NPTEL

NPTEL ONLINE COURSE

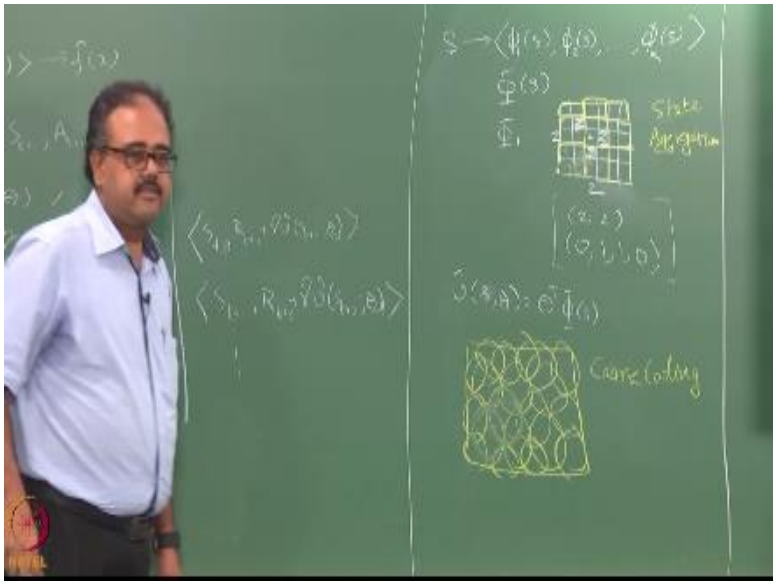
REINFORCEMENT LEARNING

State Aggregation Methods

Prof. Balaraman Ravindran
Department of Computer Science and Engineering
Indian Institute of Technology Madras

Let us see how far we go okay so one of the most a look-up tables right I tell you that lookup table is a linear function approximated we agree with me or not right how what will be my Ø will be Ø is indicator function right Ø I(s) will be 1 if s = si right so that essentially becomes a linear function approximate right so well you do not want to use so there is a useless function approximate or but just useful to think about it that way now I can define another form of indicator function approximate right known as state aggregation.

(Refer Slide Time: 01:16)



So what is state aggregation let us say that I have a very large state space a very large states phase I can typically assume that states that are close by will have the same value right suppose I

have million by million grid world right if I assume that states that are say within 10 of each other right like a 10/ 10 grid I can take any 10/ 10 grid and the states will have the same value within the 10/ 10 grid right that is you are moving into nearest neighbor kind of a setup we are not talking about that I am just saying that given that I have to reduce this is I have a million/ million right.

So how much is my value function going to change if I move 10 states in one direction I have like another 999000,990 states there to go I am trying the dimensionality I am trying to reduce k right so if I had done a lookup table right by K will be a million square right I am trying to reduce the dimensionality of that right I am going to say that way this is it is another kind of indicator function instead of saying that it is exactly this state I am going to say it is one of these 100 states still on call is here I am the samples are still on policy rate the sample generation is still on policies I am just sampling a trajectory right following π it is not like I am jumping around and sampling things there is some amount of reduction of information here right but still the sampling is still done on policy right.

So I can use this indicator function which a set I am in this big grid right so I can take this large okay let me take this large grid world right it has to have odd number of now I break it up into instead of having 30 indicator functions I am going to have only 8, 9 indicator functions 30 indicator function I have 9 indicator functions right so this kind of an approximation is called a state aggregation this kind of approximation is called state aggregation because it is talking about grid words I am drawing squares right.

If I am talking about say continuous world right so for example talking about like robot navigation problem or a manipulator looks like that where my states is given by joint angles or something like that and I am going to have a continuous state space right so I can talk about any kind of compact state right on in that state space and I can talk about memberships of that so for example I can take this is a continuous state space right I can start talking about and I am going to put this kind of circles on the on the state space right.

So all states here this is an indicator function basically for all the states here this will be one for all the states here this will be one for all the states here this will be one so what is the problem with this it is not covering the whole space right so typically what happens is gender actually covering the whole space what is the problem with that overlaps so what is a big deal with overlaps two indicator functions are going to come on instead of one better why the generalization would be better.

See earlier I was telling you when we did the grids right when we did the state aggregation I was telling you that any block of ten states I will take it will have the same value right but that is not true in the representation I am using here because that is going to be discontinuity here right these four states will get one value because the average of these 4 that is essentially what will happen right it will learn to learn to be the average value if you look at how the update happens right.

So every time I come I will update the same value right so it will learn to be the average of this 4 and this will learn to be the average of this 4 but these 2 which are actually right next to each other we will have a drastically different value right when I start doing this overlap so what will happen is well these will get updates together likewise this will get updated together but the states here will actually get updated for both essentially to convert to the average of both right.

So it will not be a abrupt shift from one to the other do you be more like a well step okay not be a graded shift it is based like this I will be one intermediate value which is the average of both and then another one so it will not be a complete jump it will be a gradual jump right so this is okay to have this kind of overlaps in fact it could you can argue that it is desirable to have these kinds of overlaps right and this of and again we need nothing here for nothing fancy here all of these are indicated functions to accept that depending on the degree of overlap that many indicator functions will come on for a state that could be one it could be two it could be three it could be more right.

Why computer what domain knowledge intuition carrot cards I mean this is why deep learning is becoming popular nowadays because you can use deep learning instead of the carrot cards right

so you have to use some kind of in this case I am assuming all of this or Geographic I mean geometric sorry geometric problems right so I am assuming nearness in the particular metric space I am looking at indicates nearness in the value function it will not always be the case right so there are people who actually look at so the way to address this is to take the state space that you have embedded in some manifold.

Where the nearness in the distance actually means that you are near in value function as well right so there are many methods as people have used that they try to come up with different kinds of basic functions which have that can also be the basis function would be what are my basis functions here this particular case whatever we are talking about what are the basis functions yeah the Ø1, Ø2, Ø so I am using some K basis functions here the people have come up with different ways of transforming this basis.

So that nearness in that transformed space right indicates nearness in the value function as well right so this is an example that I was giving yesterday it so take a grid world or take some space like this right let us say I have a some goal there right and same a parameterization consists of the X and the y and for every time step I have not reached the goal I get a reward of -1 right so how will my value function look like well in the negative direction assuming the negative direction comes out of the board right so it is going to look like this rate so it is going to slide all the way in so the peak will be somewhere here right and then from here there will be gradual decrease all the way to this point okay.

Right so that is a nice smooth value function right in fact I can think of depending on my dynamics and stuff for stuff like that that could even be linear right so it could be just a plain tipping so it could be a plane that is tipped over like this right that should be fine take appreciate is like this and then I can probably represent using a linear parameterization great now what I am going to do is I am going to put a wall know what is going to happen to this there will be some kind of a discontinuity so where will the discontinuity be so here will be nice and small thing here it will be a much larger value wait.

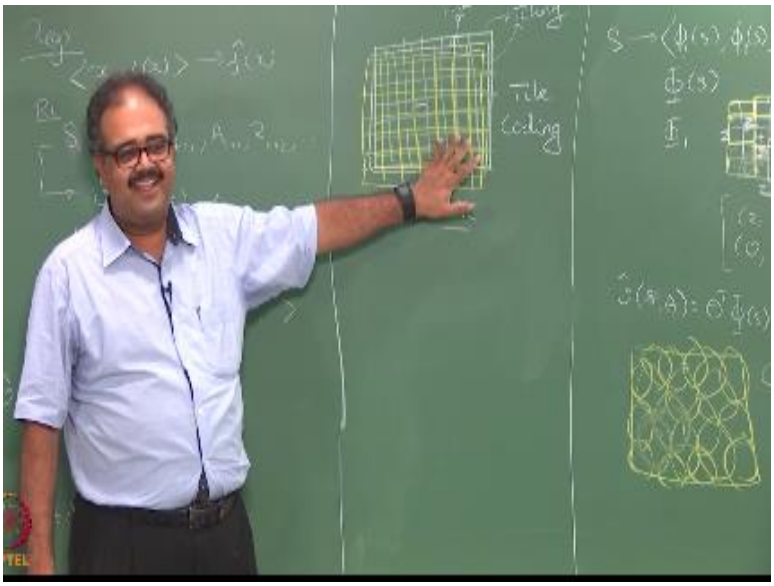
So one way to fix this is to think of some kind of an embedding where I essentially split the room open like that right so I essentially think of in the energy that is what it is right if you think about it and now it is in fact I made it into some kind of a Queasy corridor by drawing the line there right so I can think of transforming a space into something like this now if I look at the XY coordinates here right it might be a good enough approximation for the value function right so it could be a good enough near in the system it I mean obviously this is not what you have to do you have to look at a more complex embedding just trying to give you some intuition as to what you try to do when you find those embeddings okay.

You try to get a transformed space so that in fact if you think about it so my y-coordinate runs like that here so it is not a proper linear thing so this will be my y-coordinate right so something like this will allow to think about so it is a complex thing so what people do work on stuff like this I they look at all kinds of different basis so that you can do this kind of generalization at the end of the day so we will not worry about then let us stay in I say in a more innocent world for the time being where we assume that this kind of a generalization is fine without having to worry about any other basis transformations okay.

So this is called course coding so I am doing some very coarse representation of the state right sometimes called course coding so the problem with course coding is that the there is no uniformity in the number of bits that I am using for my number of on bits right I am using for representing a state so some states could have only one bit some states could have multiple bits and so on so forth and so you could people came up with a more systematic way of doing this course coding okay.

And as many of the things as described in the book this is not the way they came up with whatever I am going to talk about right so they do something called tile coding so then this is going to use this space because it is the last thing I am going to do today so I will stop with this.

(Refer Slide Time: 14:32)



So let us consider some input space right so tile coding that is the following is flicks it up into a kind of regular tiles okay but this is not giving you that nice overlap features at the course coding had it is just almost like state aggregation so what tile coding additionally does is that it does not do one such grid it does multiple such grids it is got really confusing where it is very simple if you think about any state any point in this state space it will always light up two of the indicator functions one from the green grid one from the yellow grid right.

And it gives you this overlap that you wanted right so all the states in this yellow grid will get updates from this place in all the states in this green grid will get this update from this state so in some sense you get the overlap right so there is some set of states here which contribute to this which do not contribute under the green credit so there is some amount of overlap that happens from one cell to the other from the green to the yellow and vice versa it gives you the advantage of course coding.

It so it is a form of course coding but it is more systematic because every state will have two indicator functions lighting it right so each one of these is called that tiling right so that is the tiling and that is a tiling and each one of this curve any guess so each one of this is tile and the

whole grid that we talked about is called the tiling so in this example I use two tiling's people typically use like anywhere between four to seven tiling's to get good performance right and so how many indicator functions light up for a given input state number of tiling's right so has many telling us there are so far seven tiling's there will be 7 things at later right.

So it turns out to be a very good function approximate for many RL tasks that people were looking at about 2, 3 years ago till about 2, 3 year ago so this is sent out to be a really good so obviously you cannot play go or backgammon using tile coding well you need something more clever than then it turns out that you need something more so that is the biggest advantage of all this deep learning methods is that you do not have to define the Ø they find the Ø from the data that is the biggest advantage of many of the tile coding methods is it fine though.

What are the appropriate Ø from the data itself and then uses that Ø to define a value function and then learn short value function everything is done in one short it is not like I first learn the Ø and then learn the value function or anything is a single network as I keep training it learns a Ø and the value function also the amazing thing looks like we are getting a lot of advantages for Ø what is the downside you need significantly powerful computing equipment first otherwise it is not going to run anytime soon and you need a lot of data.

So that is the sum of the biggest challenges so tile coding and one popular instance of tile coding was a specific thing called CMAC so CMAC stands for you should tell me now cellular model actuator controller right so essentially it was originally proposed as a model of the cerebellum I mean a human cerebellum right and so from there cerebrum cerebellum and then from there I mean of course it is I do not know if people still consider.

It as a serious model of the cerebellum but this comes from way before the 60s 50s and 60s okay this is a guy called Albus who proposed the CMAC and it turns out that CMAC to some kind of tile coding and then they do a whole bunch of tricks to compress the tiling's right so nowadays with a lot of memory at disposal you do not would not want to compress the tiling's but the classical CMAC implementation uses a lot of hashing based tricks to compress the tile tiling representation.

So that you do not have to store the entire vector θ if you think about it suppose each tiling is 10/ 10 right so I have number of tiling's in 200 T θ's head and repair tiling's become more and more and more finer right and state space is very large and I want to keep smaller tiles right then the number of θ is that I need to store becomes larger and larger I do not actually have to store the tile stem cells because they are so regular.

I can have some kind of a function right that computes the tile memberships I can have some kind of I can have a functional form that will tell me which tiles come on right but I have to store the θ at the end of the day this is also a look up table right except that I determine the membership which entry I look up in the table in a very complex systematic but complex way and instead of just looking at the index I so people use hashing and you know you know you can you can use hashing to compress tables.

So just like that they use hashing to compress the entries in this and so it has advantage sand disadvantages in fact it gives you unintended generalization right because two different tiles could be hash to the same weight right so one tile from here right one tile from here another tile from here could hash to the same weight that gives you some kind of destructive generalization right but then there are some conditions under which you can show that this is not too bad and so on so forth.

But I personally do not like hashing but we have found out that in some cases from very large state spaces as having hashing actually helps you to learn faster so you really have not managed to explain why that is the case book right so I will stop here then so we will talk about eligibility traces and control and other things in the next class.

IIT Madras Production

Funded by

Department of Higher Education

Ministry of Human Resource Development

Government of India

www.nptel.ac.in