Chapter 1

Monday, December 17, 2018 10:40 AM

Page 1 to 5:

Introduction:

Humans learn by interacting with environment. When an infant plays, waves its arms or looks about, it has a direct sensorimotor connection to its environment and this connection produces information about the cause and effect, consequences of actions and also what to do in order to achieve goals.
In everyday life we are aware about how the environment reacts to what we do and we try to influence what happens through the behavior. Learning from interaction is a foundational idea in nearly all theories of learning and intelligence.
Rather than directly theorizing about how people or animals learn, the book explores idealized learning situations and evaluate the effectiveness of various learning methods.
The book explores designs for machines that are effective in solving learning problems, evaluating the designs via mathematical analysis or computational experiments. So, the book is more focused on goal-directed learning from interaction.

1.1. Reinforcement Learning

RL is learning what-to-do (how to map situations to actions) to maximize a numeric reward signal. The learner has no idea about which actions to take, but it has to discover which actions yield the most reward by trying them. But in some cases the actions not only affect the immediate reward but also the next situation and through that all the subsequent rewards.

Trial-and-error search and delayed reward are the two important characteristics and distinguishing features of RL.

The RL problem can be formalized using the ideas from dynamical systems theory , like optimal control for the incompletely known Markov Decision processes. Markov Decision processes are intended to include just three aspects - sensation, action and goal.

The learning agent should be able to sense the state of its environment to some extent and must be able to take actions that affect the state and also the agent should have a goal or goals relating to the state of the environment.

RL is different from Supervised Learning in the fact that the Supervised learning is provided with the labeled examples (labeled by a knowledgeable external supervisor) and models will train from them.

RL is also different from the Unsupervised Learning(USL) in the fact that the USL is typically about finding the structure hidden within the collections unlabeled data. But RL is trying to maximize a reward signal instead of trying to find the hidden structure.

Of all the forms of machine learning, reinforcement learning is the closest to the kind of learning that humans and other animals do, and many of the core algorithms of reinforcement learning were originally inspired by biological learning systems.

Uncovering the hidden structure in an agent's experience certainly be useful in reinforcement learning, but by itself it does not address the problem of maximizing a reward signal.

Hence the RL is considered as a new paradigm apart from Supervised and Unsupervised learning.

The challenges that arise in RL is the trade-off between exploration and exploitation.
Exploitation:
To get a lot of reward the agent has to prefer actions that it has tried before and found to be effective in producing reward. This is called as exploitation and the agent has to do it to obtain reward

Exploration:
But in order to discover the actions that offer good reward, the agent has to try actions that it has not selected before and find them . This is called as exploration and it is needed to make better action selections in the future.

The agent must try a variety of actions and progressively favor that appear to be best. On a stochastic task each action must be tried many times to gain a reliable estimate of its expected reward.

RL explicitly considers the whole problem of a goal-directed agent interacting with an uncertain environment. But much of the machine learning research concerned with supervised learning divide that big problem into a lot of sub-problems, the focus on isolated subproblems is the limitation for this approach.

RL agents have explicit goals, can sense the environments, and can choose actions to influence their environments.

1.2. Examples

Some of the examples that explain the reinforcement learning:

1. Chess player makes a move:
   The choice is informed by planning (anticipating possible replies and counter replies) and by immediate judgements of the desirability of particular positions and moves.
2. An adaptive controller adjusts the parameters of refinery's operation in real time:
   It optimizes the yield/ cost / quality trade-off on the basis of specified marginal costs without strictly following the set points originally set by engineers.
3. A gazelle calf learning to run after half an hour it is born
   The calf will struggle to its feet minutes after being born and it will learn using trial and error

All of the above examples involve the interaction between an active - decision making agent and its environment, within which the agent seeks to achieve a goal despite uncertainty about its environment. The agents actions affect the future state of the environment (e.g., the next chess position, the level of reservoirs in refinery), which will affect the actions and opportunities available to the agent at the later times. Foresight and planning, which take into account the indirect and delayed consequences of actions, are required to choose correct actions. The agents can judge its progress towards the goal based on what it can sense directly.
The knowledge the agent brings to the task at the start—either from previous experience with related tasks or built into it by design or evolution—influences what is useful or easy to learn, but interaction with the environment is essential for adjusting behavior to exploit specific features of the task.

1.3. Elements of Reinforcement Learning

Beyond the agent and the environment there are four main sub-elements of a reinforcement learning system:

1. A policy
2. A reward signal
3. A value function
4. A model of the environment (optional)

Policy:

1. Agent's way of behaving at a given time.
2. Mapping from perceived states of the environment to actions to be taken
3. In terms of Psychology set of stimulus-response rules
4. May be a simple function or lookup table or extensive computation such as search process
5. It is the core of a reinforcement learning agent it alone can be sufficient to determine behavior
6. Generally, policies may be stochastic, specifying probabilities for each action

Reward Signal:

1. Defines the goal of a reinforcement learning problem
2. On each time step the environment provides the agent a single number called reward
3. Agent's sole objective is to maximize the total reward it receives over the long run
4. It denotes the events if it is good or bad for the agent
5. Reward is analogous to the pain or pleasure in biological systems
6. It is the primary basis for altering the policy
7. Reward signals are the stochastic functions of the state of the environment and the actions taken

Value Function:

1. Specifies what is good in long run
2. Value of a state is the total amount of reward an agent can expect to accumulate over the future starting from that state
3. Values indicate the long-term desirability of states taking into account the states that are likely to follow and rewards available in those states.
4. Rewards are in a sense primary, whereas values, as predictions of rewards, are secondary.
5. Only purpose of estimating values is to achieve more reward
6. Value are the most important ones in making and evaluating the decisions
7. Action choices are made based on value judgements, that bring states of highest value because these actions obtain the greatest amount of reward over the long run
8. Values are hard to be determined because it has to be estimated and re-estimated from the sequences of observations an agent makes over its entire lifetime
9. Method for efficiently estimating the values is an important one

Model of the environment (Optional):

1. It mimics the behavior of the environment
2. Given a state and an action the model might predict the resultant next state and next reward
3. Models are used for planning, they provide the way of deciding on a course of action by considering possible future situations before they are actually experienced.
4. Methods for solving reinforcement problems that use models and planning are called model-based methods, opposed to simpler model-free methods that ae explicitly trial- and-error learners.

1.4. Limitations and Scope

Reinforcement learning relies heavily on the states. They are passed as input to the policy and value function and as both input to and output from the model. State can be considered as a signal conveying to the agent some sense of "how the environment is" at a particular time.
Most of the reinforcement learning methods considered in this book are structured around estimating the value functions, but it is not necessarily to do the same way. Other methods such as genetic algorithms, genetic programming, simulated annealing and other optimization methods never estimate value functions.
In algorithms like the one above, multiple static policies interacting over an extended period of time with a separate instance of the environment. The policies that perform well and getting the most reward and random variations of them are carried over to the next generation of policies and the

process is repeated. These are called as evolutionary methods because they are analogous to the way biological evolution produces organisms with skilled behavior. These methods are particularly effective if the space of policies is sufficiently small or can be structures so that the good policies are common or easy to find. They are also having advantage on problems in which the learning agent cannot sense the complete state of its environment.
Evolutionary methods won't learn while interacting with the environment. But the ones that learn when interacting with the environment are much more efficient than the evolutionary methods. Because evolutionary methods ignore much of the useful structure of the environment. They do not use the fact that the policy they are searching for is a functions from states to actions and also they do not notice which states an individual passes through during its lifetime or which action it selects.
But evolutionary methods along with the learning methods work together greatly.

1.5. An Extended Example : Tic-Tac-Toe

In tic-tac-toe, two players take turns playing on a three-by=three board. One player plays Xs and the other Os until one player wins by placing three marks in a row, horizontally, vertically or diagonally. If the board fills up with neither player getting three in a row, then the game is a draw.
In this example draw and loss are considered bad.

It is a simple problem yet it cannot be readily solved using the classical techniques. They require complete specification of that opponent or it assumes that a particular way of playing by the opponent. Practically these information are not available. On the other hand these information can be estimated from experience, in this case be playing many games against the opponent.

First learn a model of the opponent's behavior, up to some level of confidence and dynamic programming has to be applied to compute an optimal solution given the approximate opponent model.

But the evolutionary method is applied would directly search the space of possible policies for one with a high probability of winning against the opponent. Policy is a rule that tells the player what move to make for every state of the game (every possible configuration of Xs and Os). For each policy taken estimate of its winning probability will be found out by playing some number of games against the opponent. It is essential to direct which policy or policies are to be considered next. So this method would hill climb the policy space, by successively generating and evaluating policies in an attempt to obtain incremental improvements.

In method making use of a value function, a table of numbers has to be formed and each of the number denotes each possible state of the game. Each of the number represents the latest estimate of the probability of winning from that state. It is the value of the state. Whole table is the learned value function. If one state's value is more than the other then it is considered better that the later one.
For example, for all states with three Xs in a row the probability of winning is 1. Then for all the states with three Os in a row, or the ones in which all the boxes are filled up the probability is 0. The initial values of all the other states are 0.5, representing a guess of 50% chance of winning.

Many games will be played against the opponent. In order to select the moves, states that result from those moves have to be examined and their current values have to be looked up in the table. Most of these moves have to be done greedily which leads to the states with greatest value. Occasionally states have to selected randomly (exploratory moves). These exploratory moves will be used to evaluate the states that will be missed when moved greedily (exploiting moves).

While playing the value of the states we were will be changed. This is an attempt to make the estimates more accurate. To do this, we back up the value of the state after each greedy move to the state before the move, denoted in the figure below.

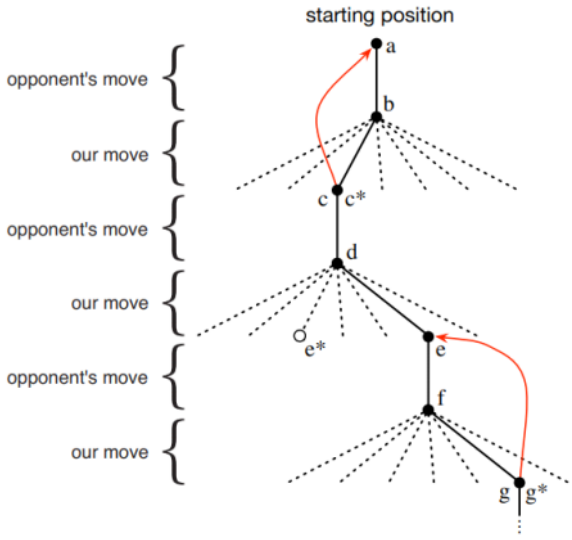

Figure Description:
A sequence of tic-tac-toe moves. The solid black lines represent the moves taken during a game; the dashed lines represent moves that we (our reinforcement learning player) considered but did not make. Our second move was an exploratory move, meaning that it was taken even though another sibling move, the one leading to e*, was ranked higher. Exploratory moves do not result in any learning, but each of our other moves does, causing updates as suggested by the red arrows in which estimated values are moved up the tree from later nodes to earlier nodes as detailed in the text.

So the current value of the earlier state is updated closer to the value of the later state. It is done by moving the value of earlier state a fraction of the way toward the value of the later state.

If we let S_t denote the state before the greedy move, and S_{t+1} the state after the move, then the update to the estimated value of S_t, denoted V (S_t), can be written as

$$V(S_t) \leftarrow V(S_t) + \alpha\left[V(S_{t+1}) - V(S_t)\right]$$

Where α is a small positive fraction called the step-size parameter, which decides the rate of learning. This update rule is an example of temporal difference learning, as the changes are based on a difference , $V(S_{t+1})$-$V(S_t)$ between estimates at two successive times.

How the method mentioned above performs well?
If the step size parameter is reduced properly over time, this method converges, for any fixed opponent, to the probabilities of winning from each state given optimal play. The moves then taken are in fact the optimal moves against the opponent. So the method converges to optimal policy for playing the game against this opponent. If the step-size parameter is not reduced all the way to zero over time, then the player also plays well against the opponents that slowly change their way of playing.

To evaluate a policy an evolutionary method holds the policy fixed and simulates many games using a model of the opponent. The frequency of wins gives an unbiased estimate of the probability of winning with that policy and used in next policy selection. Each policy change is only made after many games. Also, if the player wins all of its behavior in the game is given credit, they do not consider how specific moves were critical to the win. Even credit is given to the moves that have never occurred. So no information what happened during the game is considered in an evolutionary method, but it is considered in the case of the value function methods.

Key Features of Reinforcement Learning:

1. Emphasis on learning while interacting with an environment.
2. Clear goal and correct behavior requires planning or foresight that takes into account delayed effects of one's choices.

For example, a simple reinforcement learning can learn to set-up multi-move traps for a short-sighted opponent. So a reinforcement learning agent can achieve the effects of planning and look ahead without using the model of the opponent and without conducting an explicit search over possible sequences of future states and actions.

Reinforcement learning is not restricted to only problem in which behavior breaks down into separate episodes. It can be applicable when behavior continues indefinitely and when rewards of various magnitudes can be received at any time. The reinforcement learning can also be applied to continuous-time problems as well, but it is complicated.

Tic-tac-toe has a relatively small, finite state set. But RL can be applied to problems where the state set is very large or even infinite. The artificial neural network provides the RL with the ability to generalize from its experience, so that it takes moves based on information saved from similar states faced in the past. How well RL works with these kind of large problems depend on how appropriately it generalize from the past.

In this example the learning started with no prior knowledge beyond the rules of the game. Prior information can be incorporated into RL in a variety of ways and they are critical for efficient learning. RL can also applied to the problems when part of the state is hidden or when different states appear to the learner be the same.

RL can be applied to both if the model of the environment is available to foresee the changes or if the model of the environment is not available. Model free methods are good for the cases in which the bottle neck is the difficulty of constructing a sufficiently accurate environment model. Model free methods are also important building blocks of model-based methods.

RL can be used both at the high and low levels of a system. In hierarchical learning systems, RL can work simultaneously on several levels.