

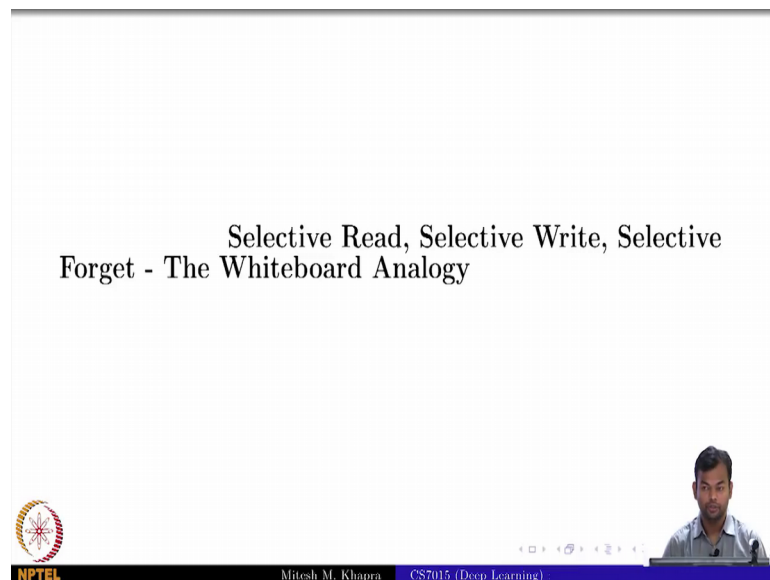
Deep Learning
Prof. Mitesh M. Khapra
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 14
Long Short Term Memory Cells (LSTMs), Gated Recurrent Units (GRUs)

So, we have been looking at a new kind of or a different kind of neural network which is recurrent neural network. And last class we spent some time on showing that it is how to train these recurrent neural networks because of the specific problem of exploding and vanishing gradients. In particular we saw that if you want to propose, if you want to back propagate the gradient from time step t , the final time step to an arbitrary time step k then you have this multiplicative term in the back propagation which could explode or vanish.

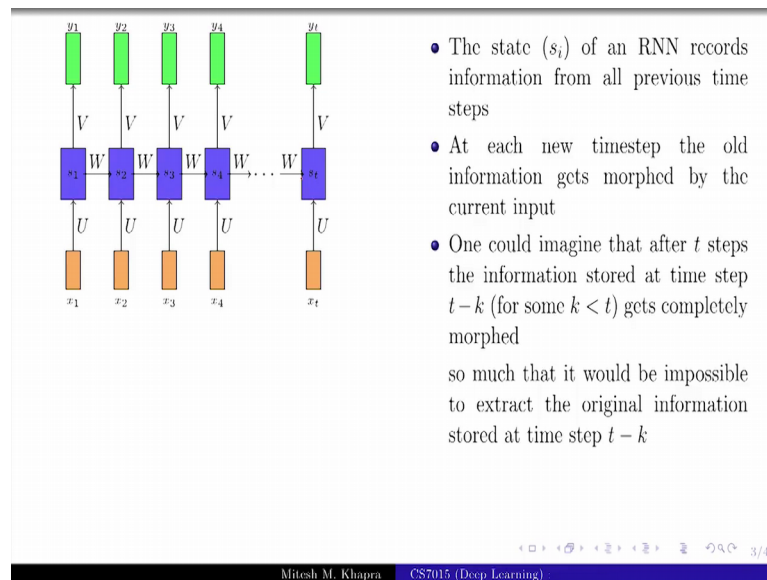
So, today we are going to try to see if we are trying to focus on something which can help us solve this problem to whatever extent, right. So, that is why we will look at LSTMs, a Long Short Term Memory Cells and Gated Recurrent Units, ok. So, let us start with that.

(Refer Slide Time: 01:02)



So, first we will introduce the idea of selective read, selective write and selective forget and then we will try to build on this intuition and see how could you could realise it by using LSTMs and gated recurrent units and whether that help in solving the vanishing and exploding gradient problem.

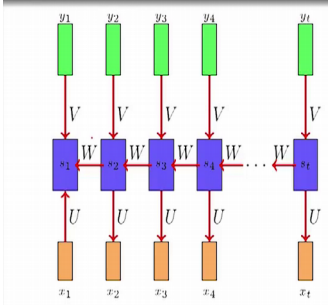
(Refer Slide Time: 01:18)



So, recording the state of an RNN records information from all the previous time steps that was the whole idea that you add this recurrent connections. So, as you keep going on at this time step the cell is not only recording information from the current time step but it also has some kind of an accumulated history from all the previous time steps, right. But now the issue is that this state or this blue coloured vector that you see is going to be of some finite size, right we will say that, it is a 100 dimensional vector or a 1000 dimensional vector but whatever be the size it is going to be some finite dimension.

Now, as you keep writing information to that cell you are morphing the information that you had written at the earlier time steps, right do you get that. So, now, that is the problem, right. Because on one hand you are saying that you want to record the information from all the previous time steps, but the at the same you at the other hand you just have a finite amount of memory to deal with. So, it is bound to read over written and the information will get morphed so much that it is completely impossible to say what was the original contribution at time step 1 or time step 2 once you have reach some time step 20, 30 or so on, right. So, that is the problem with recurrent neural networks. And we will tie it back to the problem that we had with the vanishing and exploding gradients, right.

(Refer Slide Time: 02:33)




- A similar problem occurs when the information flows backwards (backpropagation)
- It is very hard to assign the responsibility of the error caused at time step t to the events that occurred at time step $t - k$
- This responsibility is of course in the form of gradients and we studied the problem in backward flow of gradients
- We saw a formal argument for this while discussing vanishing gradients

Mitesh M. Khapra CS7015 (Deep Learning) 4/43

So, in fact, the similar problem occurs when you try to when the information flows backwards during back propagation, right. It is very hard to assign the responsibility of the error caused at time step t to arbitrary time steps before it, right to very far away time steps it is very hard, and that is the vanishing gradient problem, right. Because we have this multiplicative term and the gradients vanish, so that that is very hard to do. So, both during forward propagation the information vanishes and even during backward propagation the information vanishes, right. And we saw a formal argument of this while doing vanishing gradients. So, this is just an illustrative diagram, but we also saw that formally the guardians do vanish under certain conditions, right.

(Refer Slide Time: 03:11)




- Let us see an analogy for this
- We can think of the state as a fixed size memory
- Compare this to a fixed size white board that you use to record information
- At each time step (periodic intervals) we keep writing something to the board
- Effectively at each time step we morph the information recorded till that time point
- After many timesteps it would be impossible to see how the information at time step $t - k$ contributed to the state at timestep t

Mitesh M. Khapra CS7015 (Deep Learning) 5/43

Now, let us see an analogy for this and from here on we will build on some ideas which will help us arrive at LSTMs and gated recurrent units, right. So, the analogy is a whiteboard. So, you have a whiteboard and it is always of a fixed size, right I mean the whiteboard is not in fact, you just have some size for the whiteboard and you keep writing information on that. So, at every time steps I keep going and writing something on the board and I am trying to derive something or just try to make a story or anything, right. I just keep trying to write information on the board.

Now, since the whiteboard is fixed size at every time I am essentially morphing the information which was written at the previous time step and after many time steps it would be impossible to find out that now whatever my state of the board is how did time step one contribute to that state, right. Because, it is written all over the board I do not know where I started what I did and so on and its going to be very hard for me to find and this happens, right. When you do these long derivations on, on the whiteboard it becomes very hard to track, where did I start where was this variable defined and so on, right because it is a fixed size you cannot really I will end of deleting some terms and so on, right.

(Refer Slide Time: 04:17)



- Continuing our whiteboard analogy, suppose we are interested in deriving an expression on the whiteboard
- We follow the following strategy at each time step
- Selectively write on the board
- Selectively read the already written content
- Selectively forget (erase) some content

Mitesh M. Khapra CS7015 (Deep Learning)

So, and we will make this more concrete with the help of an example, right. Suppose I am trying to drive an expression on the board and typically what we do is we use 3 things, we use selectively write on the board, selectively read the already written content and selectively forget or erase some content. So, let us see what I mean by these 3 ideas, ok.

(Refer Slide Time: 04:39)

$a = 1 \quad b = 3 \quad c = 5 \quad d = 11$

Compute $ac(bd + a) + ad$

Say "board" can have only 3 statements at a time.

- 1 ac
- 2 bd
- 3 $bd + a$
- 4 $ac(bd + a)$
- 5 ad
- 6 $ac(bd + a) + ad$

Handwritten notes:
 $a = 1$
 $b = 3$
 $ac = 5$
 $bd = 33$
 $ac = 5$
 $axc = 5$
 $b = 3$
 $b \times d = 33$

Selective write

- There may be many steps in the derivation but we may just skip a few
- In other words we select what to write

Mitesh M. Khapra CS7015 (Deep Learning)





So, first we look at selective write. So, this is my problem this is the derivation that I want to do on the board and I have a very small whiteboard which allows me to write

only 3 steps, ok. That is the situation in which I am operating, ok. So, I am given the values of a b c d and I want to compute this expression $ac + bd + a + ad$, ok. Now, the first thing that I am going to compute is ac , right that make sense because that is the first term that I need. So, I will write ac equal to 5, and the second thing I will write is bd equal to 33, ok. So, now, why did not I do the following? I could have done this a equal to 1, c equal to 5, then a into c is equal to 5, then b is equal to something, then b into d is equal to something.

So, what am I doing here while writing on the board? Why did I write only two steps? Because I know I have a finite size for the whiteboard, right. So, I am only trying to write information which is important, right I am not writing everything because I know that I am going to run out of memory, right. So, that is why I did not write these intermediate steps that a equal to 1, c equal to 5 and then a into c equal to 5 and so on, right. I juts wrote the results which are important. So, I am selectively write in to the board because I am dealing with a finite size memory, and this is exactly what we do while writing in a note book or a whiteboard or anywhere, right. We just do not write everything one because you are lazy but second is also because we do not have enough space, right. So, that is about selectively writing.

(Refer Slide Time: 06:02)

<p>$a = 1 \quad b = 3 \quad c = 5 \quad d = 11$</p> <p>Compute $ac(bd + a) + ad$</p> <p>Say "board" can have only 3 statements at a time.</p> <ol style="list-style-type: none"> 1 ac 2 bd 3 $bd + a$ 4 $ac(bd + a)$ 5 ad 6 $ac(bd + a) + ad$ <p style="text-align: right;"> $ac = 5$ $bd = 33$ $bd + a = 34$ </p>	<p>Selective read</p> <ul style="list-style-type: none"> • While writing one step we typically read some of the previous steps we have already written and then decide what to write next • For example at Step 3, information from Step 2 is important • In other words we select what to read
---	---

Now, selectively reading I already have something stored on the whiteboard. So, this is the state of the whiteboard at this point. Now, at the next time step I need to read some

information, but I do not need to read everything that is on the whiteboard I just need to read some information. What is the information that I need to read for the next time step? bd , I do not need to read ac , right. So, I am just doing a selective read of the information or the state which is already stored on the whiteboard.

Now, what is happened is I have exhausted my space where I had just 3 steps that could be written on the whiteboard and I have done that. Now, what do I do for the next thing? Now, I need to compute ac into bd plus a . I will have to selectively erase, so what will I erase? bd , right.

(Refer Slide Time: 06:44)

$a = 1 \quad b = 3 \quad c = 5 \quad d = 11$

Compute $ac(bd + a) + ad$

Say "board" can have only 3 statements at a time.

- 1 ac
- 2 bd
- 3 $bd + a$
- 4 $ac(bd + a)$
- 5 ad
- 6 $ac(bd + a) + ad$

$$ac = 5$$

$$ac(bd + a) = 170$$

$$bd + a = 34$$

Selective forget

- Once the board is full, we need to delete some obsolete information
- But how do we decide what to delete? We will typically delete the least useful information
- In other words we select what to forget

$$a \times bd + a = 5 \times 34 = 170$$

Mitesh M. Khapra
CS7015 (Deep Learning)

So now as the whiteboard is full and I will have to selectively delete some information and as this obvious in this trivial example that you can get rid of bd because you have already encoded the information in bd plus a . And now the next step which is ac into bd plus a can do on the whiteboard and this is how you keep doing at every time step. You selectively write at every stage.

Here again note that I am not written ac into bd plus a would be something like I do not know what was it 5 into 34 and then that is equal to 170, right. I am not using two steps I am just writing everything in a single step because I do not have enough space, right. So, selectively writing, selectively reading, and selectively forgetting things which are there in a constant memory is something that we do regular, right.

(Refer Slide Time: 07:30)

$a = 1 \quad b = 3 \quad c = 5 \quad d = 11$

Compute $ac(bd + a) + ad$

Say "board" can have only 3 statements at a time.

- 1 ac
- 2 bd
- 3 $bd + a$
- 4 $ac(bd + a)$
- 5 ad
- 6 $ac(bd + a) + ad$

$ad + ac(bd + a) = 181$
 $ac(bd + a) = 170$
 $ad = 11$

- There are various other scenarios where we can motivate the need for selective write, read and forget
- For example, you could think of our brain as something which can store only a finite number of facts
- At different time steps we selectively read, write and forget some of these facts
- Since the RNN also has a finite state size, we need to figure out a way to allow it to selectively read, write and forget

NPTEL Mitesh M. Khapra CS7015 (Deep Learning) 10/43

And then other ways of motivating this; so you could even think of our brain as something which can store only a finite number of facts, right as we keep going or if we will learning more and more things we can only retain a finite number of facts.

And what happens inadvertently is that you erase some of the steps not consciously of course, you do not have a delete button or anything but you erase some of these things that you forget a lot of things which had happened a year back or so. And also at various times if I ask you what was this which I have done in last class most of you forget to do the selective read but that is what you do, right you always do selective forget but that is what we typically do in our brain also, right. Any time when you are dealing the finite size memory you will always have this 3 operations either they are explicit or implicit, but the intuition is that you end up doing this, ok.

So, now, since the RNN also has a finite state size can we do something like this selective read, write and forget, so that one during forward pass even if the information gets morphed. It gets morphed in a principle manner, right. So, even in the whiteboard example I was morphing the information, I was deleting the information written at time step 1, time step 2, but I was being a bit smart about that. I was retaining some good information and only deleting what was not required. So, can we do this in analogy? So, this analogy really sets it up but now the solution is not going to live up to the expectation, but it will have some it will be something in this direction, ok.