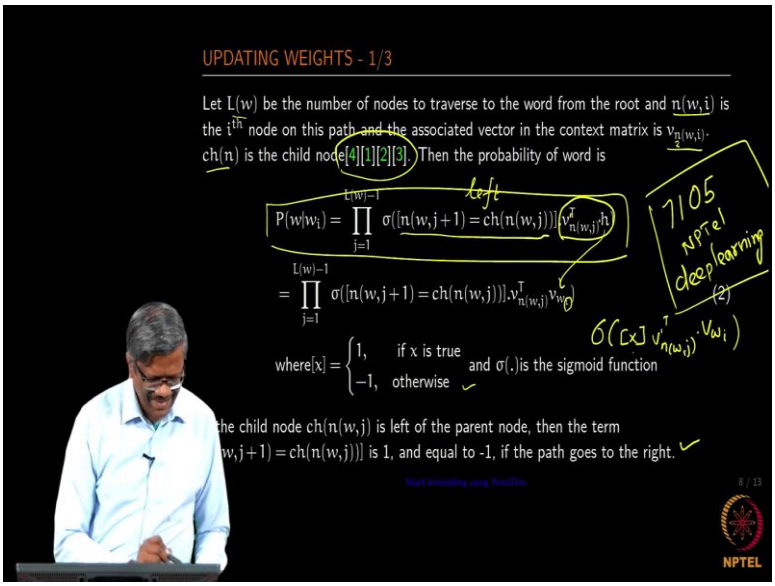
Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Department of Computer Science and Engineering
Chennai Mathematical Institute

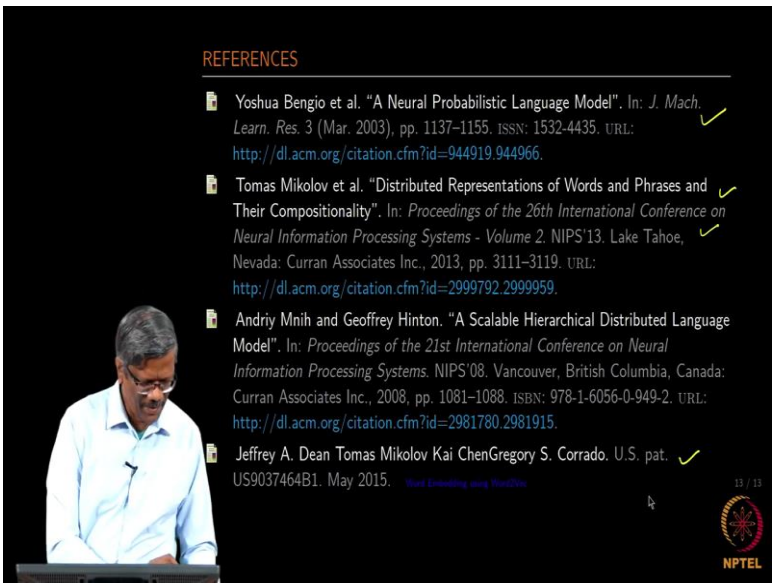
Lecture - 49
Updating the weights using hierarchical softmax

(Refer Slide Time: 00:15)



So, I am introducing some terminologies, you will find these references here in these papers ok; let me show you those first.

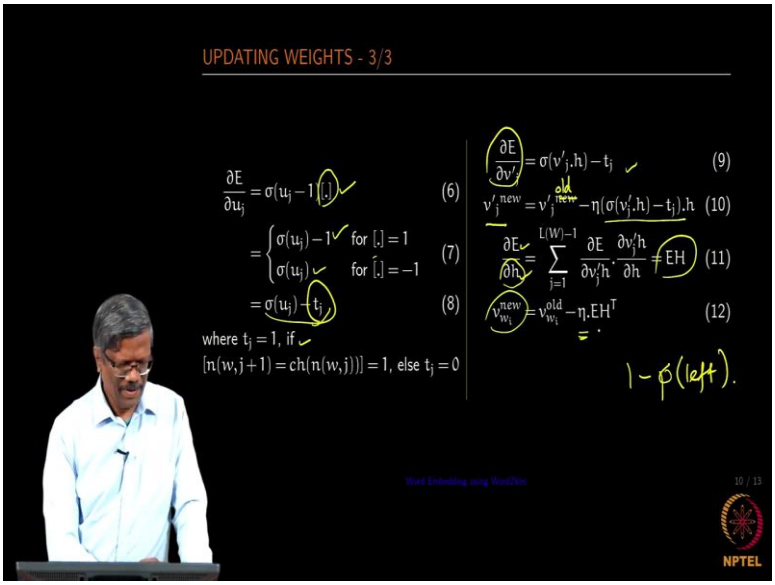
(Refer Slide Time: 00:33)



So, these are the four papers that I have used in the session. One is the very important paper that you should read and it is available publicly, then the second one Distributed Representation of Words and Phrases and Their Compositionality ok. So, this is something that you should definitely read ok, this paper talks about word two vec.

And, then the third one talks about the hierarchical distributed models, how the tress can be how we can place the words in the binary trees in the hierarchical fashion. And, then this is the original pattern filed by Google, this also gives you a piece of information about how these processes are completed in terms of finding the word vectors.

(Refer Slide Time: 01:35)



So, those are the four papers that you should definitely read in order to further understand this and so on ok. I also would like you to listen to one of the lectures I think it is 7105 from NPTEL on Deep Learning, especially on the hierarchical Softmax. So, it briefly explains the theory of the hierarchical model and that would also be a good reference for you to understand this material alright. So, this is another one that you can read ok. So now, let us introduce some terminologies and then see how the weights can be updated in this model alright.

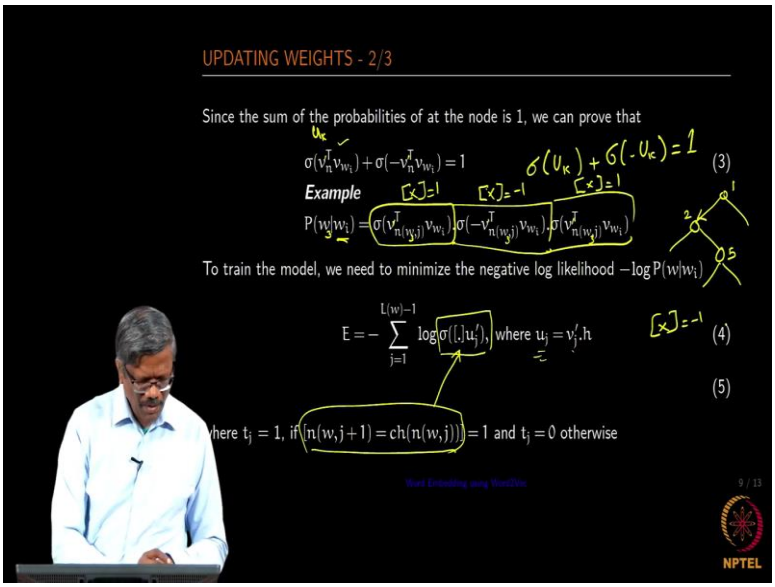
Let L w be the number of nodes to traverse to the word from the root node ok. So, let us say that there are in the case of the 8 nodes we have the length as 3 and we generally represent that as every node as n, w comma i. So, it is the I the node on this path and the associated vector for this node is v n w_1 ok. Suppose if this is node 2 and that

corresponds to this second row of the context matrix and then c h n is the child node ok. So now, the probability can be found based on whether we are going to be traversing to the left or right.

So, what is the decision we are making from one node to the other? So, the decision that we are going to make is done we have to traverse to the left or do we have to traverse to the right ok. And, then this is given by this condition here going to the j+1 node from the child node ok. So, if it is I equal to if this is going to traversing to the left this will be equal to 1 and then if it is if the j the node or j + 1 the node in order to reach from the child node j you have to move to the right, then the value is going to be − 1 ok. So, this is the equation for you to find the probability, it is a generalized equation ok. So, in this case if you look at this I have taken this is $v = v'_n . h$ ok. And, then this his nothing, but the row in the embedding matrix ok.

So, in this case as I mentioned if x is true if this condition is true so we just going to be representing this as x so, that is sigma x ok. So, if this x is true means we are from the node parent node to the child node we are traversing to the left-right if x= −1 we are traversing to the right side ok. So, in that case I think I spoke about this as well here alright.

(Refer Slide Time: 06:03



So, now how do we update this? We now know how to compute the probability you know from the root node to the word that we have in question. We also know that when you take the sigmoidal of the value which we can represent as u k you know for simplicity sake ok, σu_k sigmoidal of ok; so, this is easy to prove ok. If you want to find out the probability of the word given the input word as given in the example, it is going to be first you remember we are going to be traversing from the left I think this is how it is.

So, this is our root node 1 2 I think this is 5. So, we move here that is to the left so, there is no change in the sign of the equation that we had shown earlier and then from here we are taking a right. So, there is a that x brings in -1 here right here -1 [noise[and the last one equal to 1. So, this is an example that we have used earlier ok. So, for that, for you to compute the word 3 so, we can use this model you got it. So, once we know the probability of each of the word computed using the sigmoid through the values that we have gotten from the output layer.

Now, it is time for us to really find out what is going to be the error and how do I minimize the error so, that the network learns ok. The error is again given by the negative log-likelihood. So now, when you want to find the error we now have to apply a different conditional; when we move from the bottom we have computed the probability of the word. And, we know that is it is not the right value because initially when you initialize all the weights in the embedding layer as well as in the context layer it is definitely not going to be corresponding to the word that we are inputting. So, there is going to be some error there. So, how do we correct the errors?

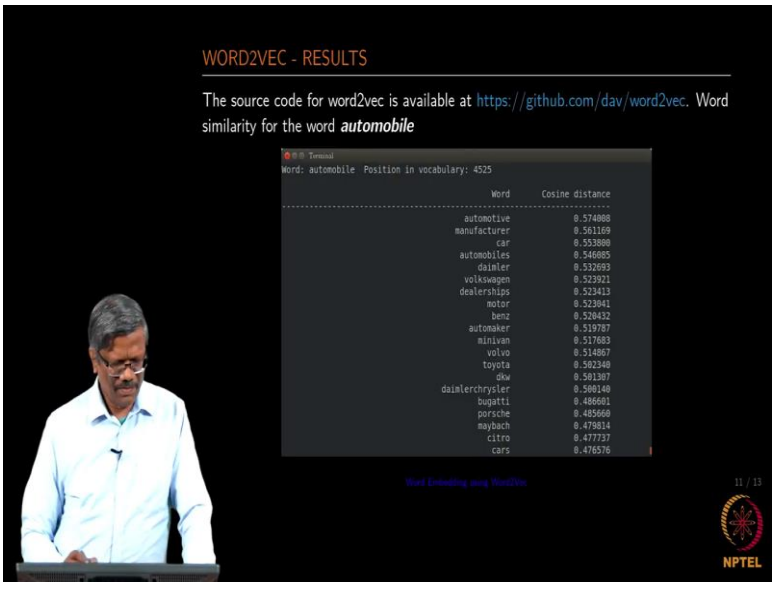
So, when we go back from the bottom to the top right so, we need to make some changes. So, when we move up there so, we need to assume that the values are not right and some changes have to be made. When we computed the left node probability we actually find the right one by doing this operation right. So, when we move up we know that we value has to be changed, and then when you take the partial derivate with respect to u_j, this is what you get. And, this condition comes out of that and when the condition is equal to 1; that means, it is the left node this is the equation that we need to look at. If you have moved right then this is the equation that we have to look at ok.

So, in general we can term it as t_j. So, instead of this, we are now moving it into a standardized notation where, $\frac{\partial E}{\partial u_j} = \sigma(u_j - t_j)$. I am sorry if you are traversing to the left yes and if you are traversing to the right it is going to be 0. So, that means we are making some changes in the values when we move up and the values are updated using this equation. And, the new value v_j new is this is old is computed using this. The same fashion like we have done earlier, the changes of the embedding weights should be done with respect to the partial derivative dou h, that is you are trying to change the weights based on the error with respect to this right which is EH; I think we have derived.

$$\frac{\partial E}{\partial h} = \sum_{j=1}^{L(w)-1} \frac{\partial E}{\partial v_j' h} \cdot \frac{\partial v_j' h}{\partial h} = EH$$

And, the weights in the embedding layers are updated using this formula and you know what is eta is all about ok. So, this is a complex part ok.

(Refer Slide Time: 11:49)



So, I like you to really go through this video a few times, read the reference material that I have given, and try to understand this. So, this is the core of the recent trend ok, this is what is used in the NLP nowadays. So, this derivation, the reduction of complexity, and the neural net model is very important and you need to have a clear understanding of that if you want to really take it to the next level.