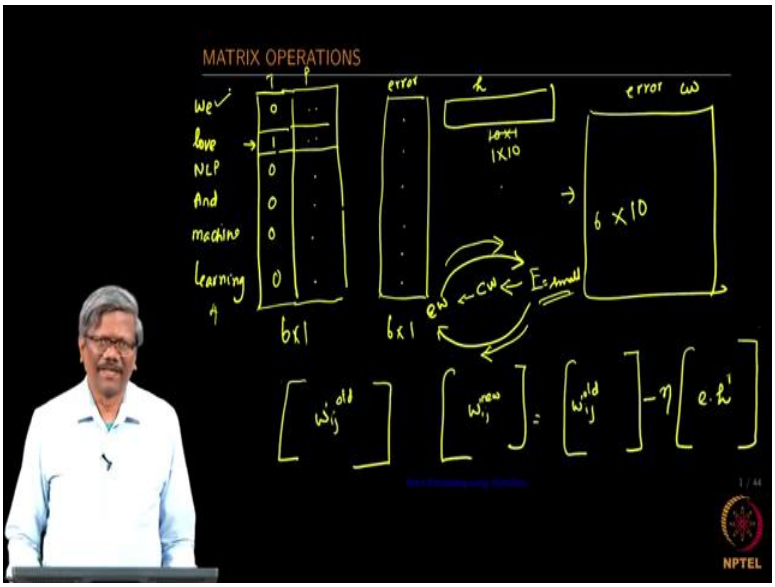
Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Department of Computer Science and Engineering
Chennai Mathematical Institute

Lecture – 44
CBOW and Skip-gram models

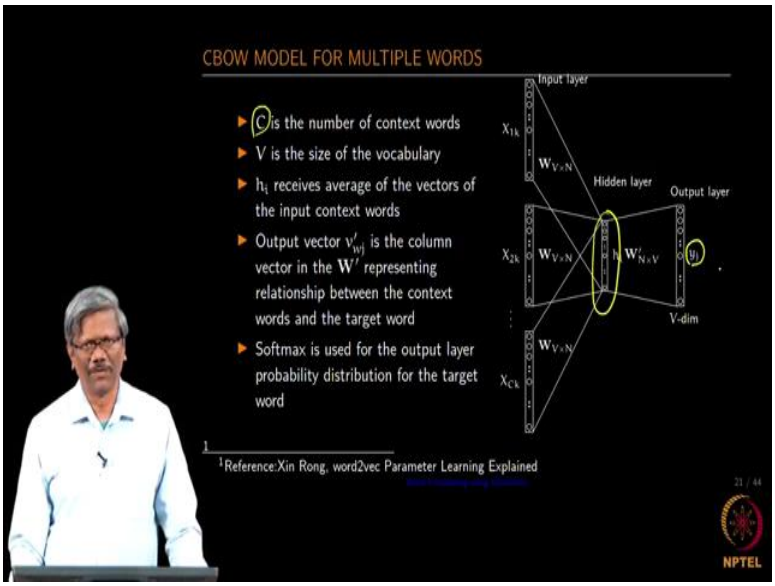
(Refer Slide Time: 00:18)



How do we really use multiple words? I am sure you understand based on what we have done. We have been talking about one-word learning right. There is the only bigram, that you are looking at, when you provide we want the network to learn the law. So, this is a very simple architecture that we have chosen to make you understand the concept of creating the embedding vector.

So, once this operation is completed. So, we can actually throw away the context layer and the hidden layer. The embedding layer will give you actually, the index of the word that we want to really find ok. Supposing, you want to find the word vector for law, it is nothing, but the index of that row in the embedding matrix alright.

(Refer Slide Time: 01:18)



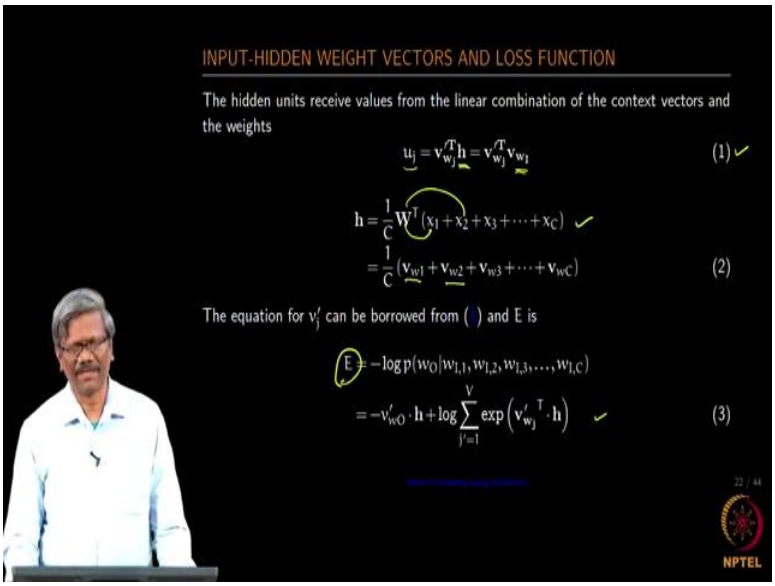
So, now let us take the same thing and extended to the multiple context word right. So, when you have multiple words, how do you really take this forward. So, I am going to be talking about only the CBOW model and I am going to be providing an example in this skip-gram model as well using the python program.

So, now let us look at this multiple words model. In this case we are going to be taking as more than one word as a context word right. So, in this case, C is the number of context words and then V is the size of the vocabulary, which can vary from mean the 10 to 1 million depending on the size of your corpus and then in this case h_1 receives the average of the vectors of the input. So, in this case since there are multiple contacts coming in the h is a linear combination of all of those.

So, we call it an average of the vectors of the input here. And then the computation of the output layer is very similar to what we have done earlier, there is no change there and then there is an error that is a computer. So, we will now, have only one again word that we want to find out right, based on the context we want to find the target right.

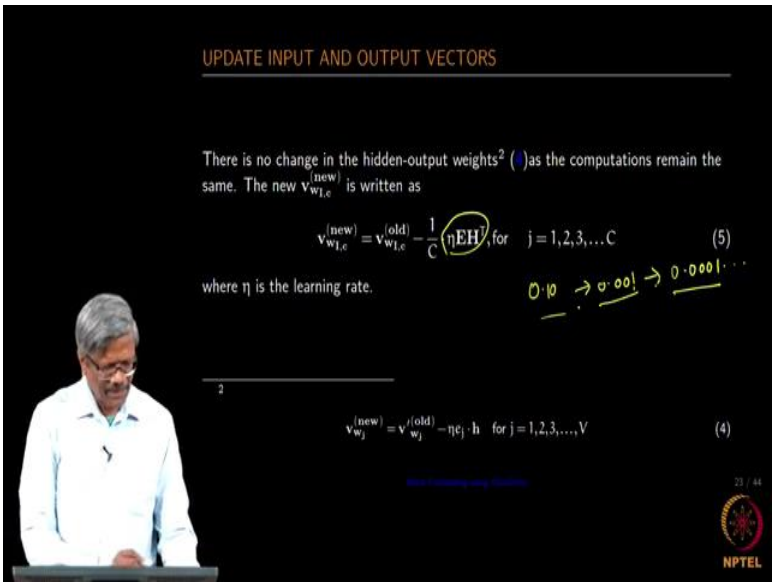
So, we know what the target is, because it is a learning process and then find the error, do whatever we have to do to minimize the error. So, this is how it works in multiple words as well. So, if you use the CBOW model for multiple context work, it is very similar except for the input to the hidden layer computation ok.

(Refer Slide Time: 03:13)



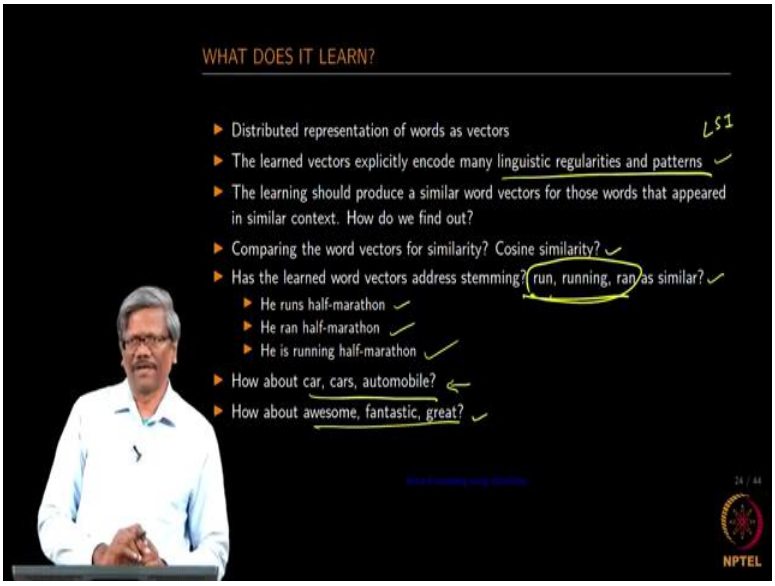
So, as I mentioned earlier so this is how we compute the vectors. The user is the context vector that we are talking about correct. So, it is nothing, but the context vector context matrix that we already have and the dot product of it with the hidden layer values and then the hidden layer value is nothing, but the set of input context. So, his calculated using this formula and then it is given us this ok, if you do this dot product you get the rows of each of the embedding matrix. Again, we can find the error. So, we can borrow the equation borrow from this equation, we can find the error value using the entropy right. So, this is what we want to minimize. This was again the same as what we had seen earlier, alright.

(Refer Slide Time: 04:37)



Again, if you look at the updation of weights, based on the error there is no change correct and then eta our learning parameter which varies from 0.1 to 0.001 or sometimes it could be and so on ok. So, you have to really choose this value to suit your need alright.

(Refer Slide Time: 05:12)



So, what does it learn? So, like in the previous case it learns the distributed representation of words as a vector ok. So, as once this is completed, once you have processed all the input combinations and you have provided the input and train the

network, you will have the distributed representation of every word in the vocabulary right. The learned vectors explicitly encode many linguistic regularities and patterns. So, this is a very important aspect that we have to understand right.

So, we have been talking about this right from the time of LSI correct. A word is known by the company it keeps, you remember that from the word; that means, when the word is surrounded by these similar words several times then we can really identify what that word means what; that means, in so on ok. In the same fashion that this context surrounds similar words will be able to relate those similar words, because of the patterns of the context and the regularity that surround that particular central word that we are talking about. So, the vectors that we are talking about in the, in this model will not just have the details related to it alone, it also has some relationship among the similar words and so on so forth ok.

So, it is not a single word presentation. So, we will talk we will see how it is when we go to the example. So, the learning should produce similar word vectors for those words that appeared in a similar context. How do we find this out? How do we compare those words to find out whether these words occur in the corpus and they are similar, can we use cosine similarity? For example, if you have created one vector and then we have about 1 million word vectors that are created so we can take that word and then try to find out the cosine similarity, to the rest of the words in the vocabulary and then using that cosine similarity, you will find out, how close other words are with the chosen word ok.

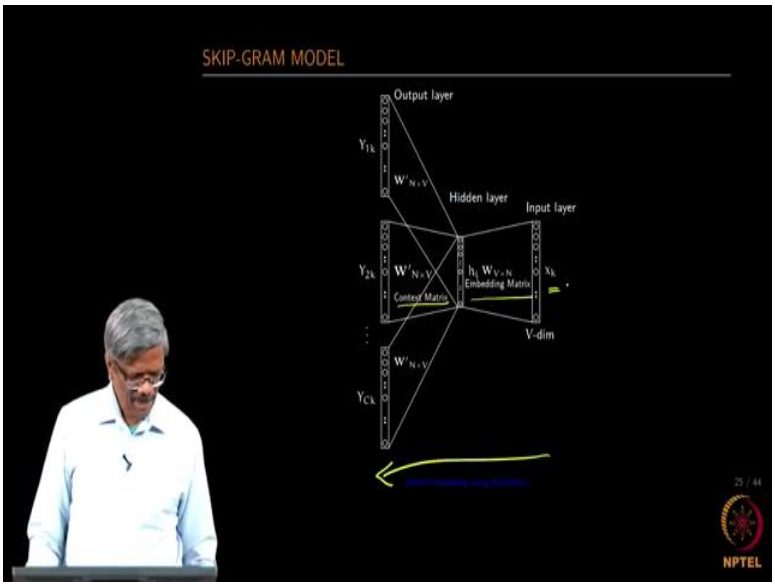
So, this is another one that we can look at the example. Does it address stemming like run, running, ran and so on this really requires the context to be similar in order for us to really combine these similar words right? So, we definitely require a large corpus that keeps using these words in a similar context. So, if we have a different context for each of these words, I am sure no network will really find them out.

So, our assumption is based on effort. These words can be known by the company it keeps; that means, this the context word surrounding these run, running and ran would be very similar only, if they are similar we will be able to take these as similar words. See for example, some examples of he runs half marathon, he ran half marathon, he is running half marathon. So, something like you knows to show how we can really relate these words, you know when we process the corpus.

The how about car, cars, and automobiles. So, the same exact same explanation the whole crew here took. If these words appeared in a similar context, then there is a high probability that these words provide higher similarity values. How about this especially, in the reviews. If you see again similar context it is possible that these words could be co could be found could be similar ok.

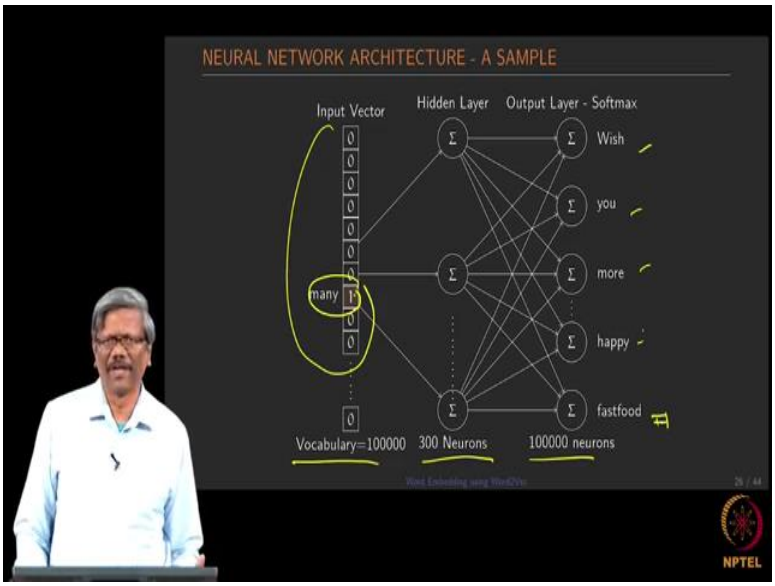
So, again the context is a king here. So, if you do not have the right context and these words appeared all the time in different contexts. There is no way that we can relate them ok. So, that is where these regularities and patterns come into play. So, our basic assumption is these words appear in a similar context and it is true in most cases ok. That is why most of the applications that we are going to be developing and that are developed, show similarities for these words alright.

(Refer Slide Time: 10:29)



This is another I am sorry, this is a skip-gram model ok. So, we have to look at it from this direction ok. The input layer is here and then we have the hidden layer and the output layer. You know well that for the skip-gram, we are going to be providing one input word and the target is going to be our context correct, so, we have the embedding matrix we have the contact metrics and so on. So, what happens here in the skip-gram model? We when we provide the input, we assume that the word that is output would be its context words, if not we go back and keep adjusting the weight right. So, until the word gets its context right alright.

(Refer Slide Time: 11:36)



So, what I just wanted to mention is; you know with the last one before the slide on skip-gram, we have concluded the word victory using the CBOW model ok. So, if CBOW model we have taken the 1 gram, in the CBOW model, we have taken the bigram to identify, the target word and then we have extended it to multiple context words and then we also have shown how and what they learn right, through the examples of the mattresses alright.

So, in the next one as I mentioned earlier, were going to be talking about this skip-gram. In the skip-gram, you are going to be providing the central word and the target words are our context words, right. The computations are very similar, there is no change. So, what I am going to do right now is; I am going to be taking you right into the coding aspect of this ok. So, this is another simple architecture that shows how you can create a skip-gram based model.

Here, we have the input vector and we are inputting many as a word as input and then we have a 300 neuron hidden layer and then we have 100 1000 neurons as our output layer, which is the same as the vocabulary size. So, when you input this, we expect this to provide you the context and these values would be higher than the fast food that you find at the bottom ok. So, this is how we actually capture the context and the input word properly.