Reinforcement Learning

Introduction

Introduction

The general aim of Machine Learning is to produce intelligent programs, often called agents, through a process of learning and evolving. Reinforcement Learning (RL) is one approach that can be taken for this learning process. An RL agent learns by interacting with its environment and observing the results of these interactions. This mimics the fundamental way in which humans (and animals alike) learn. As humans, we have a direct sensori-motor connection to our environment, meaning we can perform actions and witness the results of these actions on the environment. The idea is commonly known as "cause and effect", and this undoubtedly is the key to building up knowledge of our environment throughout our lifetime.

The "cause and effect" idea can be translated into the following steps for an RL agent:

1. The agent observes an input state
2. An action is determined by a decision making function (policy)
3. The action is performed
4. The agent receives a scalar reward or reinforcement from the environment
5. Information about the reward given for that state / action pair is recorded

By performing actions, and obersving the resulting reward, the policy used to determine the best action for a state can be fine-tuned. Eventually, if enough states are observed an optimal decision policy will be generated and we will have an agent that performs perfectly in that particular environment.

Supervised and Unsupervised Learning

Supervised and Unsupervised learning are two quite different techniques of learning. As the names suggest, supervised learning involves learning with some supervision from an external source (i.e. a teacher) whereas unsupervised learning does not. An example of supervised learning is a student taking an exam, having it marked and then being shown which questions they answered incorrectly. After being shown the correct answers, the student should then learn to answer those questions successfully as well. An example of unsupervised learning is someone learning to juggle by themselves. The person will start by throwing the balls and attempting to catch them again. After dropping most of the balls initially, they will gradually adjust their technique and start to keep the balls in the air.

How does this relate to Reinforcement Learning? As described in the steps above, an RL agent learns by receiving a reward or reinforcement from its environment, without any form of supervision other than its own decision making policy. So, RL is a form of unsupervised learning. What this means is that an agent can learn by being set loose in its environment, without the need for specific training data to be generated and then used to teach the agent.

Exploration and Exploitation

One of the interesting problems that arises when using Reinforcement Learning is the tradeoff between exploration and exploitation. If an agent has tried a certain action in the past and got a decent reward, then repeating this action is going to reproduce the reward. In doing so, the agent is exploiting what it knows to receive a reward. On the other hand, trying other possibilities may produce a better reward, so exploring is definitely a good tactic sometimes. Without a balance of both exploration and exploitation the RL agent will not learn successfully. The most common way to achieve a nice balance is to try a variety of actions while progessively favouring those that stand out as producing the most reward.

Uses for Reinforcement Learning

A variety of different problems can be solved using Reinforcement Learning. Because RL agents can learn without expert supervision, the type of problems that are best suited to RL are complex problems where there appears to be no obvious or easily programmable solution. Two of the main ones are:

> Game playing - determining the best move to make in a game often depends on a number of different factors, hence the number of possible states that can exist in a particular game is usually very large. To cover this many states using a standard rule based approach would mean specifying an also large number of hard coded rules. RL cuts out the need to manually specify rules, agents learn simply by playing the game. For two player games such as backgammon, agents can be trained by playing against other human players or even other RL agents.
>
> Control problems - such as elevator scheduling. Again, it is not obvious what strategies would provide the best, most timely elevator service. For control problems such as this, RL agents can be left to learn in a simulated environment and eventually they will come up with good controlling policies. Some advantages of using RL for control problems is that an agent can be retrained easily to adapt to environment changes, and trained continuously while the system is online, improving performance all the time.

A detailed analysis of using RL for playing backgammon can be found here on the IBM research website. If card games appeal to you, here is a nice applet that learns to play blackjack.

Next...

Value Functions, Temporal Difference Learning, On-Policy and Off-Policy learning and Action Selection policies.

Previous page Next page