Deep Reinforcement Learning (Sp25)

Instructor: Dr. Mohammad Hossein Rohban

Summary of Lecture 10: Model Based RL

Summarized By: Arshia Gharooni

1 Introduction to Model-Based Reinforcement Learning

Model-Based Reinforcement Learning (MBRL) involves learning the **transition dynamics** of the environment and using this learned model for decision-making. Unlike **Model-Free RL**, which learns a policy directly from experience, MBRL first builds a predictive model of the environment and then uses it for planning and control.

2 Overview of Model-Based RL

The main question in model-based RL is: **What if we know the dynamics?** How can we make decisions based on known dynamics? The key planning techniques used in MBRL include stochastic optimization methods, Monte Carlo Tree Search (MCTS), and trajectory optimization. The goal is to understand how planning works with known dynamics in both discrete and continuous spaces.

Model-based RL involves learning a **dynamics model** and using it for action selection. The challenge is how to choose actions under perfect knowledge of the system dynamics. Techniques like optimal control, trajectory optimization, and planning are used to generate policies.

3 Recap: Model-Free RL

Model-free RL directly learns a policy or value function from data, without explicitly modeling the transition function. In contrast, model-based RL explicitly learns the transition function, enabling planning and more sample-efficient learning.

4 What if We Knew the Transition Dynamics?

Some domains have **known transition dynamics**, such as:

- Games like Chess, Go, and Atari, where the transition function is explicit.
- Easily modeled systems such as car navigation.
- Simulated environments, including robotics and physics-based simulations.

In other cases, we may **learn the dynamics** through system identification (fitting unknown parameters of a known model) or general learning (fitting a model to observed transition data).

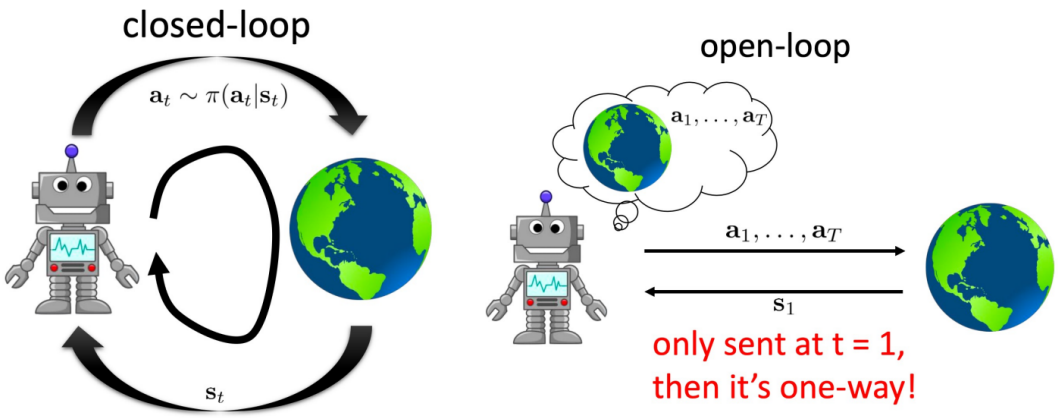

Figure 1: open-loop vs. closed-loop case

5 Open-Loop vs. Closed-Loop Planning

5.1 Closed-Loop Planning

Actions depend on the current state, following a policy:

$$a_t \sim \pi(a_t|s_t) \tag{1}$$

5.2 Open-Loop Planning

In open-loop planning, the entire action sequence $(a_1, ..., a_T)$ is precomputed and executed without feedback from the environment. The dynamics are given by:

$$s_{t+1} = f(s_t, a_t) \tag{2}$$

Actions are **only sent at** $t = 1$, making it a one-way execution.

6 Stochastic Optimization

Optimal control is often formulated as an optimization problem:

$$a_{1:T} = \arg\max_{a_{1:T}} J(a_{1:T}) \tag{3}$$

where the objective function is the cumulative reward:

$$J(A) = \sum_{t=1}^{T} r(s_t, a_t) \tag{4}$$

and state transitions follow:

$$s_t = f(s_{t-1}, a_{t-1}) \tag{5}$$

6.1 Random Shooting Method

The simplest way to optimize is to sample random action sequences and choose the best one:

1. Sample actions from a distribution.
2. Evaluate the reward for each sampled sequence.
3. Select the sequence with the highest reward.

7 Cross-Entropy Method (CEM)

CEM improves random shooting by refining the action distribution iteratively:

1. Sample action sequences from a probability distribution $p(A)$.
2. Evaluate their expected returns $J(A)$.
3. Select the top M elite sequences.
4. Update $p(A)$ based on the elite sequences.

8 Monte Carlo Tree Search (MCTS) (Discrete Case)

MCTS is a tree-based planning method that balances exploration and exploitation using an Upper Confidence Bound (UCB) formula:

$$Score(s_t) = \frac{Q(s_t)}{N(s_t)} + 2C\sqrt{\frac{2\ln N(s_{t-1})}{N(s_t)}} \tag{6}$$

8.1 UCT Tree Policy

1. If a node is not fully expanded, select a new action.
2. Otherwise, choose the child node with the best UCB score.
3. Rollout simulations with a random policy to approximate values.

9 Uncertainty in Model-Based RL

A key challenge in MBRL is **distribution shift**, where model errors accumulate as planning steps increase. Approaches to mitigate this include:

- Bayesian Neural Networks (BNNs) to quantify model uncertainty.
- Ensemble models (Bootstrap ensembles) for robust predictions.
- Model-based rollouts with uncertainty estimation.

The optimization objective incorporates uncertainty:

$$\max_{a_t} \mathbb{E}[R(a_t)] - \lambda \text{Var}[R(a_t)] \tag{7}$$

which discourages over-reliance on uncertain predictions.

10 Model-Based RL with Policy Learning

10.1 Challenges of Backpropagation into Policy

Backpropagation in model-based RL faces issues like vanishing/exploding gradients and sensitivity to trajectory optimization errors. Instead of directly optimizing actions, an alternative is to use model-free RL on synthetic rollouts generated by the model.

10.2 Dyna-Style Model-Based RL

1. Learn a model from real-world data.
2. Use it to generate synthetic rollouts.
3. Train a model-free RL agent on these rollouts.

11 The Curse of Long Model-Based Rollouts

Model-based rollouts introduce errors that compound over time. One solution is to use **short-horizon rollouts with frequent re-planning**, which prevents the accumulation of model errors. **Model Predictive Control (MPC)** is a commonly used method for mitigating these issues.