Modern Computer Vision

Prof. A.N. Rajagopalan

Department of Electrical Engineering

IIT Madras

Lecture-17

Then the other thing right that let us talk about is actually adjusting the learning rate alpha. So, can we even can we even get a see tweak that learning rate alpha. So, what this actually means rate is the following. So, for example, should I just keep an alpha which can see for example, right I mean what kind of see choices do you have. The choices that you have is keep L R is let us say learning rate keep it a constant keep L R constant. It does not it does not seem to be a bright idea right keeping this alpha alpha L R by mean I mean this learning rate or reduce alpha after you know every let us say write n number of iterations or something.

But again nobody knows how many iterations for a for you know for a certain problem after every n you know iterations. This is to say that right I mean you know if I I mean if I again right I mean it should be a more sensible way of doing right that is what that is what we will eventually see. These are just you know the most naive ways of doing reducing alpha after after every n iteration. This is to say that you know may be may be right as I as I come kind of see closer and closer to the minima I should have a smaller alpha otherwise I might actually overshoot right.

So, this is this is coming from that angle because you might wonder why would I want to reduce alpha and reduce my step size right. That means after I have done done sufficient number of iterations may be I should just write you know make my alpha smaller. So, that my step size is do not become too big. Then you can have exponential d k something like alpha is equal to you know alpha naught let us say write minus minus is k t or something. Then at this at time t equal to 0 you got it as 1 and then may be right you keep on reducing it as t goes on as time increases or I mean right there are there are so many ways in which in which right you can do this but then none of these is really a systematic way of doing it.

These are more like right heuristics even the other ones are not exactly right theoretically proven or anything, but those are far more right insightful in terms of what you can do. So, there is something called well you know do not have to worry about the name, but then it is called you know adaptive it is called adagrad. So, adaptive gradients well the name is

not important, but what is important is this right. So, the way. So, instead of doing all this at what is normally done is this.

So, you have like r t I will explain what is r t this is r t minus 1 plus gradient of theta. Now, when I write gradient of theta you again mean the l with respect to l just a short form I am writing it as gradient theta t it is not gradient of theta it is gradient of l with respect to theta square and theta t plus 1 is equal to theta t minus. Now, this alpha right is changed to something is modified like this delta plus and of course, you know different people there are one or two variants of this right one variation is something like this another variation is delta plus r t this is also acceptable. And this delta is of course, a positive number greater than 0 just to just to avoid you know divide by 0 that is all the role of delta is this r t is what is important because r t r t is this r t delta is only to avoid you know divide by 0. Then you have a gradient of you see theta t right this is our this is the this is our thing right whatever we normally use.

So, what it is saying is you know modified learning rate alpha by this factor right which is alpha by delta plus r t. So, the idea is as follows right now if you look at r t right r t I mean again right I mean taking r t minus 1 and then adding up and so on. So, right it is like saying that you know for example, when you find gradient of l right you know with respect to this unknown theta and suppose suppose right suppose we have certain values. Now, if let us say if now these are your gradients if some of them are small let us say right and if you kind of keep your alpha fixed then basically what will mean is you know when you have when you have to go in go in you know n orientations right n directions. What will happen is in you know in a certain this one direction the step that you will take is going to be automatically smaller because of the fact that the gradient right along that is already small and if you fix your alpha then it will mean that you will just move you know smaller right in that sort of a this one direction.

Whereas the idea behind using this kind of a frame work is to actually is to make sure that if the gradients are small then you actually end up accelerating. So, that your next step size right you know it does not it does not right become too small otherwise what will happen is we just freeze your alpha and suppose the gradients happen to be low right in certain orientations then you will end up taking very very small small step size right along that along that is a directions along those directions. And therefore, right so this term root R T right is always right you know which is actually sitting here and of course, let us assume that R naught is a number greater than or equal to 0 to start with then what will happen is this R T right. So, you can imagine that and you know this is like I say element wise. So, this you know delta theta T it is like I it is like I say vector.

So, when you say you are squaring you are actually squaring right you know element wise

it is not delta theta transpose delta theta it is not that right. So, kind of element wise squaring and what you are saying is for each element right you will try to see how much how much is my this one gradient square and then put that into R T and the idea is that if for a certain this one direction if my say delta theta T happens to be small which I will know anyway because I am actually computing it then what I will do is then automatically I know that my R T is small and therefore, my alpha by root R T will then effectively become higher and therefore, it helps me kind of move forward faster. The only problem with this is that as time progresses right as time progresses then what will happen is this R T right irrespective of or any other thing right it will start to accumulate over a time right because you are doing R T is equal to R T minus 1 plus you see right you know this one delta theta T square. Therefore, over a period of time this R T will will will sort of what you say will will get to a large value and therefore, even if your delta theta T is small because R T is already accumulated right. So, what will happen is your the required movement right that you want will not happen because of the fact that R T is already accumulated and therefore, people do not actually use it exactly in this form this is a modification right that they actually make to R T such that such that right you know you can actually you can you can have a you know a forgetting factor just like you had that forgetting factor right with respect to with respect to V T that we showed earlier similar to that right similar thing is actually employed here, but then here it is just you know convex sort of a combination ok.

So, yeah so, this only thing is right. So, used in this form this grows with time this will grow with time this means this R T this will grow with time and hence and hence the step size will of course, right you can ask I mean why do why do you even accumulate right you can ask why do not we just you know take that this one delta theta T at that time why do they why do they accumulate and then sort of slow themselves down what could be the reason see for example, what if I had written R T is equal to say delta theta T square or something. No, what I am saying is I mean they they use a previous history right in order to be able to go right move. So, I am asking right I mean and because of that the R T gets gets accumulated over time right and and that is the reason why we are saying that as time goes on the R T would have would have become sufficiently big and and and write delta theta T even if it is small right you would not be able to make you know make a large step because alpha by root R T will still be still still not be large enough right. So, this will grow with time and hence the step size will will become smaller will become small and will become small even even if the gradient is small.

Well the idea is that right I mean history is always a good thing to use what what has happened in the past right I mean that is a reason you know this kind of, but then history should not be enforced in a sort of a blind way that is the reason why any kind of exponential weighting is good ok. So, this was the next one right that actually does this is

what is called Adam and it is actually a combination of you know two kind of different things, but let us not worry about names and all right, but just to know that you know there is something called you know Adam ok. This is the more I mean for example, in your deep network at all when you implement right you will say that I used and you see Adam Adam optimizer and so on right and that is this adaptive moments. The moments comes from comes for a certain reason that I will tell you why the use of moments suddenly wonder where is this where is this kind of a statistical quantity where did that enter into the picture, but right that is what that is what is Adam and the way right that works. So, yeah this is yeah.

Sir in which situation is it useful? In which situation is it useful? Previous algorithm means the one the one here ok. So, what we are saying is right. So, if this term was not there and if you had simply alpha times delta theta t and delta theta t is like is actually a vector right theta t plus 1 is actually a vector. So, each element each each of these weights for example, is going to is going to is going to get updated with time. So, what was so what this is saying is that if this if you compute a gradient and if that gradient for certain weights if it is small, but if you freeze alpha right we just use one global alpha for let us say all the weights then what will happen is happen is your step size right will automatically be smaller.

That means, those weights will get will get updated much less because your alpha is fixed. Whereas, if you divide it by root of R t where this where this R t is being accumulated from a history and R t is a function of actually you know delta theta t. Then if delta theta t is small then alpha by that root R t will then be a sufficiently large number. So, effectively you are giving a larger push for only for those weights not for everybody right only for the only. So, for example, right if this weight if this has a gradient that is small then that then the step size that you will alpha that the learning rate you will apply is alpha by root R t.

So, so that weight otherwise it would have taken forever right because the fact that it has a gradient that is very low, but instead of that you just accelerate it. But all this will have still that kind of you know a pitfall that you know somewhere you might actually you might actually cross over and all, but then we are hoping that you know generally a terrain it is not like right you know within the first first few iterations you will get to the minimum right. So, the so the idea is that you apply all this. So, that initially right you cover as much ground as you can and you do not really worry about oscillation and stop everything right. It is not like oh you know what if I have a large alpha and therefore, what if I cross over a minima or something right minima and all you will hit eventually.

So, the idea is that initially right when you want to when you want to make that move you make it make it in a make it in a fairly significant way right that is the idea. Then this

one right Adam the way this Adam works right is as follows. So, the equations are like this. So, it so it has it has two things they are very very similar S t is equal to let us say rho 1 S t minus 1 plus 1 minus rho 1 right which is why I said that you know this is kind of some kind of a convex you know convex combination of and then delta the gradient not not not delta gradient theta t and r t is equal to some rho 2 again I mean rho 1 and rho 2 right can be between between you know 0 and 1. Then r t minus 1 plus 1 minus rho 2 and this is gradient theta t square and theta t plus 1.

So, you can always replace theta t plus 1 think of it as some w t plus 1 or a b t plus 1. So, theta t minus alpha by delta plus root of r t tilde I will tell you what is r t tilde is into S t tilde S t tilde. So, so we try to right compare with the earlier one and then one thing right which I will just leave it to you leave it for you to show is a simple thing right can you can I think it is very easy for to show that r t that you can write it in a kind of form that 1 minus rho rho 1 rho 2 right in this case whatever right. So, 1 minus rho 2 if you wish summation i equal to 1 to t this I just leave it to you then gradient theta t square e raise to sorry rho raise to t minus i no this should be theta i not I mean theta t should be theta I think it is wrong here it should be theta i e raise to t minus i this I will just leave it to you to show it is very simple to show. So, the so again right this again some kind of an right exponential weighting which means that every element will get.

So, for example, if you are at if you are if your i is at t right then this rho t minus i will just become 1 and you will have like you know 1 minus rho times delta theta t square right. But then if you if you look at the past history they will have like rho into 1 minus rho right delta delta theta t minus 1 and then rho into sorry rho square into 1 minus rho and then rho cube into 1 minus rho and because of the fact that rho is between 0 and 1 typically right typically rho will be like 0.8 0.9 or something. And therefore, what will happen is again that kind of a exponential exponentially weighted decaying will all happen right that means the past gains will get you know things too much in the past will get automatically you know weighed you know weighted down.

But then right the only thing ok so in that sense right these things are very very similar to what you have already seen. And then of course, you know you would have wondered you know why do we have an r t sort of a tilde here why could not we have simply had you know why could not we have we have had just r t. Now this r t tilde is actually a function of r t and it is given as r t tilde is given as r t by 1 minus rho 2 to the power t and S t tilde is similarly S t by 1 minus rho 1 to the power t and this is called a bias correction. I would not kind of go into the actual details I mean like I said right we cannot enter into all the details of what is bias correction and so on. Yes, but then just to suffice it to say that that you know see ideally right what happens is if you have to if you have to think of for example, whatever I say the same argument applies for both you know r t tilde as far

as I will say S t tilde.

So if I say that if I say that right delta theta t comes from a distribution a probability sort of a distribution and if this S t is an is actually an is trying to estimate that quantity then an unbiased sort of an estimate right would demand that you know expectation of S t be equal to right expectation of delta or you know not delta this quantity gradient of this right and what you can show is of course, certain certain you see assumptions have to be made here it is not that straight forward, but there is a work that shows that they are this work is by is by actually Kingma 2014 ok this is like 2014 so not that old right. So, so well you can argue in the sense that right whether those whether those assumptions are valid or not, but under certain I would say reasonably strong assumptions ok and I have the I have the derivation, but you know point is not to go through that, but you can show that you know effectively what happens is you can show that under certain assumptions this can be shown to be you know 1 minus rho 1 to the power t right is equal to is equal to expectation of delta of C theta t ok this you can show I mean right I mean you know ideally ideally right you would want this to happen, but then but then right what really happens is I mean if you were to take this equation on its face value then then unless you scale this guy by this by this quantity 1 minus rho 1 power t right you do not really get the get the expected value of this one the you know on the this one right hand side what you have and therefore, right that is the reason why you have a bias correction. So, this 1 minus rho 1 t or rho 2 t well S t is rho 1 right and similarly the same argument holds for if you read that paper right you will know nothing very great about the derivation very simple, but certain assumptions are made ok which which you can argue ok whether whether those are really valid and so on, but if you make those assumptions it it turns out like this and therefore, there is a bias correction which which actually enters you know into the picture delta and all is still the same always a number greater than 0 in order to you know prevent a division by 0 and this is the most most used sort of an optimizer right. So, if you see right I mean you know you know it has everything it has a momentum right which comes through this right and then and then you know you know you know and then it has a learnable alpha right which comes through this where this R t tilde right I mean you know is you know is happening through this and because of the form that this S t has taken right you need you need this kind of you know a bias correction for both S t tilde as well as as well as you see R t tilde and the same argument the same thing if you simply assume that right expectation of R t should be equal to you know expectation of you know delta delta theta t square delta delta gradient theta t square then then then same thing whatever is a derivation you follow for this right I mean you will you can also follow for this I mean same same steps if you do right then you will arrive at this bias correction ok. Now, this is what you will normally use Adam is the most most used thing ok.

Now, this is this as far as the optimization part is concerned right there is also something

called a regularization right and regularization comes in when when you are actually worried about worried about a generalization right I mean optimization is all about speeding up and so on and when and how do I get there faster and so on regularization is something else regularization is about you know I have trained over a certain number of examples and how do I make my network network sort of generalized outside of that set right and if you if you if you can typically do a training. So, so right now that now that you understand what you see epochs are right and as the as epochs go up. So, what will happen is you know when if you look at your training training error right when we say training error you mean the L theta that you are computing over your see training examples what will happen is right it will it will start falling right I mean after after sufficient number of epochs you may really you know go close to 0 ok it can happen. But what can also happen is so what basically so you know so even in your own training right what you will do is there was also a validation that you do what this means is there are a certain bunch of examples that you do not use for training just use them for you know validation that means you want to know how well how well you might end up doing if you tried this network on examples outside of the training set right. So, if you plot that right there again now what will happen is you know so you will have something like this and then beyond a point it can happen that your that your validation error starts to grow go up ok this is something that you will normally see.

So, what this means is that see as your as your as your as your as your epochs keep increasing right you will first epoch 10th epoch 100th epoch and so on then what will happen is your you see training training error will start to go start to go down because you are forcing it to be as small as possible right I mean you know right that is a whole idea I told you gradient descent is actually greedy right it will never allow for an increase in the error. So, you so it has to drop how far it drops is we have to see, but then it has to drop it can never increase, but what can happen is your validation error right. So, for example, right I mean see this point right up to this point it looks like well I may I may my training error is not down to 0 definitely, but but but then on the validation side there is of course, a gap right I am not doing as well as I am doing on my training set, but still right I seem to have learned something right in the sense that if that error margin is acceptable I will say it is ok, but then beyond that right if I if I if I try to push my network to really do very well on my training set I might actually end up right over fitting to that and if that over fitting happens then what will happen is on the validation error you will start you will start actually right on that part you will be even right network will start to falter and and and this error will start to grow right. Therefore, it is never a good idea to actually push your training error to 0 you may think that why I am doing a fantastic job on my training example, but that is not that is not the idea at all right. So, regularization is something that is about generalizing outside the set.

So, even if you do not do so well on your training set you can still be ok with it as long as the validation error gap is not is not of course, right too high otherwise of course, you know you will have to go down until until right that error becomes acceptable. Therefore, therefore this regularization you know can be kind of say done in so many ways which is actually which is to which is to which is actually to to to increase the increase the generalizability or generalizing capability of a network generalization capability of a network that is the goal. Whereas, optimization was all about speed up generalization capability of a network and such a thing is called actually actually early stopping early stopping is one way to actually to actually achieve this. That means, you stop off early right you do not you do not you do not you do not keep on going ok. So, early stopping is one way to actually regularize in the other way to regularize is data augmentation.

Data augmentation actually means that right you may have examples with you, but then you can create more examples out of those examples which means that you can bring in variations that you know that that will actually help your that will expose a network to even other kinds of examples. So, for example, right you can have data augmentation that can come in a geometric way or you can have data augmentation that can come in a photometric way. The geometric way means that I take an image right which is which is a training example for me, but now if I could have if I shear it I kind of say rotate it right and do and I do kind of right all kinds of things. I mean if it is a classification problem it is easy to tell know even if I shear it it is still the same object right if I if I rotate it I still want it to be the same object I may if I do so. So, there are bunch of bunch of geometric things right which you can do geometric is a transformation which you can do there are bunch of geometric transformations which you can do on on your existing examples right you are you are just generating more examples from what you have it is not like somebody is given you these you are generating more just because you want your generalizability of the network to increase.

Because the network has could not have probably seen these examples you are trying to show it examples of that kind other could be photometric transformations right. Photometric means you can actually increase I mean you can do something to the intensity themselves the image remains the same you are not rotating shearing or anything, but then you could add noise to it you could actually blur you know you could actually add some blur to it or you can add some shadow effects to it you can do things to simple things that are at that are at the photometric level which means at the you know intensity level and these are photometric transformations. So, these are of the kind. So, here right this could be rotation, translation, shear, whatever right I mean you can think of so many ways in which you can actually effect scale right you can do all of that and in you know every time you still want a label to be unchanged. Photometric transformations you can you can think about noise adding noise by which noise means I mean adding noise jitter, blur, shadows

whatever you know right illumination gradient on the image you can you can if it is easy to do right if it is too complicated then probably you do not want to try it, but these again right this again is again another simple way by which you can in fact, these are things that let us say people commonly do ok.

In fact, in fact, this kind of a thing is what is also used for what is called a contrastive learning because even after you do that right it still remains a positive example and that you know a priori ok. Then the other thing right is about is about you know the weights themselves right. So, it is about having a formal formulation where you say that your L theta right is ok. Now let me say this L till till tilde theta is your original theta plus let us say let us say let us not use alpha let us say some other you know because let us say beta norm w square right. That means now there we explicitly want your norm of you say w see till now we had we had used we had used an L theta right that was only cost that we used.

Now you can throw in additional terms right which which specifically address the weight part which means that which means that you do not want suppose suppose I start with some sort of a network right with certain layers and weights and so on. It does not mean that you know I am actually ok with with all the weights being large and all that why should that be the case right. Why not I actually make sure that that there are only those weights that are really needed to do my work should be out there and the end of and you know let us not have the weights blow up because all that will mean computationally that I will have to store so many weights and then I will have to use all of them and then therefore you might want to look at look at of course this can also be an L 1 norm depends ok. The simplest form is actually having an L 2 norm on norm on this one w 1 so that you get actually a modified cost ok. This I will talk about in the you know next class as to why this right what is the what is the implication of this you know with respect to the eigenvalues of the of the of the say hessian of you know hessian of of this guy right I mean you know dou L theta by by this one dou theta and we will see ok what what that means and this is also another way what is called a drop out which is again another form is regular.

So there are now 2 3 things which we will see in the you know next class.