UNIT V

REINFORCEMENT LEARNING

Model Based Learning

Model-based reinforcement learning. As we'll see, model-based RL attempts to overcome the issue of a lack of prior knowledge by enabling the agent — whether this agent happens to be a robot in the real world, an avatar in a virtual one, or just a piece software that take actions — to construct a functional representation of its environment.

While model-based reinforcement learning may not have clear commercial applications at this stage, its potential impact is enormous. After all, as AI becomes more complex and adaptive — extending beyond a focus on classification and representation toward more human-centered capabilities — model-based RL will almost certainly play an essential role in shaping these frontiers.

To Model or Not to Model

"Model" is one of those terms that gets thrown around a lot in machine learning (and in scientific disciplines more generally), often with a relatively vague explanation of what we mean. Fortunately, in reinforcement learning, a model has a very specific meaning: it refers to the different dynamic states of an environment and how these states lead to a reward.

Model-based RL entails constructing such a model. Model-free RL, conversely, forgoes this environmental information and only concerns itself with determining what action to take given a specific state. As a result, model-based RL tends to emphasize planning, whereas model-free RL tends to emphasize learning (that said, a lot of learning also goes on in model-based RL). The distinction between these two approaches can seem a bit abstract, so let's consider a real-world analogy.

Imagine you're visiting a city that you've never been to before and for whatever reason you don't have access to a map. You know the general direction from your hotel to the area where most of the sights of interest are, but there are quite a number of different possible routes, some of which lead you through a slightly dangerous neighborhood.

A state graph from a paper on RL approaches for simulated urban environments

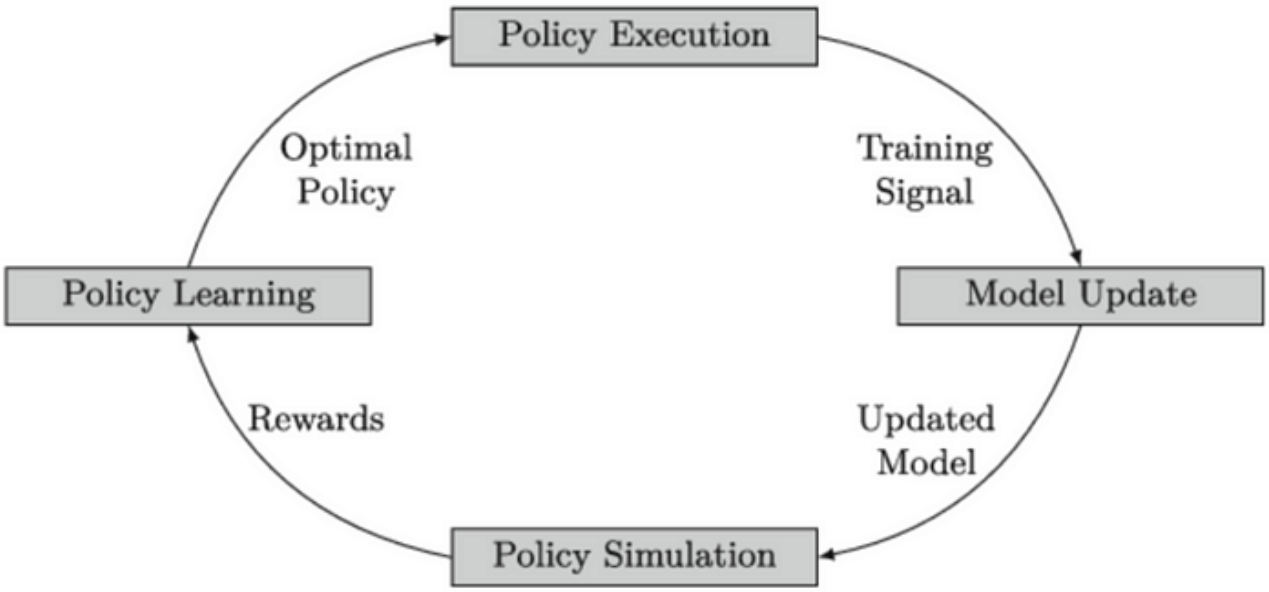
One navigational option is to keep track of all the routes you've taken (and the different streets and landmarks that make up these routes) to begin to create a map of the area. This map would be incomplete (it would only rely on where you'd already walked), but would at least allow you to plan a course ahead of time to avoid that neighborhood while still optimizing for the most direct route. You could even spend time back in your hotel room drawing out the different possible itineraries on a sheet of paper and trying to gauge which one seems like the best overall option. You can think of this as a model-based approach.

Another option — especially if you're the type of person who's not big on planning — would simply be to keep track of the different locations you'd visited (intersections, parks, and squares for instance) and the actions you took (which

way you turned), but ignore the details of the routes themselves. In this case, whenever you found yourself in a location you'd already visited, you could favor the directional choice that led to a good outcome (avoiding the dangerous neighborhood and arriving at your destination more efficiently) over the directions that led to a negative outcome. You wouldn't specifically know the next location you'd arrive at with each decision, but you would at least have learned a simple procedure for what action to take given a specific location. This is essentially the approach that model-free RL takes.

As it relates to specific RL terms and concepts, we can say that you, the urban navigator, are the ***agent***; that the different locations at which you need to make a directional decision are the ***states***; and that the direction you choose to take from these states are the ***actions***. The ***rewards*** (the feedback based on the agent's actions) would most likely be positive anytime an action both got you closer to your destination and avoided the dangerous neighborhood, zero if you avoided the neighborhood but failed to get closer to your destination, and negative anytime you failed to avoid the neighborhood. The ***policy*** is whatever strategy you use to determine what action/direction to take based on your current state/location. Finally, the ***value*** is the expected long-term return (the sum of all your current and future rewards) based on your current state and policy.

In general, the core function of RL algorithms is to determine a policy that maximizes this long-term return, though there are a variety of different methods and algorithms to accomplish this. And again, the major difference between model-based and model-free RL is simply that the former incorporates a model of the agent's environment, specifically one that influences how the agent's overall policy is determined.



A flow diagram of model-based RL

A Modest Comparison

So what are the pros and cons of the model-based vs. the model-free approach? Model-based RL has a lot going for it. For one thing, it tends to have higher sample efficiency than model-free RL, meaning it requires less data to learn a policy. In other words, by leveraging the information it's learned about its environment, model-based RL can plan rather than just react, even simulating sequences of actions without having to directly perform them in the actual environment.

A related benefit is that by virtue of the modeling process, model-based RL has the potential to be transferable to other goals and tasks. While learning a single policy is good for one task, if you can predict the dynamics of the environment, you can generalize those insights to multiple tasks. Finally, having a model means you can determine some degree of model uncertainty, so that you can gauge how confident you should be about the resulting decision process.

Moving to the cons of model-based RL (or the pros of model-free RL), one of the biggest ones is simply that by having to learn a policy (the overall strategy to maximize the reward) as well as a model, you're compounding the degree of potential error. In other words, there are two different sources of approximation error in model-based RL, whereas in model-free RL there's only one. For similar reasons, model-based approaches tend to be far more computationally demanding than model-free ones, which by definition simplify the learning process.

It's worth noting that this doesn't necessarily need to be a binary decision. Some of the most effective recent approaches have combined model-based and model-free strategies. Perhaps this isn't so surprising given the evidence that, as one paper states, "the [human] brain employs both model-free and model-based decision-making strategies in parallel, with each dominating in different circumstances."