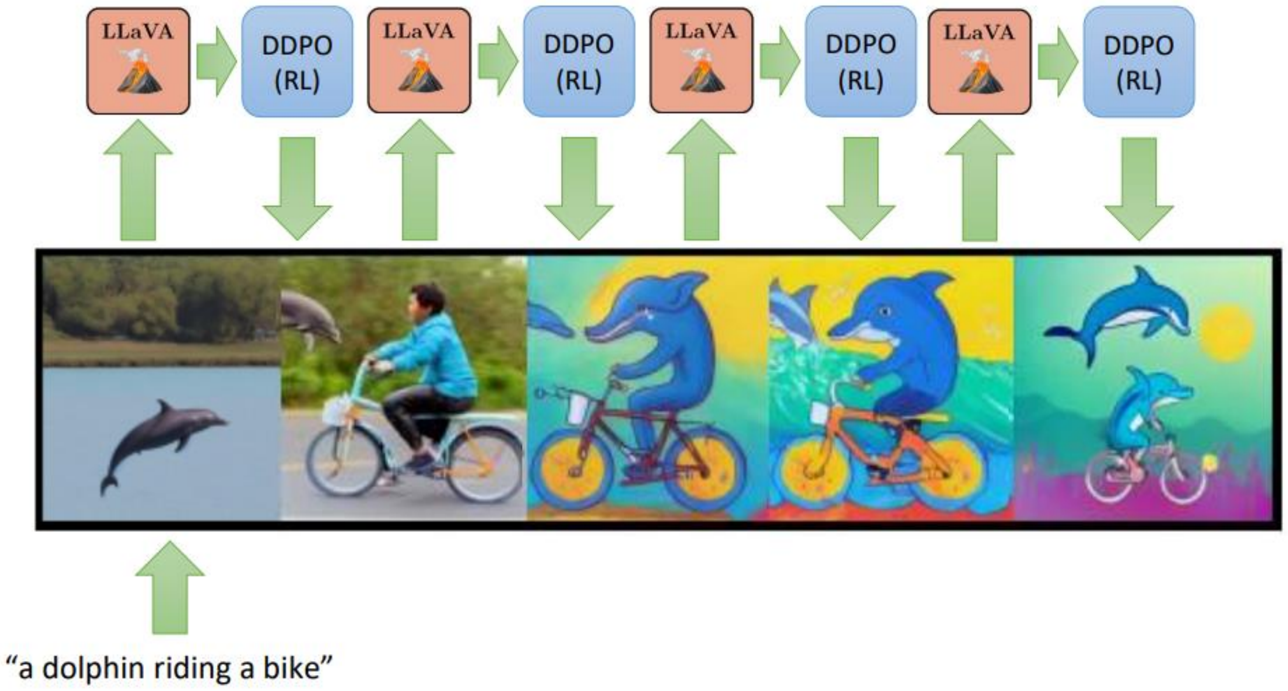
Introduction To Reinforcement Learning



Lancaster AI Reading Group – 22nd January 2025

Based on CS285 Berkeley Course

Machine Learning

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

By Tom Mitchell (Prof. Carnegie Mellon University):

Experience

Experience

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Experience

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

Experience

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

That is not always the case!

The 3 ML Settings

Supervised Learning

Unsupervised Learning

Reinforcement Learning

The 3 ML Settings

Supervised Learning	Unsupervised Learning	Reinforcement Learning
$\{(x_1, y_1), \ldots, (x_N, y_N)\}$	$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$	?
Learn p(y|x)	Learn p(x)	

The 3 ML Settings

Supervised Learning

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Learn p(y|x)

Unsupervised Learning

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

Learn p(x)

Reinforcement Learning

- Environment

The 3 ML Settings

Supervised Learning

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Learn p(y|x)

Unsupervised Learning

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

Learn p(x)

Reinforcement Learning

- Environment
- Agent

The 3 ML Settings

Supervised Learning

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Learn p(y|x)

Unsupervised Learning

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

Learn p(x)

Reinforcement Learning

- Environment
- Agent
- Actions

The 3 ML Settings

Supervised Learning

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Learn p(y|x)

Unsupervised Learning

$$\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$$

Learn p(x)

Reinforcement Learning

- Environment
- Agent
- Actions
- Rewards

The 3 ML Settings

Supervised Learning

$$\{(x_1, y_1), \ldots, (x_N,\ y_N)\}$$

Learn p(y|x)

Unsupervised Learning

$$\{x_1, \ldots, x_N\}$$

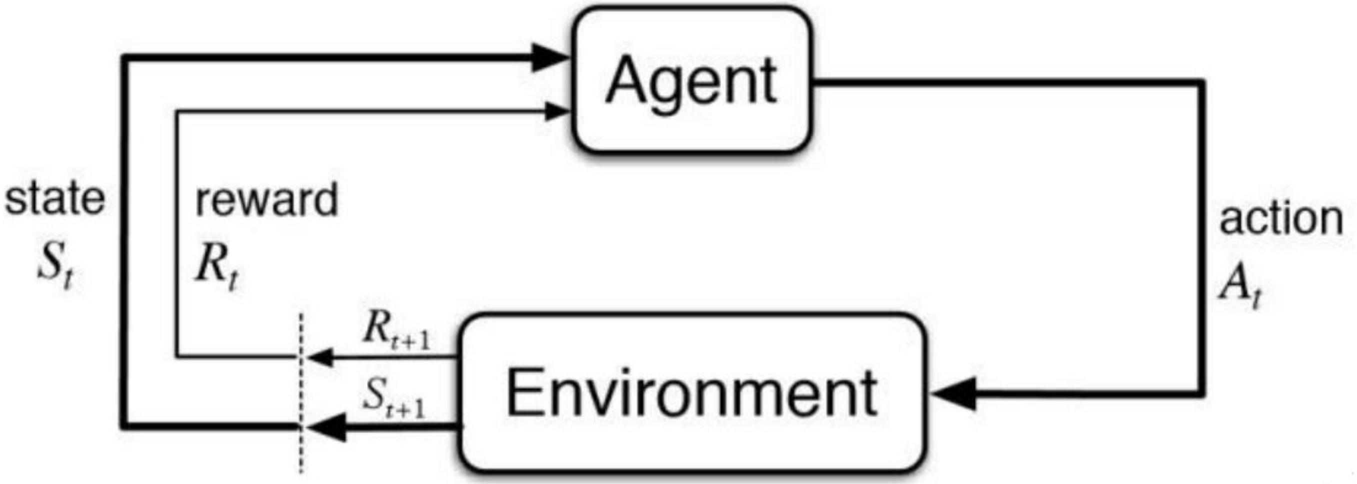
Learn p(x)

Reinforcement Learning

- Environment
- Agent
- Actions
- Rewards

How to learn a policy that maximise the cumulative reward?

The RL Framework

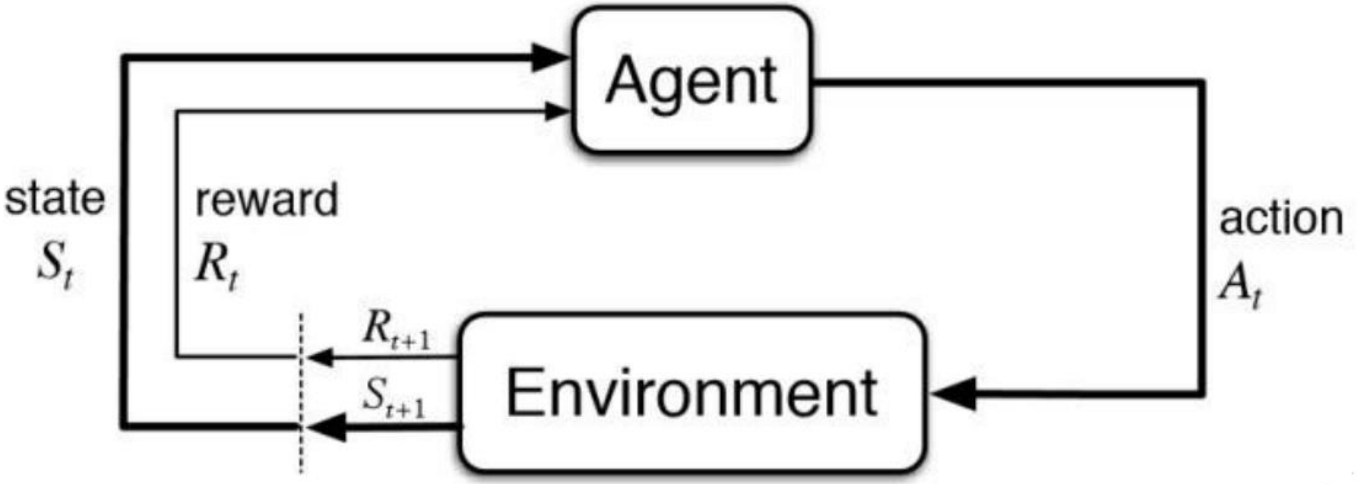


Reinforcement Learning

- Environment
- Agent
- Actions
- Rewards

How to learn a policy that maximise the cumulative reward?

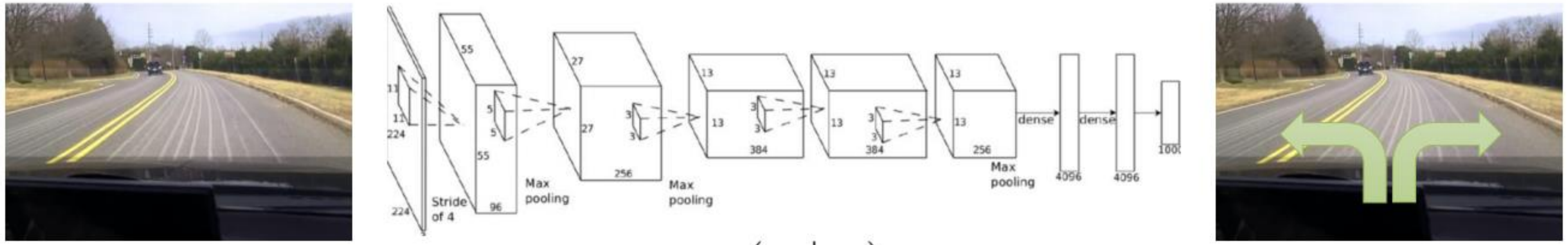
The RL Framework



Reinforcement Learning

- Environment
- Agent
- Actions
- Rewards

How to learn a policy that maximise the cumulative reward?



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

RL Examples

RL Examples

Actions: muscle contractions
Observations: sight, smell
Rewards: food

Actions: motor current or torque
Observations: camera images
Rewards: task success measure (e.g., running speed)

Actions: what to purchase
Observations: inventory levels
Rewards: profit

What about the environments?

What about the environments?

OpenAI Gym

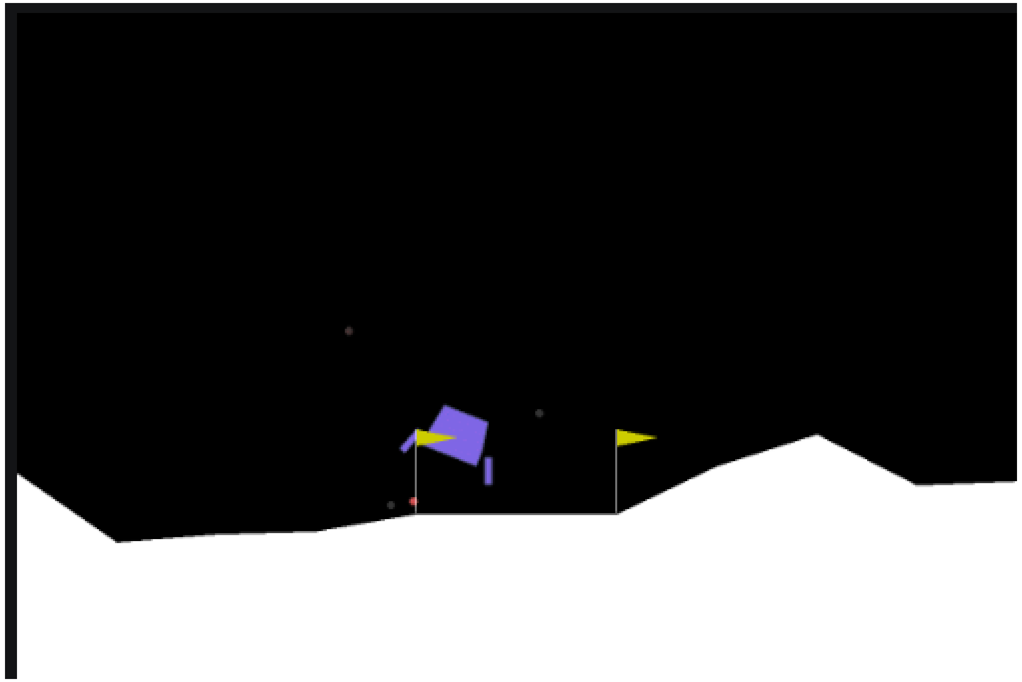
BRAX

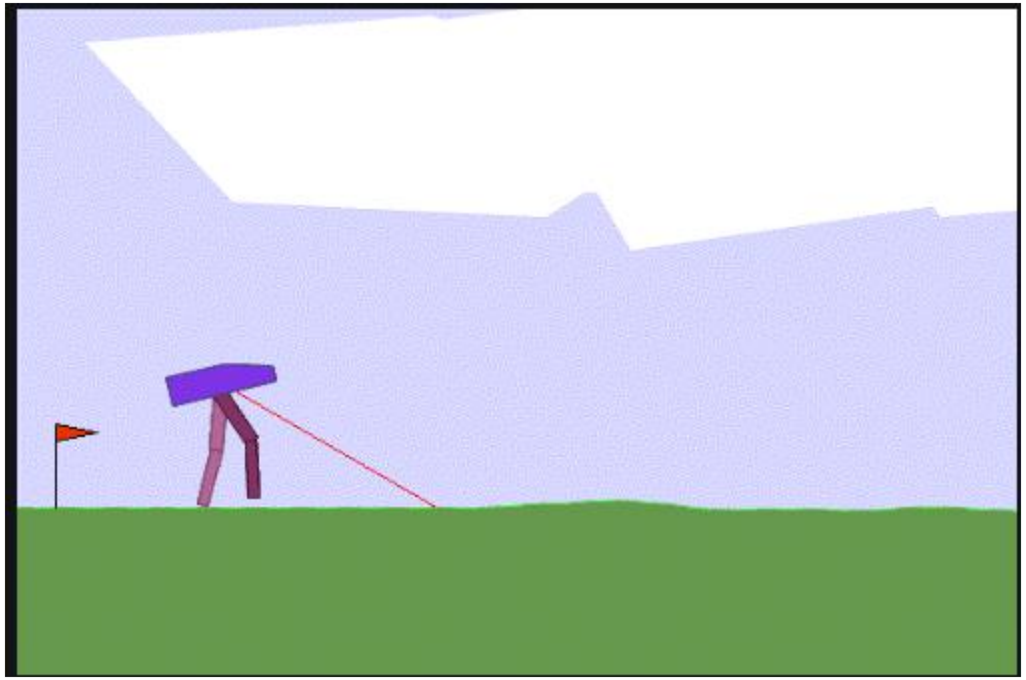
What about the environments?

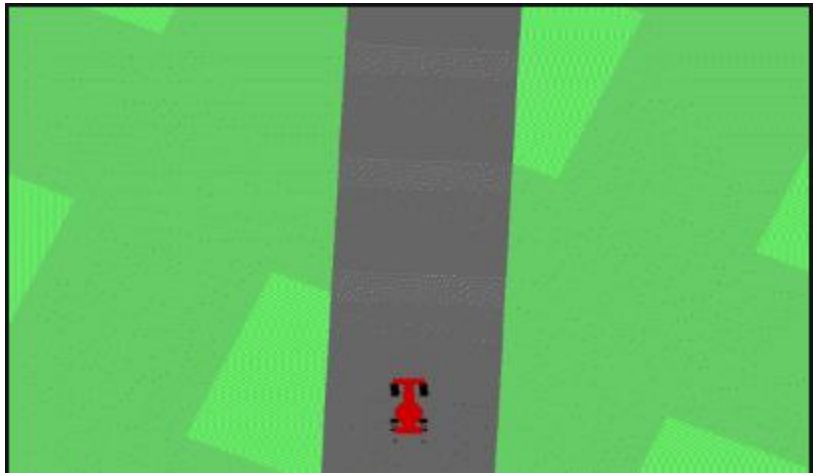
OpenAI Gym

```
import gym

# Create an environment
env = gym.make("Ant-v4", new_step_api=True)
# Reset the environment to an initial state
obs = env.reset()
# Take a random action
action = env.action_space.sample()
# Update the environment according to the action
obs, reward, done, info = env.step(action)

env.close()
```

Example Environments







How to learn a “good” policy?

How to learn a “good” policy?

A simple RL algorithm: **REINFORCE**

How to learn a “good” policy?

A simple RL algorithm: **REINFORCE**

- Sample a trajectory from the current policy.

How to learn a “good” policy?

A simple RL algorithm: **REINFORCE**

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\tau \sim p_{\boldsymbol{\theta}}(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right] \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_{i,t} | \mathbf{s}_{i,t}) \right) \left(\sum_{i=1}^{T} r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

- Sample a trajectory from the current policy.
- Estimate the gradient of the objective (use backpropagation in the case of Neural Networks).

How to learn a “good” policy?

A simple RL algorithm: **REINFORCE**

$$\nabla_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(s_t, a_t) \right] \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_{i,t} | s_{i,t}) \right) \left(\sum_{i=1}^{T} r(s_{i,t}, a_{i,t}) \right)$$

- Sample a trajectory from the current policy.
- Estimate the gradient of the objective (use backpropagation in the case of Neural Networks).
- Update the policy using a gradient based optimization algorithm.

The necessity of RL

- Some tasks are hard to get supervised pairs from (e.g. movements of a robot).

The necessity of RL

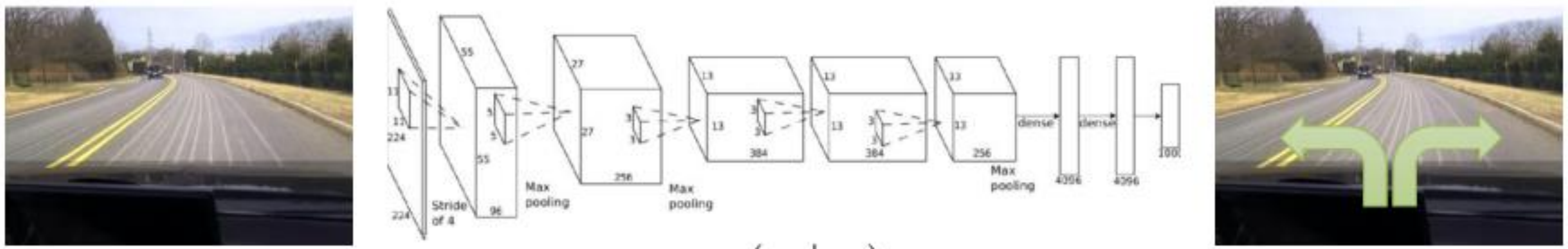
- Some tasks are hard to get supervised pairs from (e.g. movements of a robot).
- Supervised learning is not ideal for sequential decision making.

The necessity of RL

- Some tasks are hard to get supervised pairs from (e.g. movements of a robot).
- Supervised learning is not ideal for sequential decision making.
- RL allows for novel solutions.

Imitation Learning

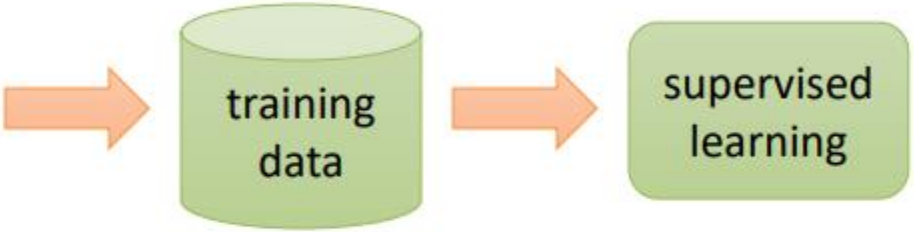
- Supervised learning is not ideal for sequential decision making.



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

$\mathbf{o}_t$

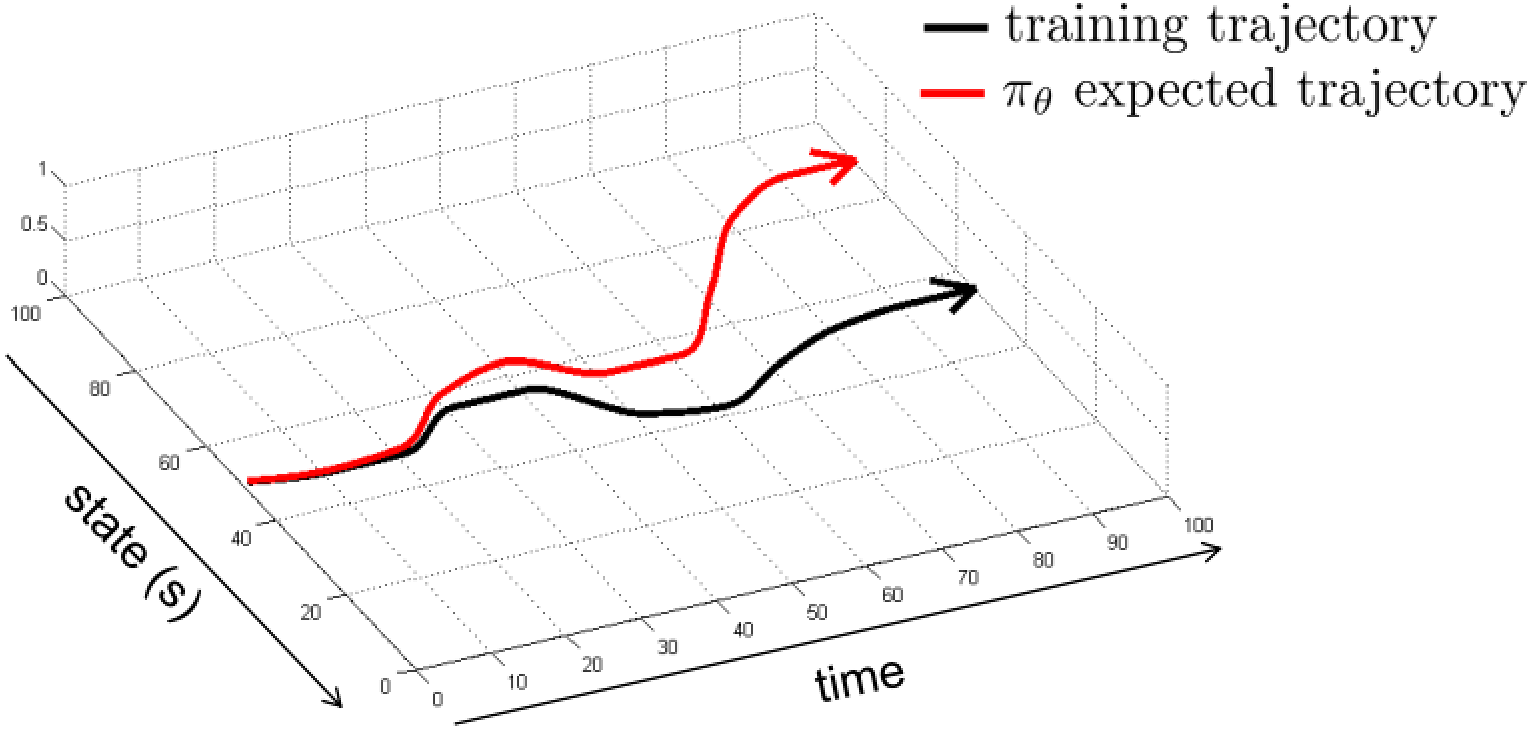
$\mathbf{a}_t$



$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$

Imitation Learning

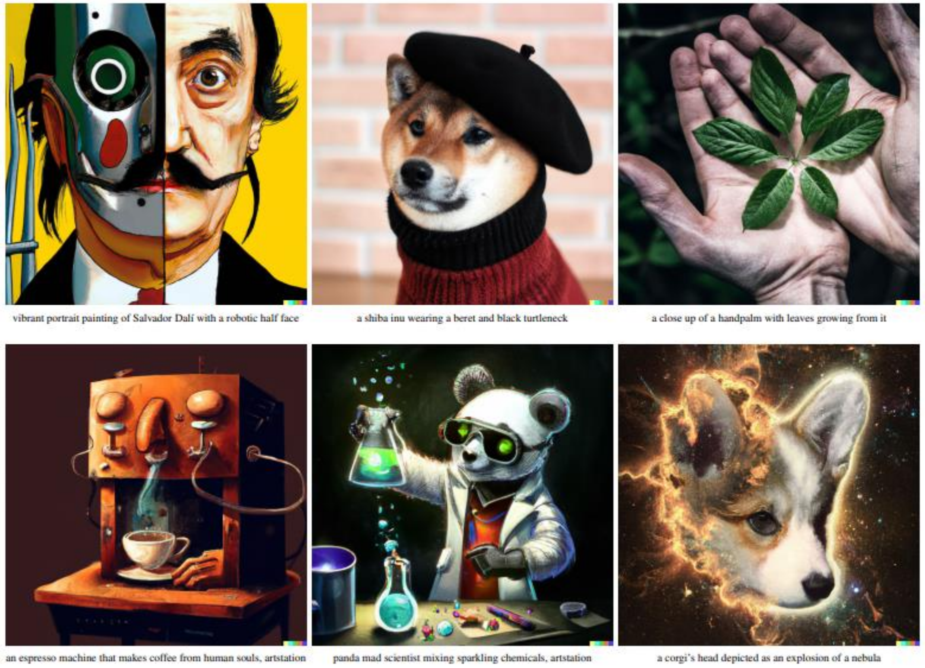
- Supervised learning is not ideal for sequential decision making.



Data is not iid!

Recent advances in DL vs RL

- RL allows for novel solutions.



ChatGPT

Recent advances in DL vs RL

- RL allows for novel solutions.

$p(x)$ $p(x|c)$ $p(y_n|y_{1:n-1})$

vibrant portrait painting of Salvador Dalí with a robotic half face
a shiba inu wearing a beret and black turtleneck
a close up of a handpalm with leaves growing from it
an espresso machine that makes coffee from human souls, artstation
panda mad scientist mixing sparkling chemicals, artstation
a corgi's head depicted as an explosion of a nebula

ChatGPT

Recent advances in DL vs RL

- RL allows for novel solutions.

Image/Video

$$p(x)$$

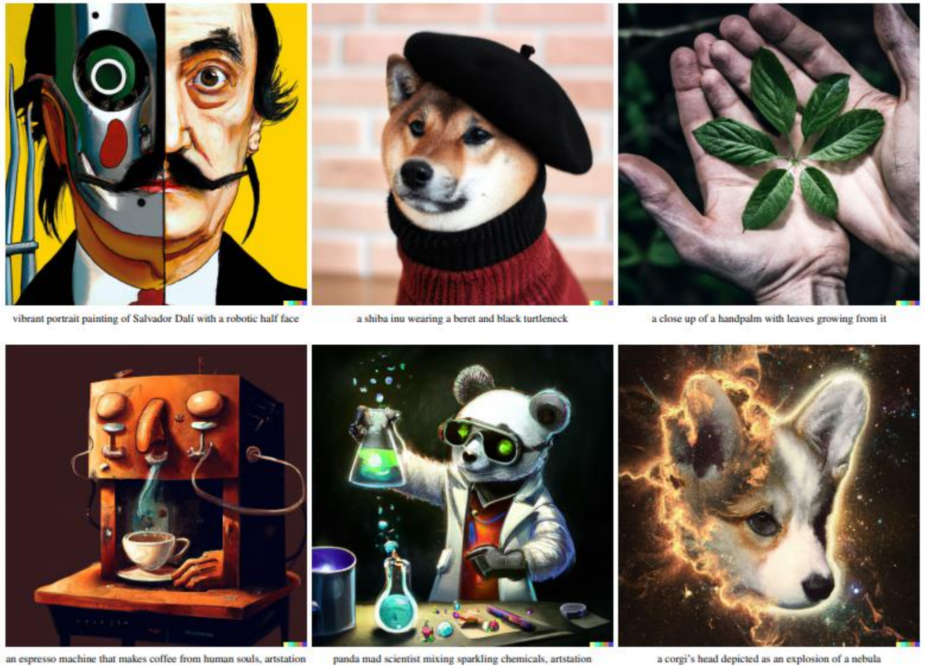
Text

$$p(x|c)$$

Next token

Prev tokens

$$p(y_n|y_{1:n-1})$$



vibrant portrait painting of Salvador Dalí with a robotic half face

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

an espresso machine that makes coffee from human souls, artstation

panda mad scientist mixing sparkling chemicals, artstation

a corgi's head depicted as an explosion of a nebula

ChatGPT

Recent advances in DL vs RL

- RL allows for novel solutions.

Image/Video

$p(x)$

Text

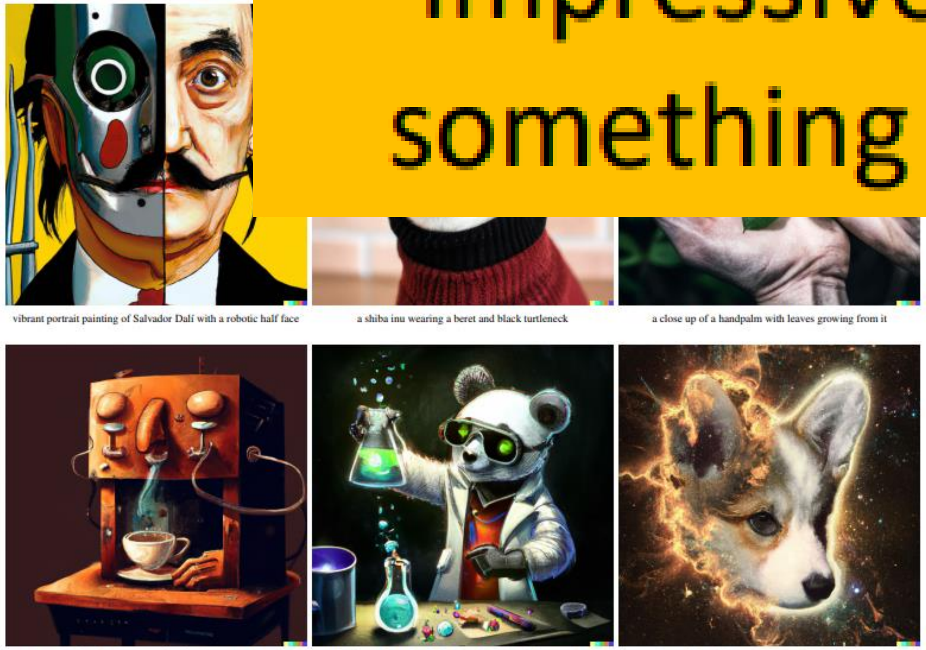
$p(x|c)$

Next token

Prev tokens

$p(y_n|y_{1:n-1})$

Impressive because it looks like something a person might draw!



vibrant portrait painting of Salvador Dalí with a robotic half face

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

an espresso machine that makes coffee from human souls, artstation

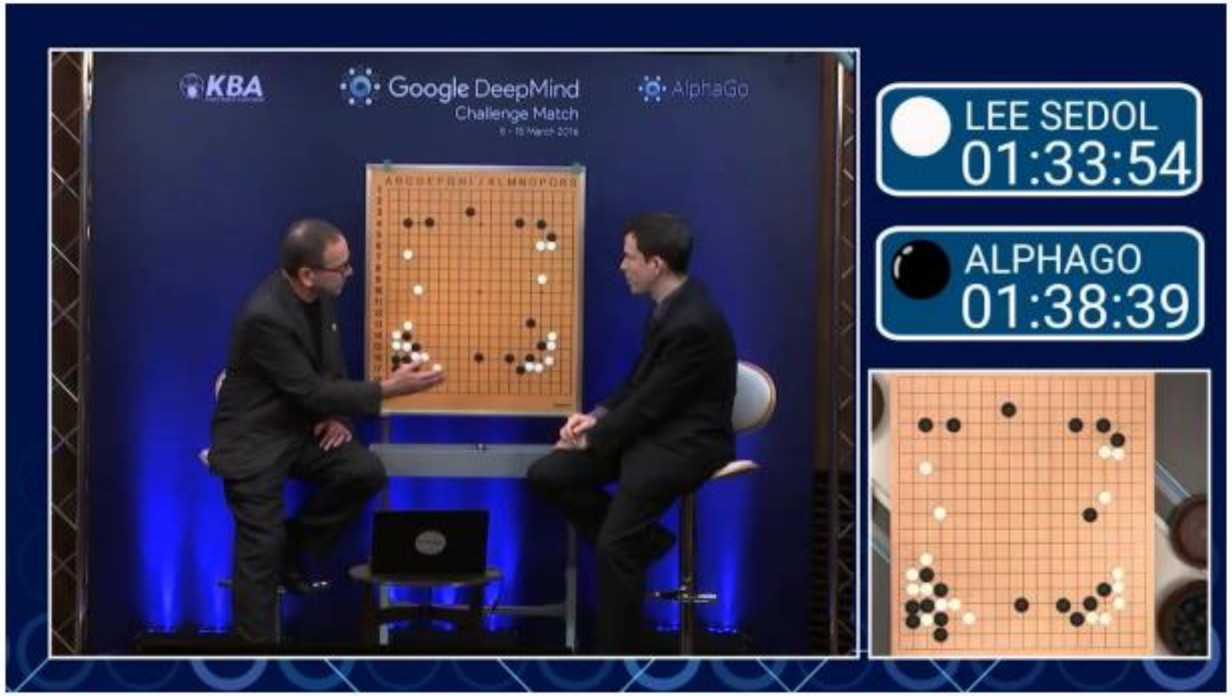
panda mad scientist mixing sparkling chemicals, artstation

a corgi's head depicted as an explosion of a nebula

ChatGPT

Recent advances in DL vs RL

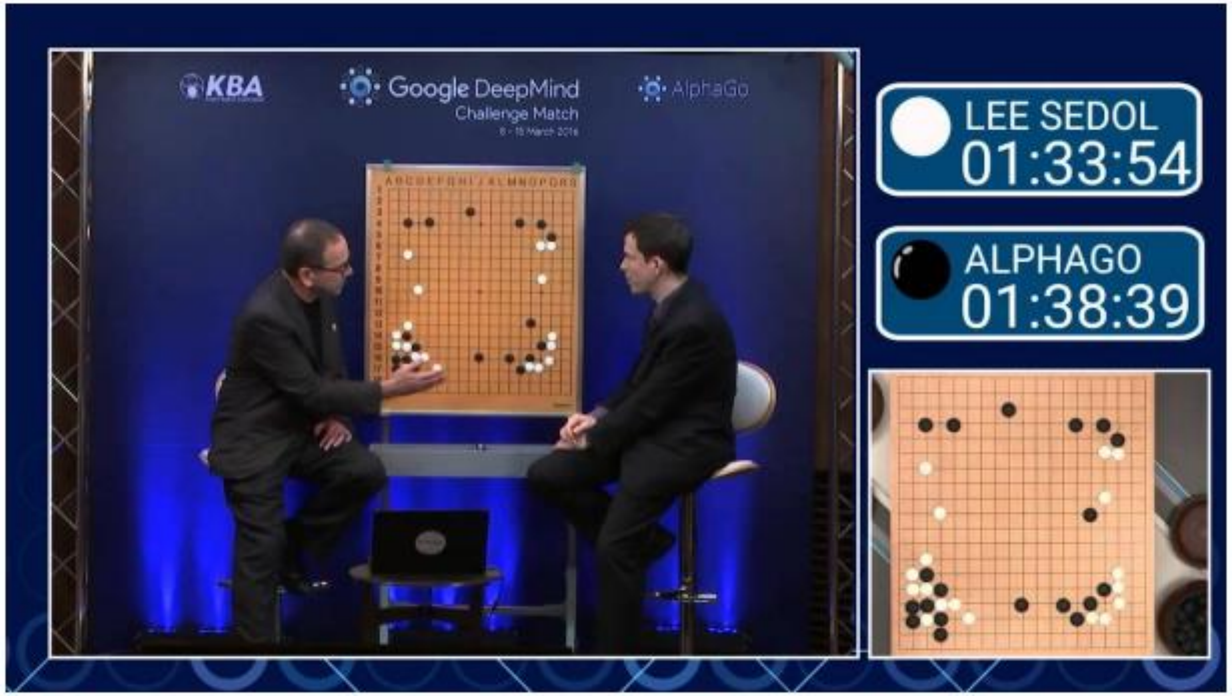
- RL allows for novel solutions.



"Move 37" in Lee Sedol AlphaGo match: reinforcement learning "discovers" a move that surprises everyone

Recent advances in DL vs RL

- RL allows for novel solutions.



"Move 37" in Lee Sedol AlphaGo match: reinforcement learning "discovers" a move that surprises everyone

Impressive because no person had thought of it!

Q and Value Functions

- State at time t: s_t
- Action at time t: a_t
- Reward function: $r(s_t, a_t)$

Q and Value Functions

The Q-Function is the **total reward from taking action** a_t in state s_t :

- State at time t: s_t
- Action at time t: a_t
- Reward function: $r(s_t, a_t)$

Q and Value Functions

The Q-Function is the **total reward from taking action** $\boldsymbol{a_t}$ in state s_t :

$$Q^\pi(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

- State at time t: s_t
- Action at time t: a_t
- Reward function: $r(s_t, a_t)$

Q and Value Functions

The Q-Function is the **total reward from taking action** a_t in state s_t :

$$Q^\pi(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_\theta}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

The Value Function is the **total reward** from state s_t :

- State at time t: s_t
- Action at time t: a_t
- Reward function: $r(s_t, a_t)$

Q and Value Functions

The Q-Function is the **total reward from taking action** a_t in state s_t :

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

The Value Function is the **total reward** from state s_t :

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

- State at time t: s_t
- Action at time t: a_t
- Reward function: $r(s_t, a_t)$

Q and Value Functions

The Q-Function is the **total reward from taking action a_t** in state s_t :

$$Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t]$$

The Value Function is the **total reward** from state s_t :

$$V^{\pi}(\mathbf{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t]$$

$$V^{\pi}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi(\boldsymbol{\alpha}_t|\mathbf{s}_t)}\left[Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t)\right]$$

Q and Value Functions

Idea 1:

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

Idea 2:

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Q and Value Functions

Idea 1:
If we have a policy π and we know the Q-Function we can improve the policy by setting:

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Q and Value Functions

Idea 1:
If we have a policy π and we know the Q-Function we can improve the policy by setting:

$$\pi'(\boldsymbol{a}|\boldsymbol{s}) = 1 \text{ if } \boldsymbol{a} = \arg\max_{\boldsymbol{a}} Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$$

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Q and Value Functions

Idea 1:
If we have a policy π and we know the Q-Function we can improve the policy by setting:

$$\pi'(\boldsymbol{a}|\boldsymbol{s}) = 1 \text{ if } \boldsymbol{a} = \arg\max_{\boldsymbol{a}} Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$$

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

Idea 2:
If $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) > V^{\pi}(\boldsymbol{s})$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Q and Value Functions

Idea 1:
If we have a policy π and we know the Q-Function we can improve the policy by setting:

$$\pi'(\boldsymbol{a}|\boldsymbol{s}) = 1 \text{ if } \boldsymbol{a} = \arg\max_{\boldsymbol{a}} Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$$

Idea 2:
If $Q^{\pi}(s, \boldsymbol{a}) > V^{\pi}(s)$ then a is better than average and we can increase $\pi(\boldsymbol{a}|\boldsymbol{s})$.

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Type of RL Algorithms

- **Policy Gradients:**
- **Value Based:**
- **Actor Critic:**
- **Model-Based:**

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Type of RL Algorithms

- **Policy Gradients**: Maximise the expected reward by direct differentiation (REINFORCE).
- **Value Based**:
- **Actor Critic**:
- **Model-Based**:

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Type of RL Algorithms

- **Policy Gradients**: Maximise the expected reward by direct differentiation (REINFORCE).
- **Value Based**: Estimate the Value or Q Function of the **optimal** policy.
- **Actor Critic**:
- **Model-Based**:

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Type of RL Algorithms

- **Policy Gradients**: Maximise the expected reward by direct differentiation (REINFORCE).
- **Value Based**: Estimate the Value or Q Function of the **optimal** policy.
- **Actor Critic**: Estimate the Value or Q Function of the **current** policy. Use it to improve policy.
- **Model-Based**:

$$Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t, \boldsymbol{a}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\boldsymbol{s}_{t'}, \boldsymbol{a}_{t'})|\boldsymbol{s}_t]$$

$$V^{\pi}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi(\boldsymbol{\alpha}_t|\boldsymbol{s}_t)}\left[Q^{\pi}(\boldsymbol{s}_t, \boldsymbol{a}_t)\right]$$

Type of RL Algorithms

- **Policy Gradients**: Maximise the expected reward by direct differentiation (REINFORCE).
- **Value Based**: Estimate the Value or Q Function of the **optimal** policy.
- **Actor Critic**: Estimate the Value or Q Function of the **current** policy. Use it to improve policy.
- **Model-Based**: Estimate the transition model and use it for planning/improving policy/other.

$$Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t, \mathbf{a}_t]$$

$$V^\pi(\mathbf{s}_t) = \sum_{t'=t}^{T} \mathbb{E}_{\pi_{\boldsymbol{\theta}}}[r(\mathbf{s}_{t'}, \mathbf{a}_{t'})|\mathbf{s}_t]$$

$$V^\pi(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi(\boldsymbol{\alpha}_t|\mathbf{s}_t)}\left[Q^\pi(\mathbf{s}_t, \mathbf{a}_t)\right]$$

Online vs Offline RL

- **Online RL:** The agent **interacts with the environment** during training.

Online vs Offline RL

- **Online RL:** The agent **interacts with the environment** during training.
- **Offline RL:** The agent **uses a fixed dataset** of previously collected experiences without further interaction with the environment during the training phase.

On-Policy vs Off-Policy RL

- **On-Policy RL:** The agent improves **the policy currently being used** to make decisions.

On-Policy vs Off-Policy RL

- **On-Policy RL:** The agent improves **the policy currently being used** to make decisions.
- **Off-Policy RL:** The agent improves **a different policy than the one it is using**.

Exploration vs. Exploitation

- **Exploitation:** Go to your favorite restaurant.
- **Exploration:** Try a new restaurant.

Exploration vs. Exploitation

- **Exploitation:** Go to your favorite restaurant.
- **Exploration:** Try a new restaurant.

- **Exploitation:** Doing what you know will yield highest reward.
- **Exploration:** Doing things you haven't done before, in the hopes of getting even higher reward.

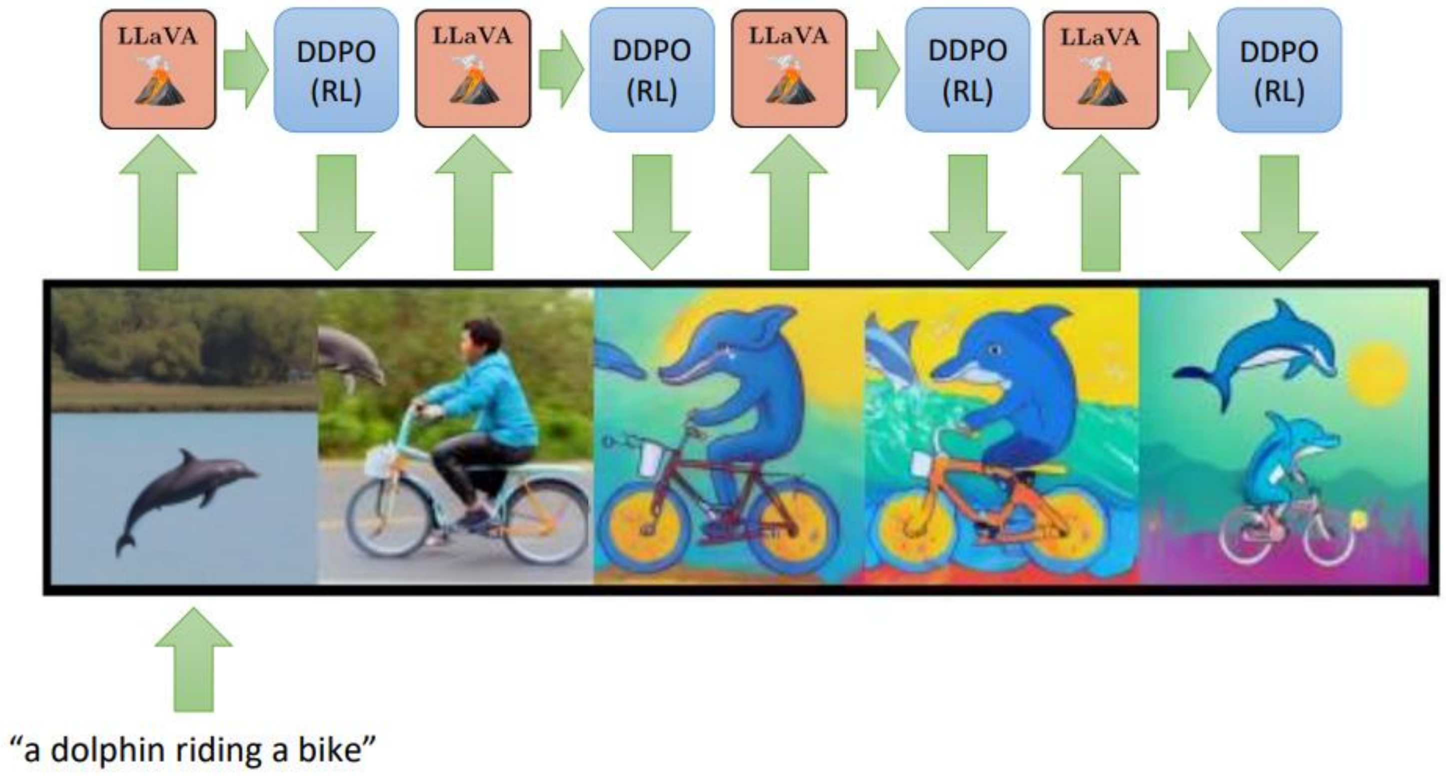
Exploration vs. Exploitation

- **Exploitation:** Go to your favorite restaurant.
- **Exploration:** Try a new restaurant.

How much should I explore?

- **Exploitation:** Doing what you know will yield highest reward.
- **Exploration:** Doing things you haven't done before, in the hopes of getting even higher reward.

Applications of RL



Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, Sergey Levine. Training Diffusion Models with Reinforcement Learning. 2023.

Applications of RL

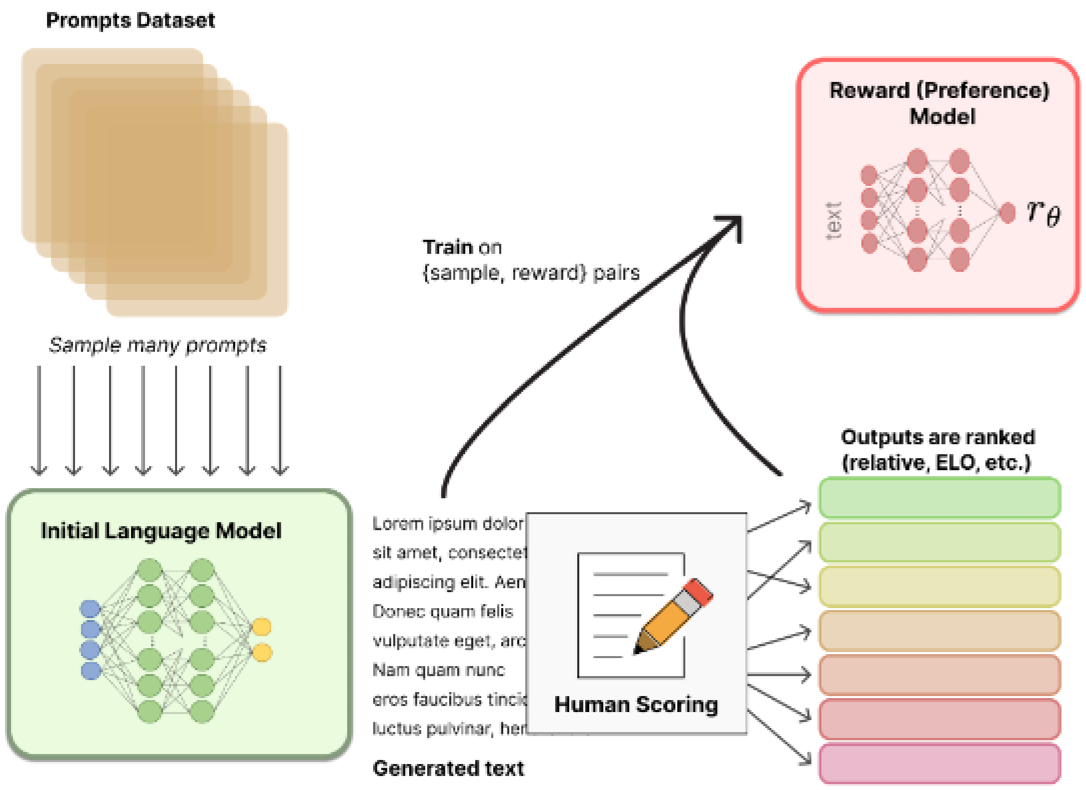
Imitation-based policies can be sensitive

source policy (motion imitation)

Smith et al., "Learning and Adapting Agility Skills by Transferring Experience." 2022.

Applications of RL

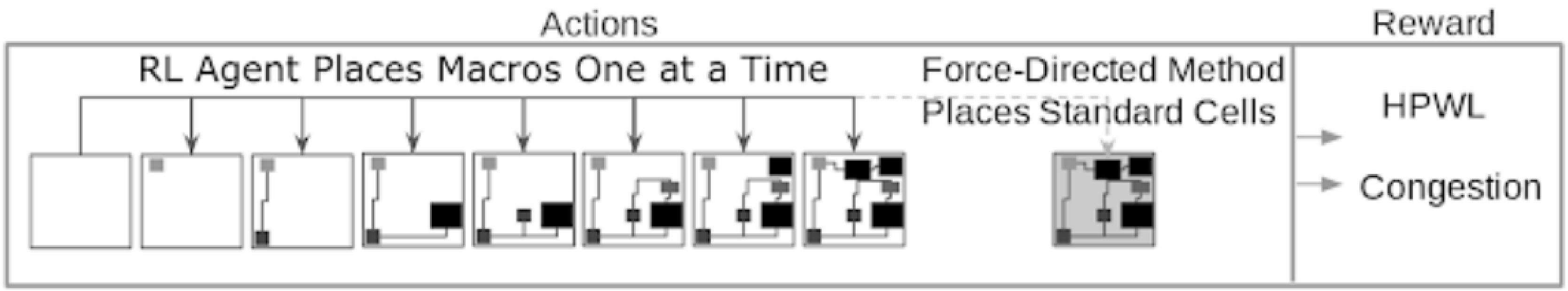
RL for fine-tuning LLMs

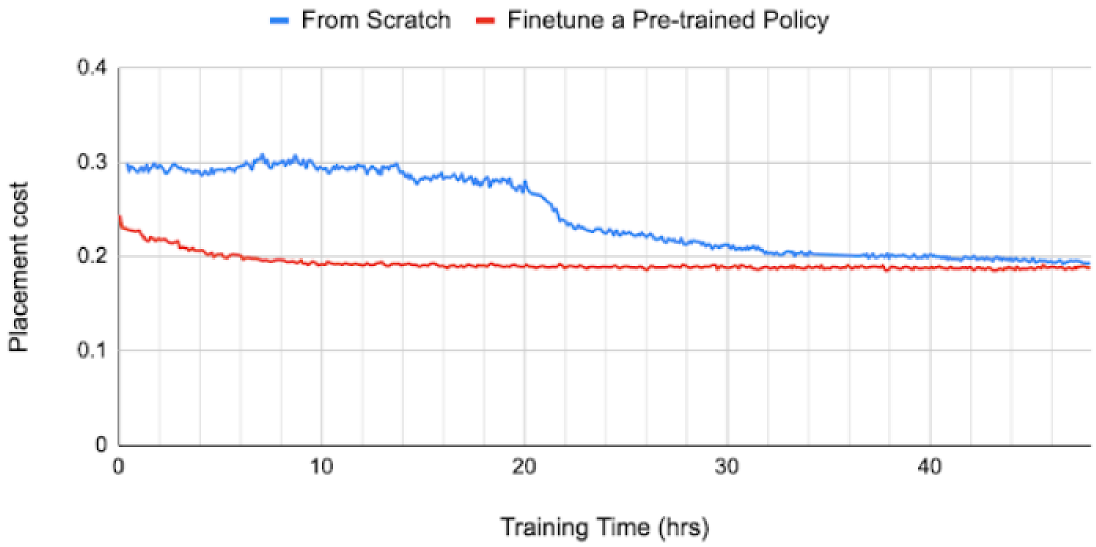


Source: https://huggingface.co/blog/rlhf

Applications of RL

RL for Chip Design

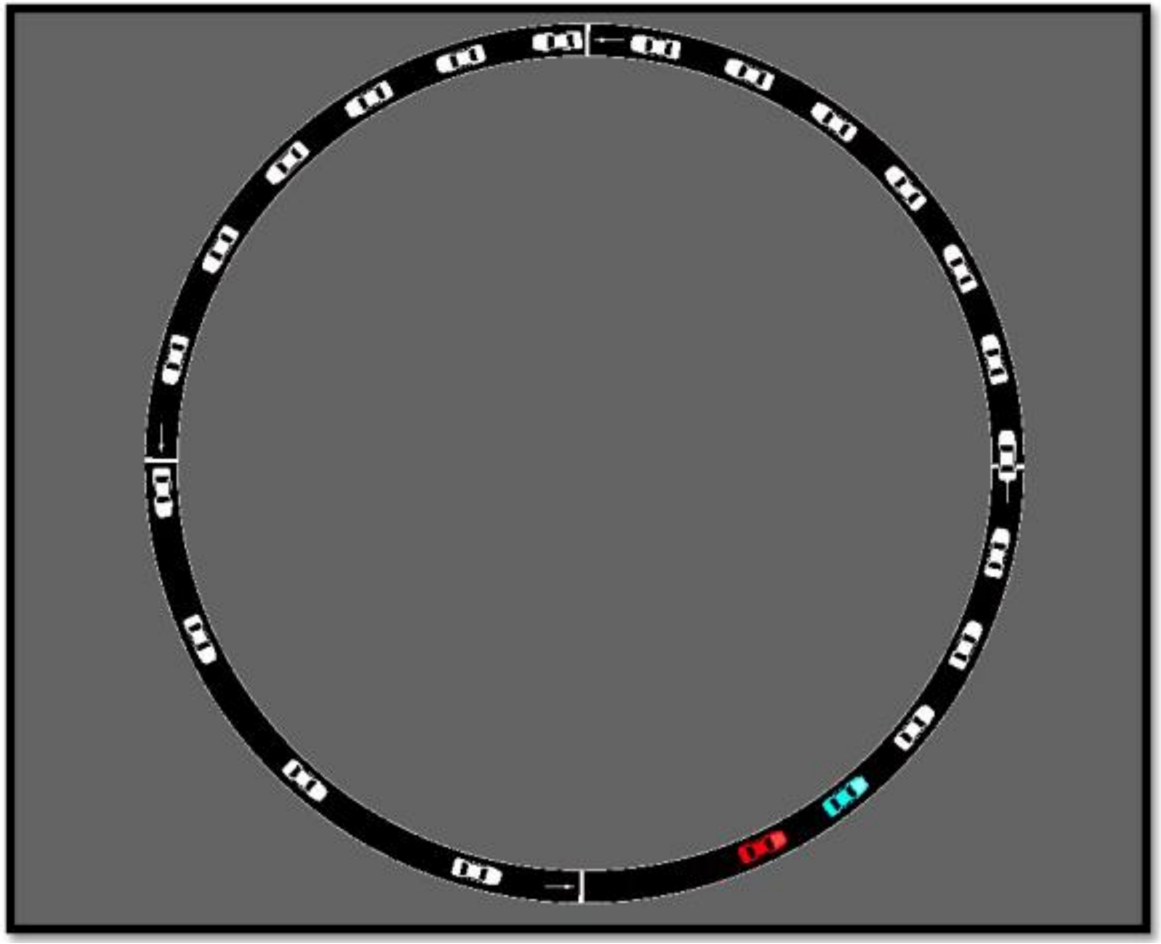


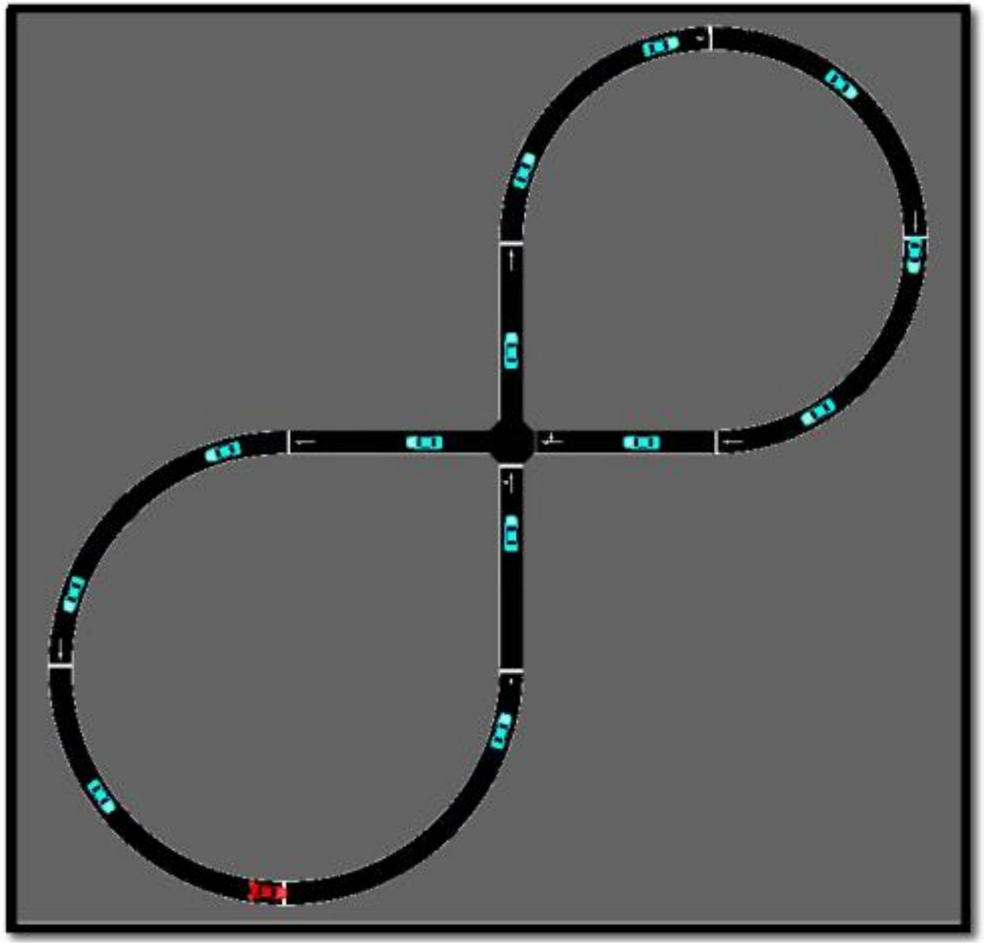


Source: https://ai.googleblog.com/2020/04/chip-design-with-deep-reinforcement.html

Applications of RL

RL for Controlling Traffic





Wu Cathy et al., Emergent Behaviors in Mixed-Autonomy Traffic, 2017