Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology-Delhi

Lecture - 79
Reinforcement Learning: TD Learning

So where are we? We have been talking about reinforcement learning as the next frontier from Markov decision processes. In Markov decision processes, the transition function and reward function was known to us. In the reinforcement learning the transition function and other reward functions are not known to us, right. So this is what happens when the agent does not have the model of the world.

The agent does not know how the world behaves. The agent has to get data from the world, observe the world and based on the observation, learn about the world and also figure out what are the best actions it should do in order to optimize its objective function, right. So that is sort of the highest level bit of reinforcement learning.

So if you think about all of machine learning, right, think about you know image classification or text classification or any such problem that is your favorite problem in machine learning. You are given some data and then you have to learn a model, right. You have to learn a model that can make predictions. But in reinforcement learning, it is one step harder problem.

See because there the data is given to you. Somebody gave you the data. In one step you have to do a lot of learning you can take 15 days to learn whatever but no more data is coming in. This is the data you are going to work with. Your job is to get the best results possible from this data, whatever you can learn from this data. That is what most of machine learning is today supervised machine learning.

However, or even unsupervised machine learning, most of it not all of it there is one branch of machine learning, which plays this long-term game. I do not know if you know about it. That branch of machine learning is called active learning. So check it out. If you want to learn more about this check it out. The machine learning algorithm says give me the label on this data point.

So there actually the assumptions are relaxed. But for most of the popular machine learning, you give me the data, I will just train my model and the model will make predictions. That is it. But the full blown version of the problem is that nobody gives me any data. I am myself, I wake up, now suddenly, I have to make sense of the model. And I may be given some actions. And those actions I am allowed to take, right?

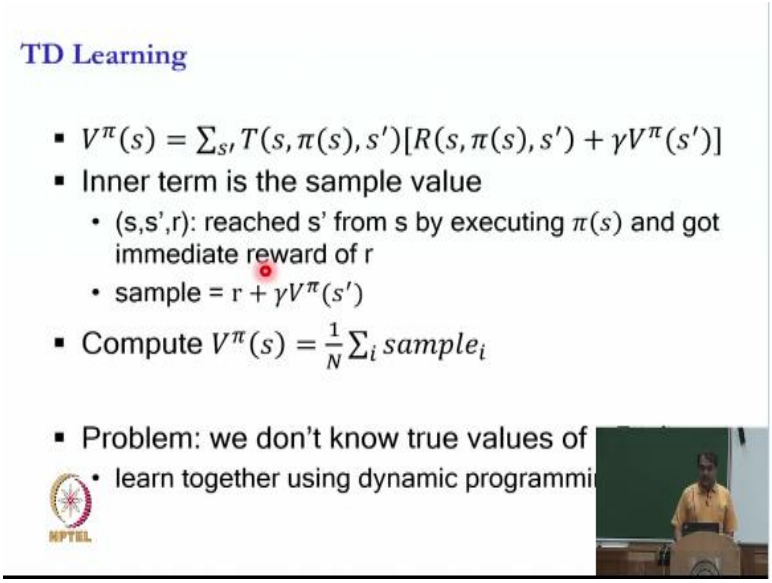
And I take an action. And since something happens, the world changes, and I get some reward. And when that happens, I learn something bit by bit about the world. And then based on that, over time, I have done enough learning so that I can try to optimize my rewards, right? Initially, I do not even know what the rewards are, how the world operates. Later, I know I have some model and I want to optimize my rewards.

So this is a very generic version of the problem. And this is called reinforcement learning and we defined it last class. And we have been talking about one aspect of it, which is policy evaluation. So in policy evaluation, you give me a policy ahead of time. So I know exactly which action to do in which state and then my job is only to compute V pi to estimate V pi right.

So to figure out what is the value of long-term expected value of taking or following the policy pi starting any given state. So this is where we were and then we also did learn that you know any expectation can be approximated by an average of a samples b we decided there are two types of learning versions one is model-based one is model-free and model-based I actually estimate the transition probabilities in the reward function and in model-free, I do not.

And so we are now within inside the model-free version where we have defined an equation and let us just quickly remind ourselves what that equation is. And this method is called a temporal difference learning method.

(Refer Slide Time: 03:57)



So the idea is that this is the equation that I want to compute except that I do not know the reward function and I do not know the transition function, right. If I had known it, I would have just been able to do you know iterative policy evaluation or what not. So what I do is I am in the state s, I take this action pi s. I get to the state s prime. I observe that I have gotten to the state s prime and I have got some immediate reward r.

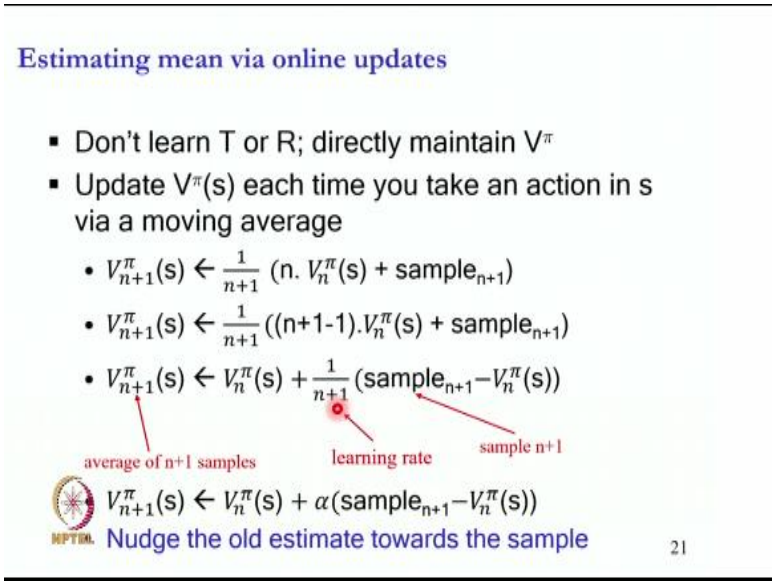
If I am in that setting, then what can I do I compute this value r plus gamma V pi of s prime. Now this V pi of s prime is assuming I have some estimate of how bad or good state s prime is. So this is my one sample. So what is this sample telling me this the sample is telling me that if V pi is the correct function and this small r is the reward I would always get then r plus gamma V pi of s prime should be the value of s, right if s prime is the state I will always reach from s.

But that is not going to happen. So at different times when I come to state s, I will execute this action pi s and I will get different rewards, I may reach different s primes. So I will get many such samples. But notice that all these samples will be got from the sampling distribution T. T is the transition function. So s to s prime will always happen with this sampling distribution.

So because I have to take expectation over the probability distribution T, I can take average over the sample value that I computed every step and so my V pi of s can be computed as 1 by N which is the number of times I reached s sum over sample i,

okay. And even if I do not know V pi of s prime that is okay because we learn together using dynamic programming.

(Refer Slide Time: 05:55)



So the dynamic programming version of this algorithm is directly maintain V pi. Update V pi each time you take an action in status via a moving average. So we have to compute an average. But instead of computing an average every time instead of maintaining all the sample i's, I will do it online. I will do it incrementally. So which means what?

So let us say I already had an average value V pi s n. n means that I have already seen s n times and this is the average that I computed. And let us say I computed a new sample, sample n + 1. So now I have to estimate V pi s n + 1. Everybody with me? That I have an average of n data points. Now I give you the n + 1th data point. I have to estimate the average of the n + 1th or n + 1 data points.

So the new average will be equal to n times the old average plus the new sample divide by N + 1, right. So this is just the standard definition almost a definition, right. Because V pi n times V pi of n is the numerator of the past n samples. Now we will do a little bit of algebra, okay. So what we are going to do is we will convert n into n + 1 - 1, which is no problem. So what we did?

We said n times V pi of n s is nothing but n + 1 – 1 times V pi of n s. They have not made any change whatsoever, but you can see where this is going. So now I will keep

n + 1 aside and -1 aside. So what happens? If I keep n + 1 aside then n + 1 n + 1 get cancelled and I get V pi of n s. And I take - V pi of n s with the other term and so basically I get this particular equation. I say V pi n + 1 s is equal to V pi n s + 1 by n + 1 sample n + 1 – V pi of n s.

Let us think about what this term is saying. So V pi n + 1 s is the average of n + 1 samples. It is saying that the new average is equal to the old average plus some difference and this difference is what is called the temporal difference. Why is it called the temporal difference? Temporal difference basically says that this is what I knew my value to be before taking the sample, before doing any look ahead, and then I did one step look ahead.

So I got one I executed one action I got one reward I got one s prime. Now I know what to what is going to happen. So if I did just one step look ahead this is the value that I estimate. Sample n + 1 is the value that I estimate. But it is not same as V pi n s. That means, there is some difference in what I expected my value to be and what my value turned out to be.

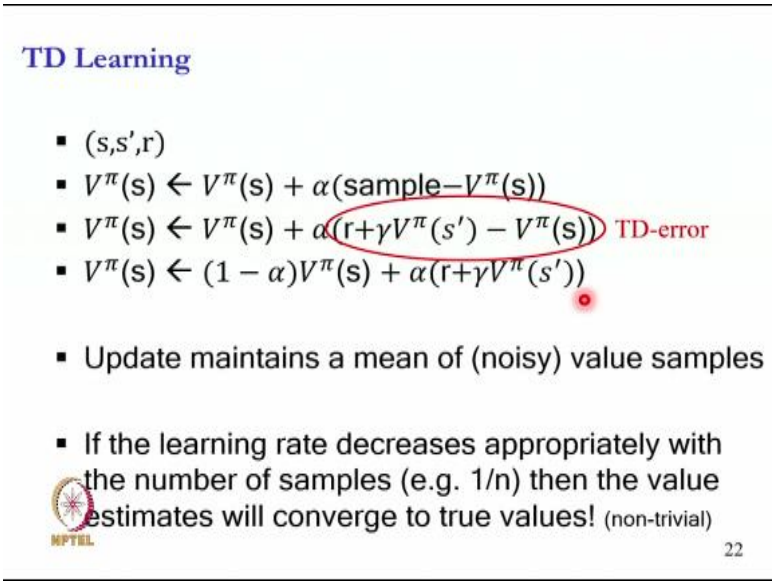
And this is the difference that I need to somehow compensate for. See if I thought my state s is worth 10. And in this run I thought it should be 20 then what do I have to do? I have to increase my average. I have to increase my estimate. On the other hand, I thought the state is as good as 10 points and it ended up in this then only worth 5 points and I have to decrease it.

And in fact, if I know if I have to do the exact computation, I have to decrease it by this difference times 1 by n + 1. This is always maintaining my true average. Now this is however, absolutely correct algebraically correct equation but we can generalize it even further. And this is a non-trivial step. I do not expect you to see it or prove it. But you can equivalently say that V pi of n plus V pi of n + 1 s is nothing but V pi n s plus some general alpha times the new sample minus the old value.

So alpha times the difference, temporal difference. So yes nudge the old estimate towards the sample, but you do not have to always nudge it with the learning rate 1 by

n + 1. You can nudge it with some general learning rate alpha, which is not even equal to 1 by n + 1, okay.

(Refer Slide Time: 10:36)



So the general version of TD learning is I was in s, I executed pi s. I get to s prime. I got an immediate reward of r. And I will be running this equation V pi of s is equal to V pi of s plus alpha times sample minus V pi of s. And you can always memorize this as old value plus alpha times the temporal difference learning rate times the temporal difference. By the way, we have now done learning thrice.

The very first time we did learning in this course, do you remember when was that? Learning the utility value in the minimax algorithm. And that is where I told you that all learning algorithms look the same. They are new weight is equal to old weight plus learning rate times some new term, some term. If you do not remember this, go back and check your notes.

This is what all of learning is and everything is about what is that term multiplied with the learning rate that is the interesting part of all learning algorithms, almost all learning algorithms. So this is the same form. It is saying that I am trying to learn this value. Think of it as a weight. And the new value is going to be old value plus learning rate times something.

And this something in the case of TD learning is the new sample minus the old value. And sample we can always substitute. So it is basically V pi of s plus alpha r plus

gamma V pi s prime minus V pi of S. So the new sample is r plus gamma V pi of s prime. And this is as we said is called the TD error, the temporal difference. The error, why is this the error because this is what I expected, but this is what I got.

So the difference between the two is sort of the error that I need to nudge my weight towards my value towards, okay. And, by the way, equivalently, we can write this down as 1 - alpha V pi of s + alpha times r + gamma V pi of s prime. So you can think of it as an average where the new sample is worth alpha and the old sample is worth 1 – alpha.

So it is like I am maintaining the average, but new values are more important to me and the old values are slowly going to become less and less and less amount or vice versa depending upon alpha. So essentially what happens the update maintains a mean of the noisy value samples. Why are these value samples noisy? They are noisy, because even this V pi of s prime is not correctly known.

We are also learning that. So when we reach s prime we will update s V pi of s prime and we reach then when we reach s, and now we have new V pi of s prime so then that will change V pi of s. So that dynamic programming behavior of a Markov decision process is going to maintain. That the successor is going to back send the value to the predecessor. That intuition is always going to stay, right.

But we are not doing it in an experiential fashion. And moreover and this is very non-trivial but if the learning rate decreases over time appropriately. So appropriate term is very important. And I will explain what that means in a few slides from now. For example, 1 by n. So 1 by n, we have already proven that if it is 1 by n, then it always converges to absolutely the true average.

But even if it is not 1 by n, but it is similar to 1 by n and we will define what that means, then the value estimates are going to converge to the true values. So this is the general form. Anyway learning that need not be 1 by n, it can be something else. And as long as it satisfies some properties, it still converges to the true values. So this is the TD learning algorithm, okay.

And this is one of the sort of fundamental algorithms in the field of reinforcement learning and this is what we are going to use to actually solve the full blown model-free RL problem.