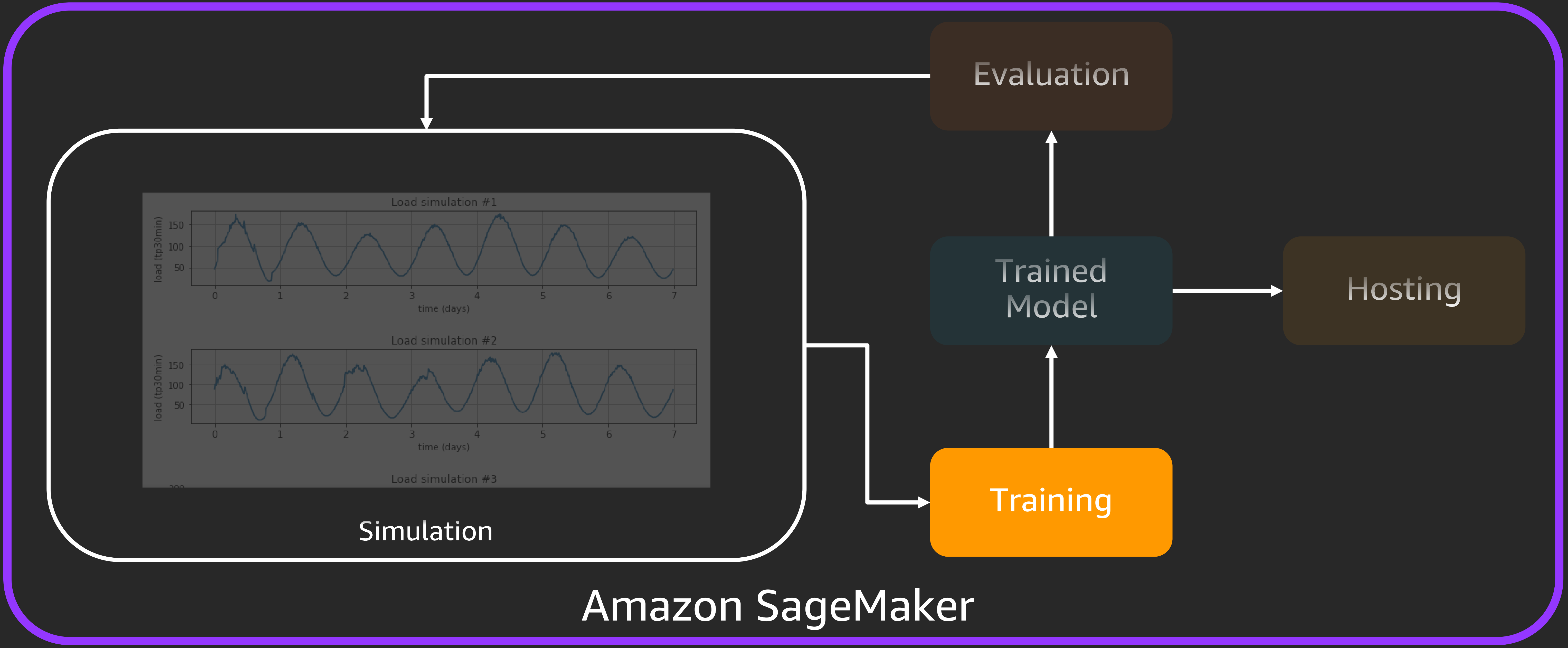
```
estimator = RLEstimator (source_dir='src',
        entry_point="train-coach.py", dependencies=["common/sagemaker_rl"],
        toolkit=RLToolkit.COACH, toolkit_version='0.11.0',
        framework=RLFramework.MXNET, role=role,
        train_instance_count=1, train_instance_type=instance_type,
        output_path=s3_output_path, base_job_name=job_name_prefix,
        hyperparameters = {
                "RLCOACH_PRESET" : "preset-portfolio-management-clippedppo",
                "rl.agent_params.algorithm.discount": 0.9,
                "rl.evaluation_steps:EnvironmentEpisodes": 5
                }
        )
```

Demo

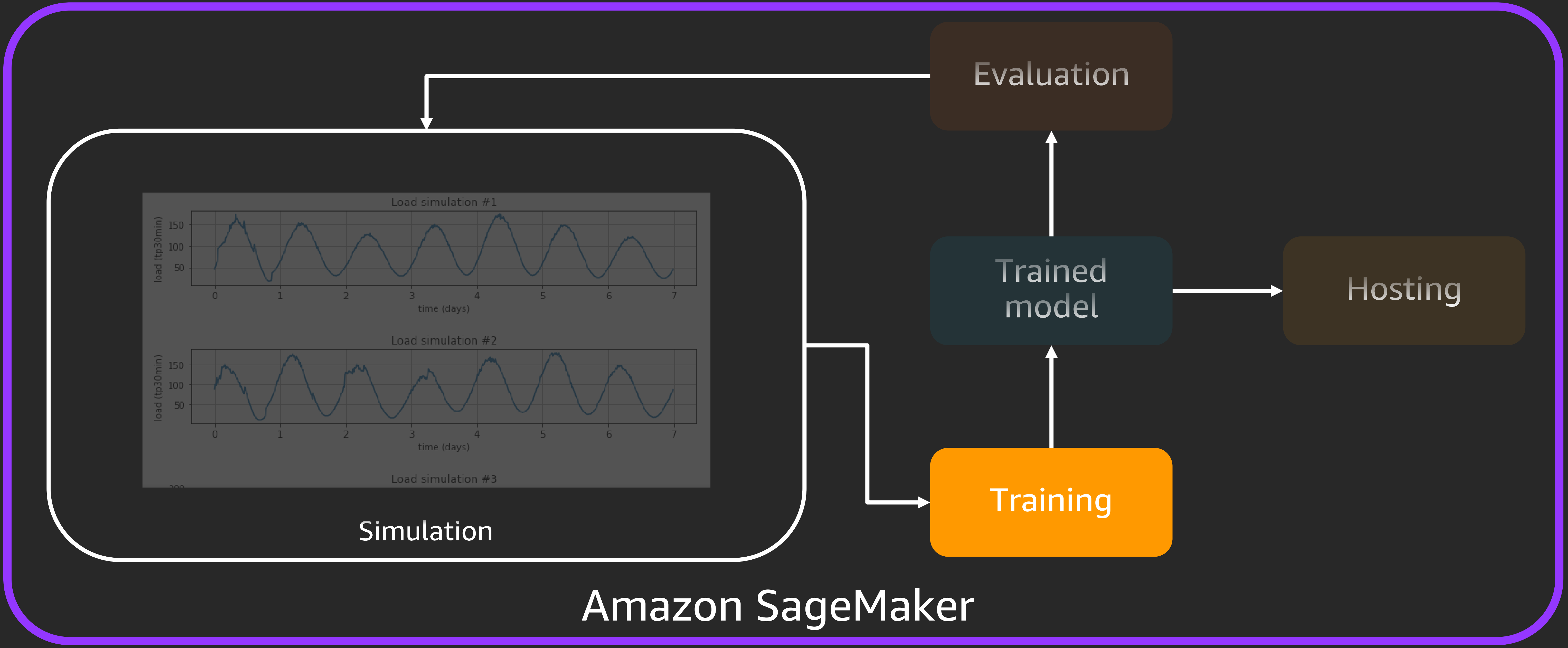
Environment/ Simulation



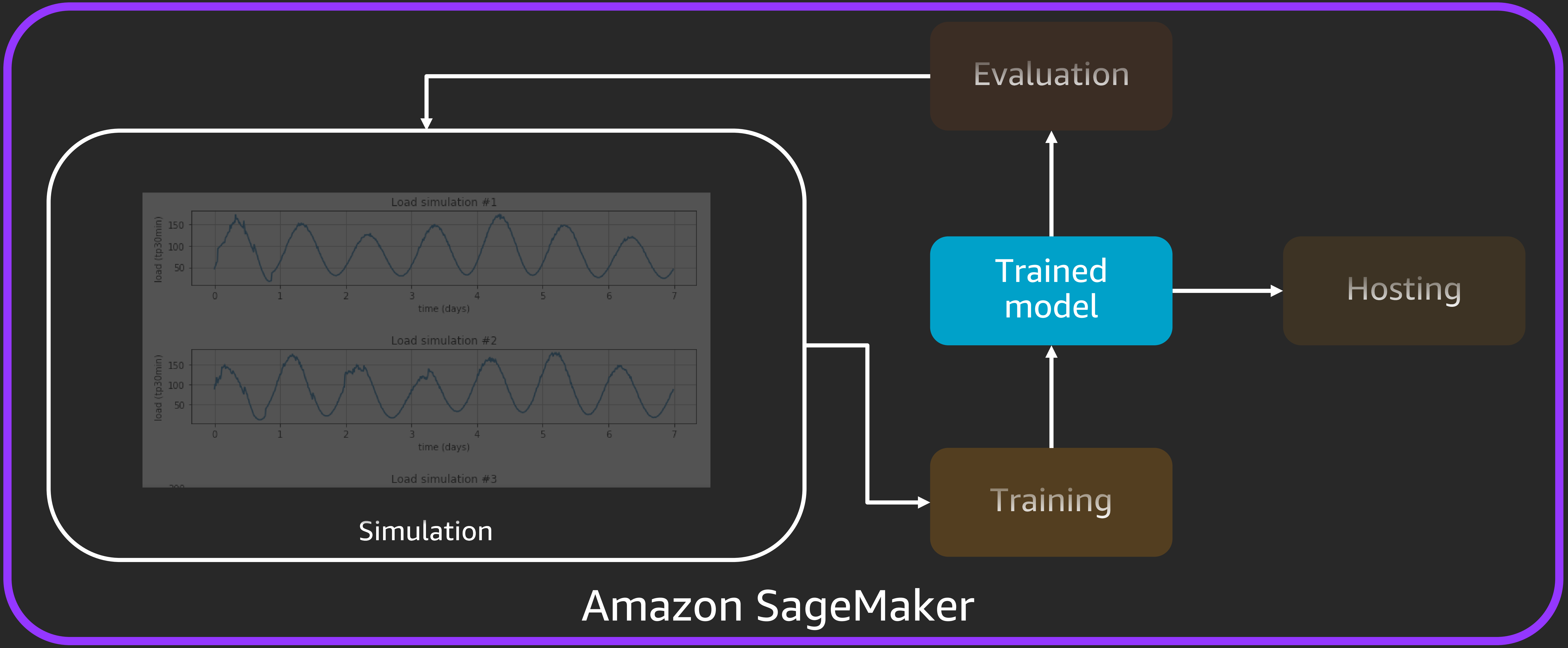
Training algorithm



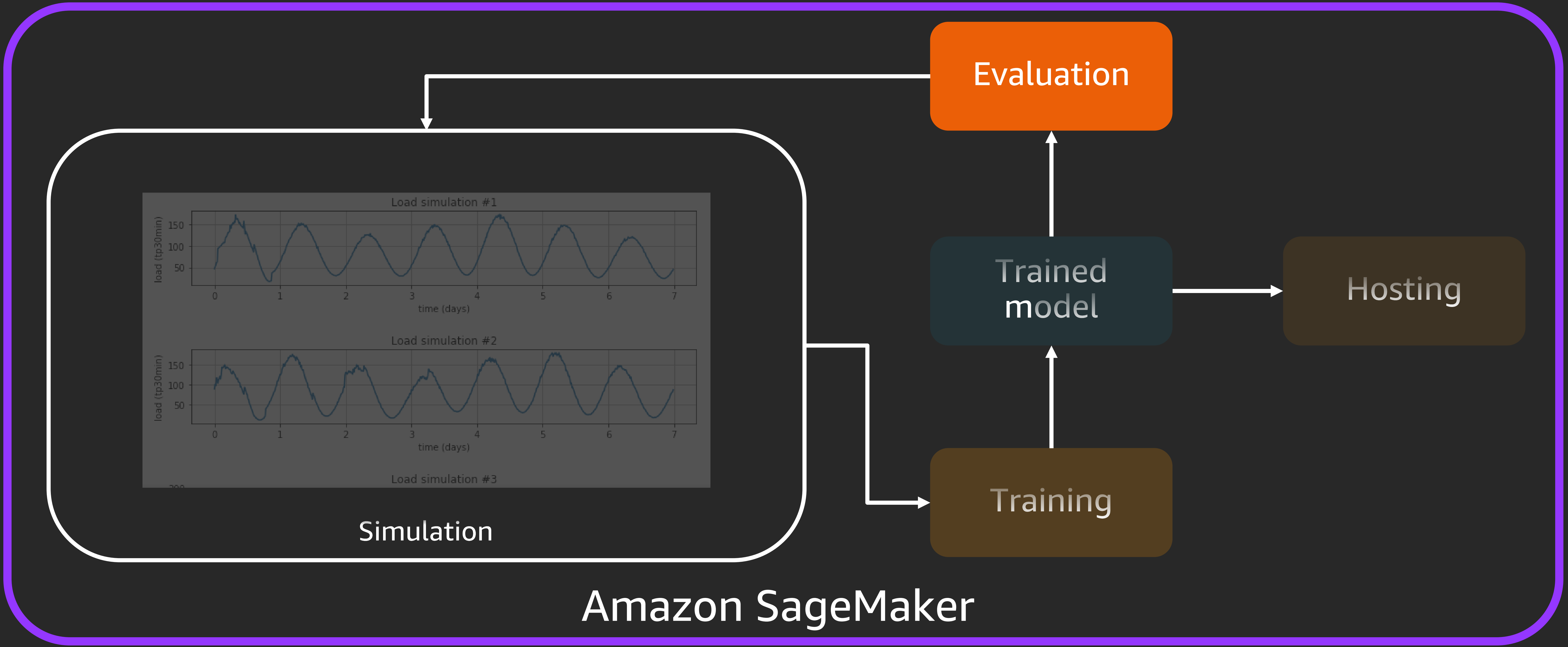
Training the model



Trained model

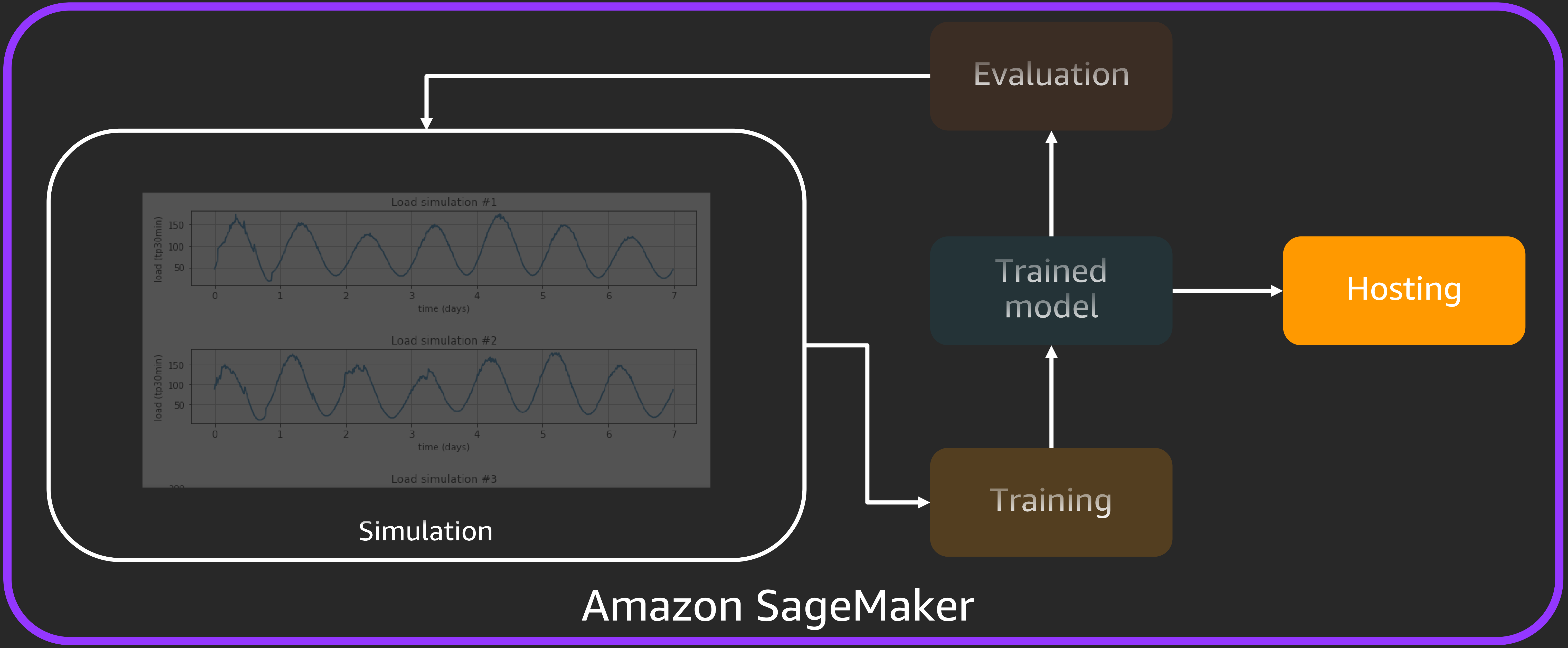


Evaluation



Evaluation
Load simulation #1
load (tp30min)
150
100
50
0
1
2
3
4
5
6
7
time (days)
Load simulation #2
load (tp30min)
150
100
50
0
1
2
3
4
5
6
7
time (days)
Load simulation #3
Simulation
Trained model
Hosting
Training
Amazon SageMaker

Hosting


Evaluation
Load simulation #1
150
100
50
load (tp30min)
0
1
2
3
4
5
6
7
time (days)
Load simulation #2
Load simulation #3
Simulation
Trained model
Hosting
Training
Amazon SageMaker

When to use RL

AWS SUMMIT ONLINE

RL Requires

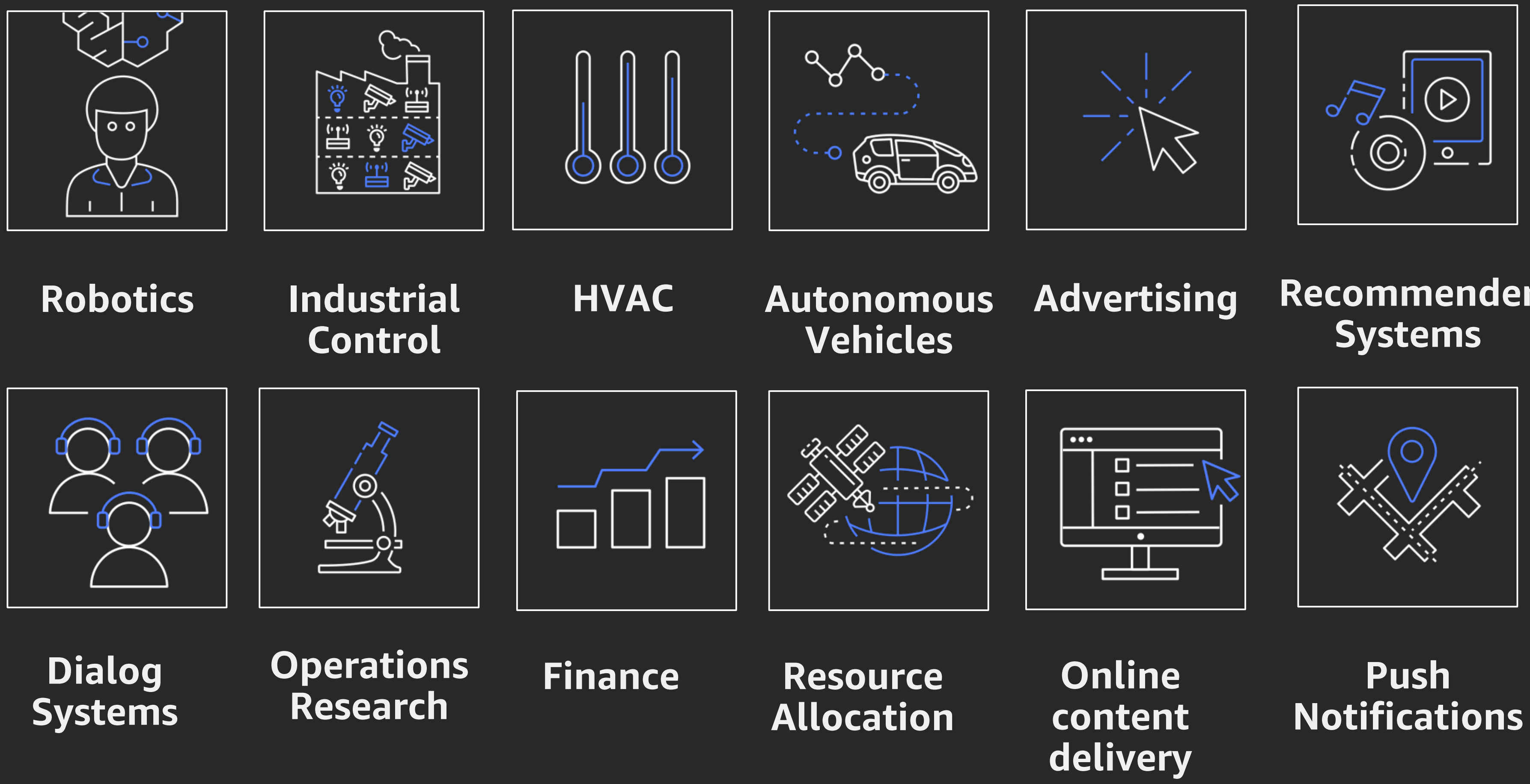
Problem type:

- Trial and error
- Definable rewards / goal
- MDP
- Control

Simulation

Algorithm

Application examples



Amazon SageMaker RL

Reinforcement learning for every developer and data scientist

Fully managed

Broad support for frameworks

Broad support for simulation environments

Key features

2D & 3D physics environments and OpenGym support

Support Amazon Sumerian, AWS RoboMaker and the open source Robotics Operating System (ROS) project

Example notebooks and tutorials

RL: Tips for success

Challenges

Sample inefficiency

Sparse rewards

Other challenges

High-dimensional continuous state and action spaces

Learning on the real system from limited samples

Batch off-line and off-policy training

Satisfying safety constraints

Partial observability and non-stationarity

Unspecified and multi-objective reward functions

Explainability

Real-time inference

System delays

https://openreview.net/pdf?id=S1xtR52NjN

Tips

Use the cloud… No really

Tips

Use the cloud... No really

Use Amazon SageMaker

- Examples
- Setup / inclusions
- Experiment management
- HPO

Tips

Use the cloud… No really

Use Amazon SageMaker

- Examples
- Setup / inclusions
- Experiment management
- HPO

Simulation FTW

Simulate as close to real as you can

Domain randomisation

Tips

Use the cloud… No really

Use Amazon SageMaker

- Examples
- Setup / inclusions
- Experiment management
- HPO

Simulation FTW

Simulate as close to real as you can

Domain randomisation

Where’s your bottleneck?

- Simulation → Parallel simulation
- Training → Distributed training

Tips

Use the cloud… No really

Use Amazon SageMaker

- Examples
- Setup / inclusions
- Experiment management
- HPO

Simulation FTW

Simulate as close to real as you can

Domain randomisation

Where's your bottleneck?

- Simulation → Parallel simulation
- Training → Distributed training

Most problems are not true MDP but partial

- Careful design of environment

Tips

Use the cloud… No really

Use Amazon SageMaker

- Examples
- Setup / inclusions
- Experiment management
- HPO

Simulation FTW

Simulate as close to real as you can

Domain randomisation

Where's your bottleneck?

- Simulation → Parallel simulation
- Training → Distributed training

Most problems are not true MDP but partial

- Careful design of environment

Improve state awareness for your agent:

- RNN, CNN over time, more input

Thank you!

Ben Thurgood

btgood@amazon.com

aws SUMMIT ONLINE