Introduction to Imitation Learning

Anup Deshmukh

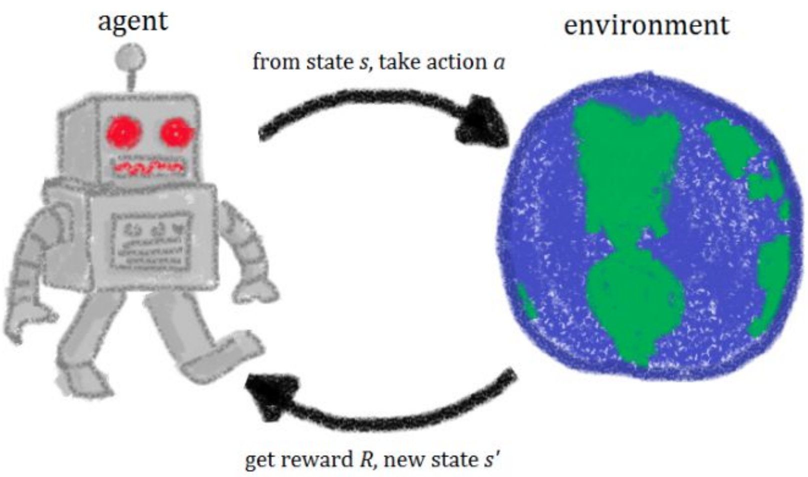
Advisor: Prof. Dinesh Babu

IIIT-Bangalore

Sections

- Reinforcement Learning 101
- Why are we not THERE yet?
- What is Imitation Learning?
- **Challenges in Imitation Learning!**
- **Posed questions and Ideas.**

1 Reinforcement Learning 101



Usual Reinforcement Learning setup

1 Reinforcement Learning 101: Approach 1

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s \right]$$

Expected

Reward discounted

Given that state

Value based : The value function tells us the maximum expected future reward the agent will get at each state.

Monte Carlo $$V(S_t) \leftarrow V(S_t) + \alpha[G_t - V(S_t)]$$

TD Learning $$V(S_t) \leftarrow \underbrace{V(S_t)}_{\text{Previous estimate}} + \alpha[\underbrace{\underbrace{R_{t+1}}_{\text{Reward t+1}} + \underbrace{\gamma V(S_{t+1})}_{\text{Discounted value on the next step}}}_{\text{TD Target}} - V(S_t)]$$

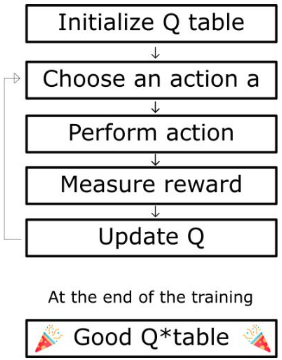
1 Reinforcement Learning 101: Approach 2

$$\pi_\theta(a|s) = P[a|s]$$

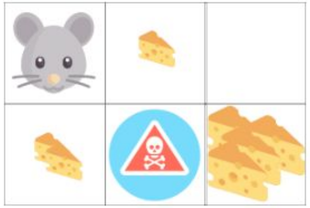
Probability of taking action a given state s with parameters theta.

Policy based: In policy-based RL, we want to directly optimize the policy function *π(s)* without using a value function.

1 Reinforcement Learning 101: Learning Method 1



Q learning: We learn the action value function by maintaining the Q table



	←	→	↑	↓
Start	0	0	0	0
Small cheese	0	0	0	0
Nothing	0	0	0	0
2 small cheese	0	0	0	0
Death	0	0	0	0
Big cheese	0	0	0	0

The initialized Q-table

1. Initialize Q-values ($Q(s,a)$) arbitrarily for all state-action pairs.
2. For life or until learning is stopped...
3. Choose an action (a) in the current world state (s) based on current Q-value estimates ($Q(s,\cdot)$).
4. Take the action (a) and observe the the outcome state (s') and reward (r).
5. Update $Q(s,a) := Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$

1 Reinforcement Learning 101: Learning Method 2

$$\pi_\theta(a|s) = P[a|s]$$

Probability of taking action a given state s with parameters theta.

$$J_{avgv}(\theta) = E_\pi(V(s)) = \sum d(s)V(s)$$

$$where \;\; d(s) = \frac{\boxed{N(s)}}{\boxed{\sum_{s'} N(s')}}$$

Number of occurrences of the state

Total nb occurences of all states

$$J_{avR}(\theta) = E_\pi(r) = \underbrace{\sum_s d(s)}_{\text{Probability that I'm in state s}} \underbrace{\sum_a \pi\theta(s,a)}_{\text{Probability that I take this action a from that state under this policy}} R_s^a$$

Immediate reward that I'll get

Policy gradient: In policy-based RL, we want to directly optimize the policy function *π(s)* without using a value function.

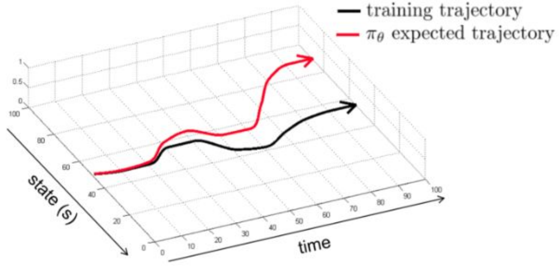
$$\begin{aligned} &Policy : \;\; \pi_\theta \\ &Objective\ function \;\; : J(\theta) \\ &Gradient : \;\; \nabla_\theta J(\theta) \\ &Update : \theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \end{aligned}$$

2 Problems in RL

- Sparsity in Rewards
  - How do you know when you have made a catastrophic move?
  - Take an example of game of Chess or Recommender Systems
- Sample inefficiency
  - How do you speed up learning in real life?
  - Again take an example of Recommender Systems

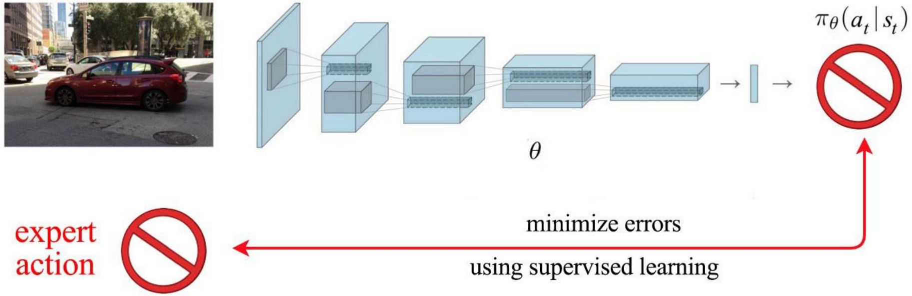
3 What is Imitation Learning?

- Idea is to implicitly give an agent prior information about the world by mimicking human behavior.
- Pretraining it on a human demonstrator's data might also make the training process faster



Source

Imitation Learning



Algorithm 43 SupervisedImitationTrain($\mathcal{A}$, $\tau_1, \tau_2, \ldots, \tau_N$)

```
D ← ⟨(x,a) : ∀n , ∀(x,a,ℓ) ∈ τ_n⟩          // collect all observation/action pairs
return A(D)                                 // train multiclass classifier on D
```

Algorithm 44 SupervisedImitationTest(f)

```
for t = 1 ... T do
    x_t ← current observation
    a_t ← f(x_t)                            // ask policy to choose an action
    take action a_t
    ℓ_t ← observe instantaneous loss
end for
return Σ_{t=1}^{T} ℓ_t                      // return total loss
```

Imitation vs. Reinforcement Learning

imitation learning

- Requires demonstrations
- Must address distributional shift
- Simple, stable supervised learning
- Only as good as the demo

reinforcement learning

- Requires reward function
- Must address exploration
- Potentially non-convergent RL
- Can become arbitrarily good

Can we get the best of both?

e.g., what if we have demonstrations *and* rewards?

Simplest combination: pretrain & finetune

- Demonstrations can overcome exploration: show us how to do the task
- Reinforcement learning can improve beyond performance of the demonstrator
- Idea: initialize with imitation learning, then finetune with reinforcement learning!

1. collected demonstration data $(\mathbf{s}_i, \mathbf{a}_i)$
2. initialize π_θ as $\max_\theta \sum_i \log \pi_\theta(\mathbf{a}_i|\mathbf{s}_i)$
3. run π_θ to collect experience
4. improve π_θ with any RL algorithm

Simplest combination: pretrain & finetune

Pretrain & finetune

1. collected demonstration data $(\mathbf{s}_i, \mathbf{a}_i)$
2. initialize π_θ as $\max_\theta \sum_i \log \pi_\theta(\mathbf{a}_i|\mathbf{s}_i)$
3. run π_θ to collect experience
4. improve π_θ with any RL algorithm

vs. DAgger

What will you do when you drift off-course?

1. train $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ from human data $\mathcal{D} = \{\mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{o}_N, \mathbf{a}_N\}$
2. run $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ to get dataset $\mathcal{D}_\pi = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $\mathbf{a}_t$
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_\pi$

5 Connection to RS: Why do we need policy gradient?

$$\pi_\theta(a|s) = P[a|s]$$

Probability of taking action a given state s with parameters theta.

$$J_{avR}(\theta) = E_\pi(r) = \underbrace{\sum_s d(s)}_{\text{Probability that I'm in state s}} \underbrace{\sum_a \pi\theta(s,a)}_{\text{Probability that I take this action a from that state under this policy}} R_s^a$$

Immediate reward that I'll get

$$\begin{aligned} &Policy : \pi_\theta \\ &Objective\ function : J(\theta) \\ &Gradient : \nabla_\theta J(\theta) \\ &Update : \theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \end{aligned}$$

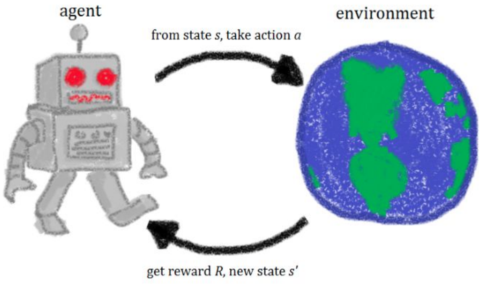
Can we arrive at the optimal policy for the generator in GAN's using the Imitation Learning?

AND

Do away with this optimization?

5 Connection to RS

$$\theta^* = \arg\max_{\theta} E_{\tau\sim p_\theta(\tau)}\left[\sum_t \underline{r(s_t, a_t)}\right]$$



Usual Reinforcement Learning setup

$$\underbrace{p_\theta(\mathbf{s}_1,\mathbf{a}_1,\ldots,\mathbf{s}_T,\mathbf{a}_T)}_{\tau} = p(\mathbf{s}_1)\prod_{t=1}^{T} \underbrace{\pi_\theta(a_t\,|\,s_t)}_{\text{policy}}\,\underbrace{p(s_{t+1}\,|\,s_t,a_t)}_{\text{model}}$$

$$\underbrace{\cancel{p_\theta(\mathbf{s}_1,\mathbf{a}_1,\ldots,\mathbf{s}_T,\mathbf{a}_T)}}_{} = \cancel{p(\mathbf{s}_1)}\prod_{t=1}^{T} \pi_\theta(a_t\,|\,s_t)\,\cancel{p(s_{t+1}\,|\,s_t,a_t)}$$

$$\textit{demonstration}\ \ p(\tau) \xrightarrow{\text{train}} \pi_\theta(a_t\,|\,s_t)$$


Thank you.