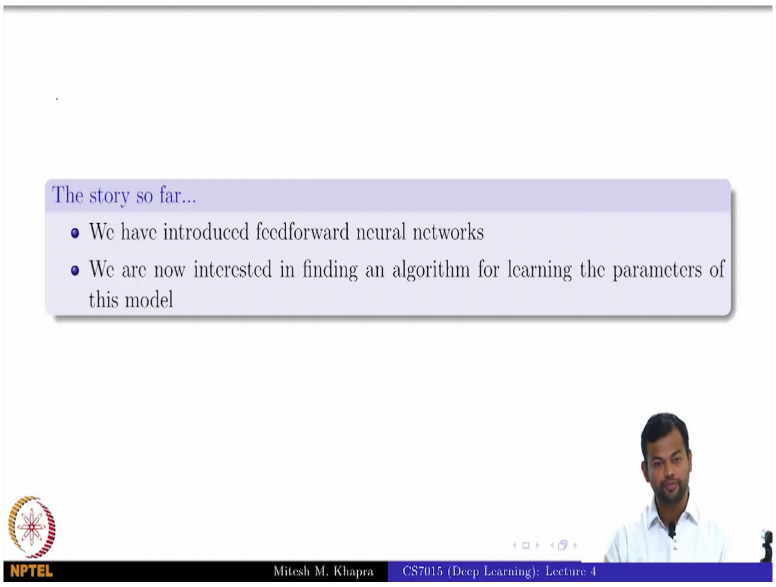
Deep Learning
Prof. Mitesh M. Khapra
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module - 4.2
Lecture - 04
Learning Parameters of Feedforward Neural Networks (Intuition)

Now, we will move on to the next module, where we want to learn the parameters of feed forward neural networks, and we first start with some intuition, and then mathematical details.

(Refer Slide Time: 00:26)

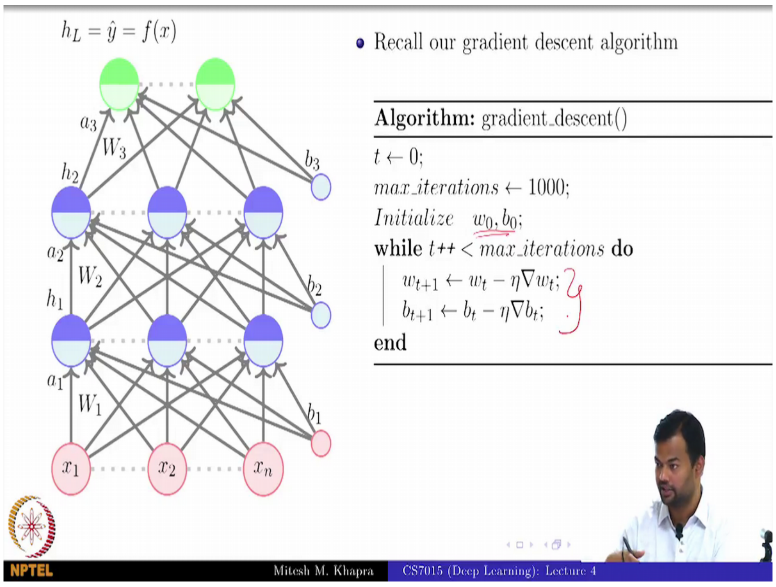


The story so far...

- We have introduced feedforward neural networks
- We are now interested in finding an algorithm for learning the parameters of this model

So, we have introduced feed forward neural networks, and we are now interested in finding an algorithm which can allow us to learn the weights of this network.

(Refer Slide Time: 00:33)



- Recall our gradient descent algorithm

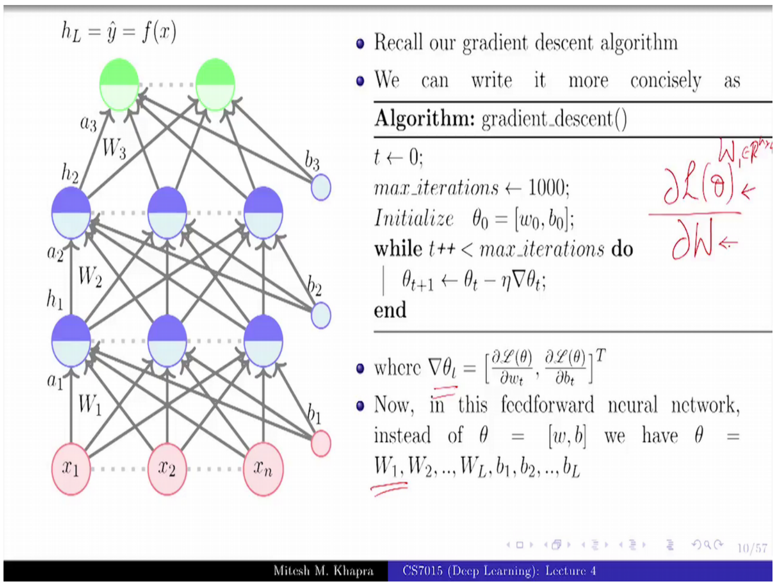
Algorithm: gradient_descent()

```
$t \leftarrow 0;$
$max\_iterations \leftarrow 1000;$
Initialize $\underline{w_0}, b_0;$
while $t{+}{+} < max\_iterations$ do
    $w_{t+1} \leftarrow w_t - \eta \nabla w_t;$
    $b_{t+1} \leftarrow b_t - \eta \nabla b_t;$
end
```

So, recall our gradient descent algorithm, this is how it looked ok, I had initialized those 2 parameters w naught b naught. And then I was iteratively doing this in a loop, at every step I was moving in a direction opposite to the gradient at that step, right?

Now, can I write this a bit more compactly? We can write using vectors.

(Refer Slide Time: 00:59)



- Recall our gradient descent algorithm
- We can write it more concisely as

Algorithm: gradient_descent()

```
$t \leftarrow 0;$
$max\_iterations \leftarrow 1000;$
Initialize $\theta_0 = [w_0, b_0];$
while $t{+}{+} < max\_iterations$ do
    $\theta_{t+1} \leftarrow \theta_t - \eta \nabla \theta_t;$
end
```

$W_1 \in \mathbb{R}^{n \times n}$

$$\frac{\partial \mathcal{L}(\theta)}{\partial W}$$

- where $\nabla \theta_t = \left[\frac{\partial \mathscr{L}(\theta)}{\partial w_t}, \frac{\partial \mathscr{L}(\theta)}{\partial b_t}\right]^T$
- Now, in this feedforward neural network, instead of $\theta = [w, b]$ we have $\theta = W_1, W_2, .., W_L, b_1, b_2, .., b_L$

So, are you if I write it this way? So, these 2 was actually nothing but vector at every point? So, I can just write it this way. So, theta is the vector containing w and b, or theta is the vector of all the parameters my network had it just. So, happened that network had only 2 parameters. So, see where am going with this? How many of you see where am going with this?

Good, so where delta theta t right just to remind you it was this the partial collection of all the partial derivatives with respect to all the parameters. In this toy example all was equal to 2, right we just had 2 parameters. Now you see where am going with this, ok. So now, in this feed forward neural network, instead of theta equal to w comma b what do we have? Theta is equal to so many parameters, ok. So, what would grad of theta t now be partial derivatives with respect to?

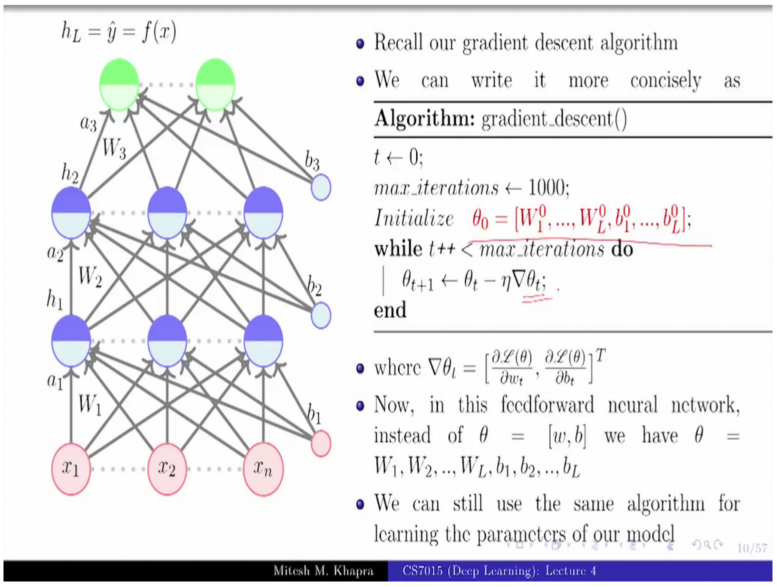
Student: (Refer Time: 01:58).

All the weights, but there is a problem here right, this is the matrix, how do you take the partial derivative with respect to the matrix? Who asked you to use the matrix? How you take the partial derivatives with respect to matrix. So, what am interested in this right the question. I know there is some loss function which is a function of theta. One of the elements of theta has this matrix W 1 which belongs to r n cross n? And now I want the derivative with respect to w. So, see what am trying to do, this is scalar and we take the derivative of that with respect to a matrix. What is all that; the derivative with respect to.

Student: (Refer Time: 02:37).

Every element of the matrix.

(Refer Slide Time: 02:41)



- Recall our gradient descent algorithm
- We can write it more concisely as

```
Algorithm: gradient_descent()
$t \leftarrow 0$;
$max\_iterations \leftarrow 1000$;
Initialize $\theta_0 = [W_1^0, ..., W_L^0, b_1^0, ..., b_L^0]$;
while $t{++} < max\_iterations$ do
    $\theta_{t+1} \leftarrow \theta_t - \eta \nabla \theta_t$;
end
```

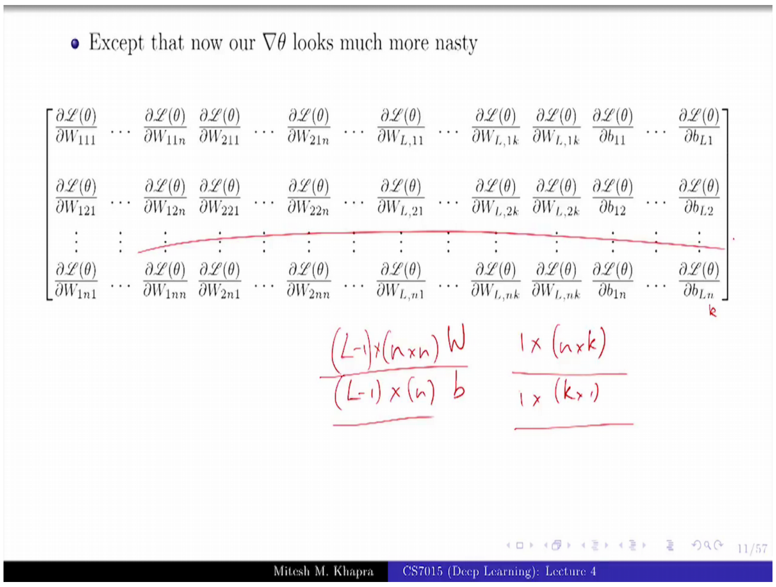
- where $\nabla \theta_t = \left[\frac{\partial \mathscr{L}(\theta)}{\partial w_t}, \frac{\partial \mathscr{L}(\theta)}{\partial b_t}\right]^T$
- Now, in this feedforward neural network, instead of $\theta = [w, b]$ we have $\theta = W_1, W_2, .., W_L, b_1, b_2, .., b_L$
- We can still use the same algorithm for learning the parameters of our model

10/57

Mitesh M. Khapra CS7015 (Deep Learning): Lecture 4

So, we can still use the same algorithm except that del this grad of hat of so now, I could just say that theta 2 hat I mean initialized all parameters and theta not, right? Compute the gradient with respect to all of them and then do this update, right. I could just instead of putting them in matrices, I could just think of them as a large vector just had initially I had just had w comma b, now this vector is even more large. In fact, I will show you actually how it is.

(Refer Slide Time: 03:08)



- Except that now our $\nabla\theta$ looks much more nasty

$$\begin{bmatrix} \frac{\partial\mathscr{L}(\theta)}{\partial W_{111}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{11n}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{211}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{21n}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,11}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,1k}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,1k}} & \frac{\partial\mathscr{L}(\theta)}{\partial b_{11}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial b_{L1}} \\ \frac{\partial\mathscr{L}(\theta)}{\partial W_{121}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{12n}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{221}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{22n}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,21}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,2k}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,2k}} & \frac{\partial\mathscr{L}(\theta)}{\partial b_{12}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial b_{L2}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial\mathscr{L}(\theta)}{\partial W_{1n1}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{1nn}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{2n1}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{2nn}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,n1}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,nk}} & \frac{\partial\mathscr{L}(\theta)}{\partial W_{L,nk}} & \frac{\partial\mathscr{L}(\theta)}{\partial b_{1n}} & \cdots & \frac{\partial\mathscr{L}(\theta)}{\partial b_{Ln}} \end{bmatrix}$$

k

$(L-1) \times (n \times n)\ W$ $\qquad 1 \times (n \times k)$

$(L-1) \times (n)\ b$ $\qquad 1 \times (k \times 1)$

Mitesh M. Khapra CS7015 (Deep Learning): Lecture 4

11/57

So, this is the grad with respect to theta looks very nasty now. This is how nasty looks, right? So, you have this weight matrix W 1, you have the derivatives with respect to first element of W 1, all the way up to the last element last element; so with respect to all the n cross n elements of W 1. What is the next entry going to be w 2 1 1 2?

Student: (Refer Time: 03:32).

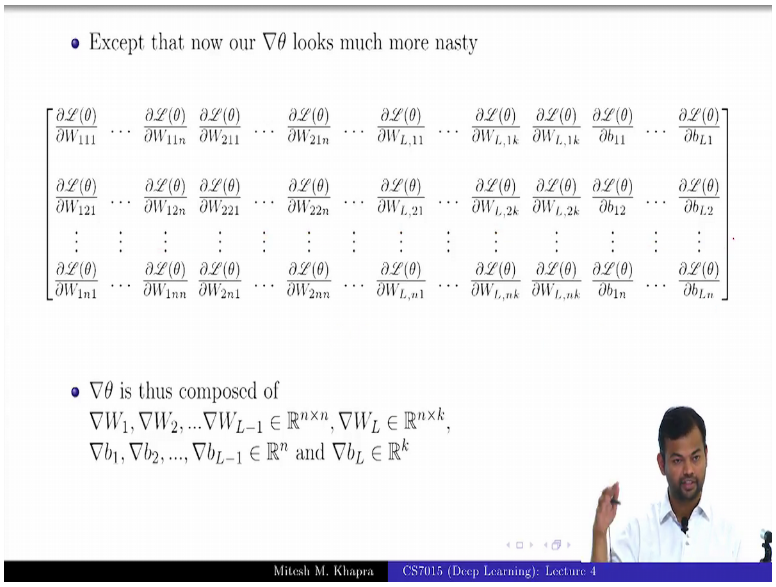
W 2 nn next after w 2 1 1 1; and then after this.

Student: (Refer Time: 03:41).

What is remaining biases?. So, you have b 1 1 b 1 n this slight error here, but intentionally this actual is k, because k is not equal to n, right the last layer has only k parameters, whereas so, that it works is this clear? So, is this are all the scale partial derivative that we need, right. You do not need to worry about taking a partial derivative with respect to our matrix. It just boils down to taking the partial derivative with respect to all elements of the matrix.

So, earlier you just had 2 parameters. Now you have these n cross n plus n cross n to l right. So, l into n cross n plus l into n that many number of parameters is what you have. You get the calculation? Right or rather you have 1 minus 1 layers each of which has n cross n parameters, right and 1 minus 1 layers which also have the biases. So, these are the W's these are the b's, then the output layer one layer which has n cross k parameters and k cross one bias. So, these are all the number of parameters that you have. And this is exactly what this size of this matrix is, right it has all these parameters. And you need to compute the partial derivative with respect to each of these parameters.

(Refer Slide Time: 04:59)



- Except that now our $\nabla\theta$ looks much more nasty

$$\begin{bmatrix} \frac{\partial \mathscr{L}(\theta)}{\partial W_{111}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{11n}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{211}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{21n}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,11}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,1k}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,1k}} & \frac{\partial \mathscr{L}(\theta)}{\partial b_{11}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial b_{L1}} \\ \frac{\partial \mathscr{L}(\theta)}{\partial W_{121}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{12n}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{221}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{22n}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,21}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,2k}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,2k}} & \frac{\partial \mathscr{L}(\theta)}{\partial b_{12}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial b_{L2}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \mathscr{L}(\theta)}{\partial W_{1n1}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{1nn}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{2n1}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{2nn}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,n1}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,nk}} & \frac{\partial \mathscr{L}(\theta)}{\partial W_{L,nk}} & \frac{\partial \mathscr{L}(\theta)}{\partial b_{1n}} & \cdots & \frac{\partial \mathscr{L}(\theta)}{\partial b_{Ln}} \end{bmatrix}$$

- $\nabla\theta$ is thus composed of
$\nabla W_1, \nabla W_2, ...\nabla W_{L-1} \in \mathbb{R}^{n\times n}, \nabla W_L \in \mathbb{R}^{n\times k}$,
$\nabla b_1, \nabla b_2, ..., \nabla b_{L-1} \in \mathbb{R}^n$ and $\nabla b_L \in \mathbb{R}^k$

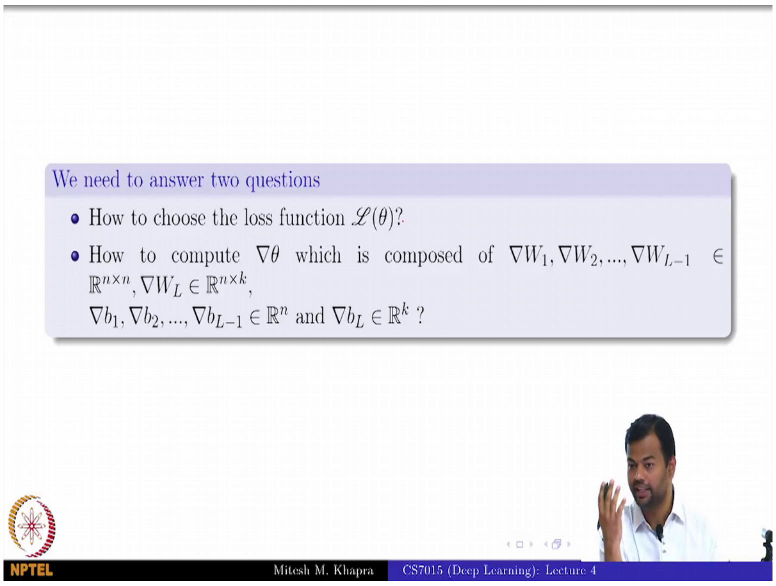
Mitesh M. Khapra CS7015 (Deep Learning): Lecture 4

So, this is what grad theta is composed of, it is composed of the partial derivatives with respect to all the parameters of your network. So now, if someone gives you each of these quantities, same oracle give you each of these quantities, then can you apply gradient descent. You can use the exactly the same algorithm that you are using earlier, just the sizes of earlier vectors changes.

How many of you are convinced that now you can use that gradient descent? There is not a trick question how many of you convinced, how many of you not convinced? Assuming that someone has given you these quantities, right? I know that it is hard to compute we will see how to compute that, but let us assume someone has given you this.

Then you can use gradient descent that is what the case I made in the previous slide that you could initialize with all the parameters compute the gradients with respect to all the parameters and just do this update, fine. So now, we need to answer 2 questions, first is this is the key question.

(Refer Slide Time: 05:55)



We need to answer two questions

- How to choose the loss function $\mathscr{L}(\theta)$?
- How to compute $\nabla\theta$ which is composed of $\nabla W_1, \nabla W_2, ..., \nabla W_{L-1} \in \mathbb{R}^{n\times n}, \nabla W_L \in \mathbb{R}^{n\times k}$, $\nabla b_1, \nabla b_2, ..., \nabla b_{L-1} \in \mathbb{R}^n$ and $\nabla b_L \in \mathbb{R}^k$?

NPTEL Mitesh M. Khapra CS7015 (Deep Learning): Lecture 4

Because we are taking derivative of what, loss functions. So, we need to know what the loss functions that is the crucial question,. And then we are taking derivatives with respect to all these elements. So, whatever I was told you that assume that oracle gives you, now you have to do the hard work and actually find it out. So, if you can answer these two questions then we are done. We have an algorithm for learning the parameters of feed forward neural networks. We all agree that if you have these two elements then we have done, ok.

So, here I will end this module.