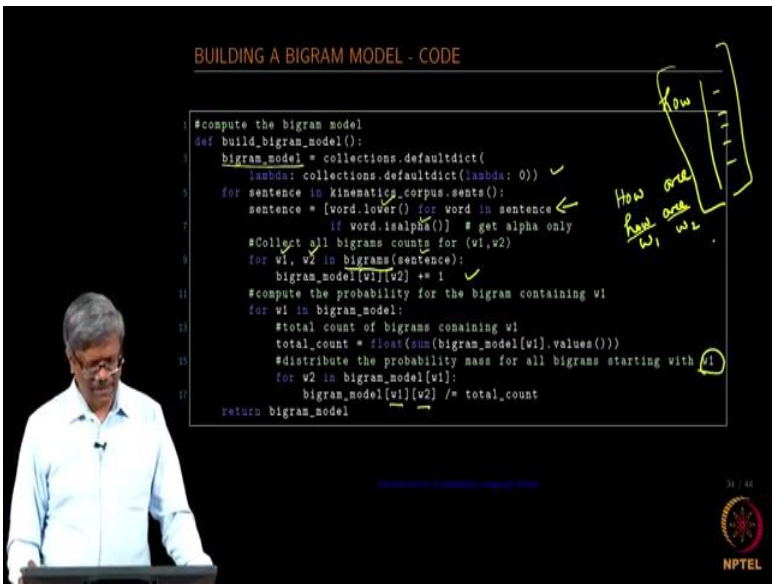
Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

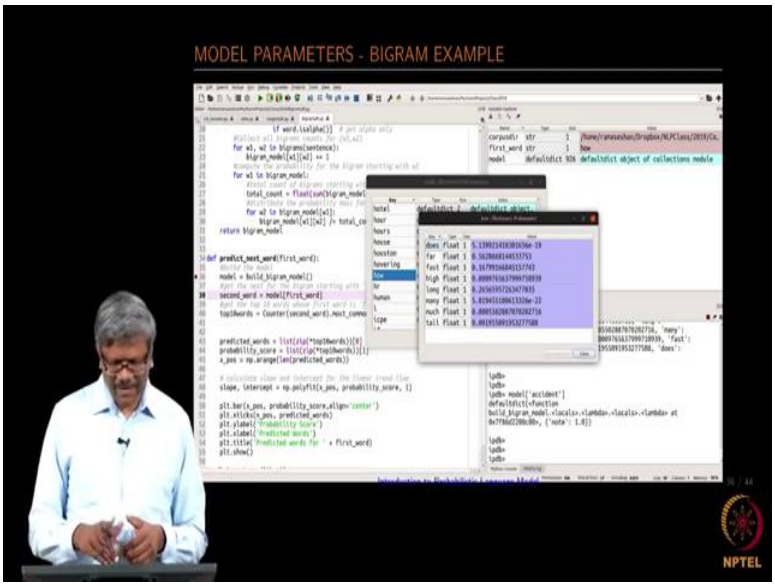
Lecture – 26
Bigram and Trigram Language models peeking inside the model building

(Refer Slide Time: 00:16)



So, let me show you some examples of how this is done and then I will show you what this is really constructing as a model, because we need really peak into what is there inside. So, that it becomes a lot more clearer to us, then what we just read in the theory side ok. So, for that I am going to be showing some code first, going to explaining that and then I will show you a debugging session of that.

(Refer Slide Time: 00:42)



And then I will show you how a particular word when it takes in the bigram and what are the combinations of the second word that is coming after the certain word and what are the probabilities associated with that and then how the model looks like and so on so forth ok.

So, now, let us look at Building a Bigram Model Code. As I mentioned in the previous session the code is available in the GitHub. So, the only thing that I have done is, I have removed the corpus location; it is given I am going to give you an exercise here, instead of you running this code you have to really take a corpus and then combine this code and then see for yourself. The only one thing I just want to mention to you here is, do not take a very big corpus ok; if you running it on your small laptop or a computer big corpus would take a lot of time, sometimes more than an hour to build the model.

So, take a corpus which is about 500 to 600 sentences, and then run the application. So, in the actual code, I have actually commented on the portion of the corpus and you have to really fill that up and then run the code; so, by doing that you will also get the experience of really running this ok. So, now, I am going to build the bigram model ok. So, in the bigram model what I am doing is; first I am creating the dictionary, the dictionary is for every combination of the word what is the probability or the frequency of each of those words. So, I need that kind of dictionary; so I am creating the model as a dictionary ok.

And then I have taken one physics solution example or the problem example and then I have 272 plus problems there and each problem is about 3 lines are 4 lines. And then each problem is considered as a document, so the entire corpus contains about 272 plus documents ok. And then I am considering each one of them as my sentence, each document as a sentence ok.

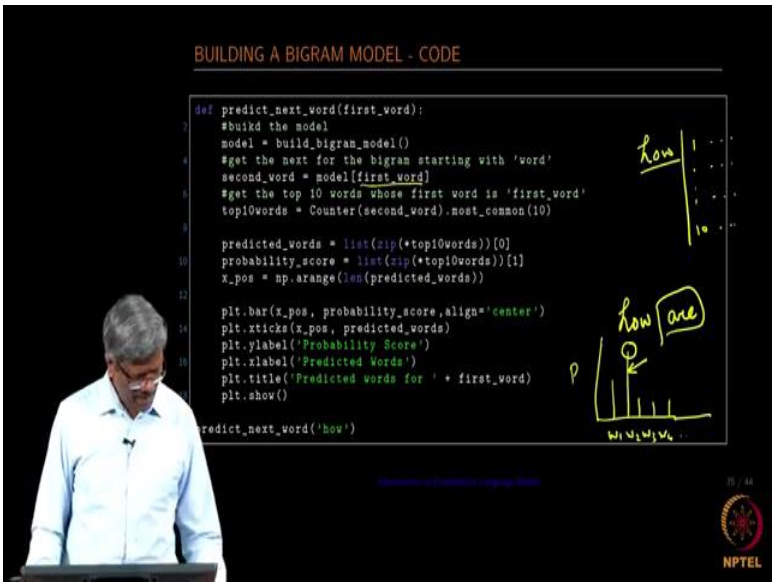
So, I am converting them into small letters, I am tokenizing it, converting into small letters and then I am only picking up the alpha words ok; I am not considering any numerical, alphanumerical and so on. So, I am only considering the words, that you find in the dictionary, and then I am going to be constructing the bigram. So, there is a, if you use NLTK there is a function that converts your sentence into bigrams. So, you will have two words for every sentence that you convert into bigrams and then I am going to be constructing the bigram model using those words.

For example if w 1 is how and is ok; so this is converted as, how are based on this, and then for the bigram model this is word 1 word 2, right. So, when it goes through the entire corpus, it keeps looking for whether this particular combination of words occurred or not in the earlier sentences or documents; if it occurred it increases the count. So, in this way the bigram models are constructed as a frequency initially ok.

And then we need to compute the probability of that, correct. So, we need to distribute the probability mass for all the bigram starting with w 1 ok. For all the word starting with w 1, I am going to be capturing all the bigram and then for every combination of the word starting with how and w 2, I am going to be finding out how many times w 1 has happened and then compute the probability.

So, for example, if you look at how, there could be several words that can occur after how, right. So, for all those words that start with how; so we going to be distributing the probability model, for example, this particular combination if I have about 5 words that formed after how. So, I will have the probability mass distributed for all those 5 words ok. So, when you sum all of those, you will have the value to be equal to 1. So, this is how I construct the bigram model.

(Refer Slide Time: 06:08)



And then the first part of the language model exercise is to predict the next word. So now, given the first word I need to predict the next word. So, how do I do that? So, this is how we are doing it; first is we build the bigram model like what I have shown in the previous slide, and then I am going to be getting the second word based on the first word. So, this model contains the dictionary of all the word starting with the first word, right; and then it contains the second word with respect to the bigram and it also provides the probability of each one of those 2 words occurring together.

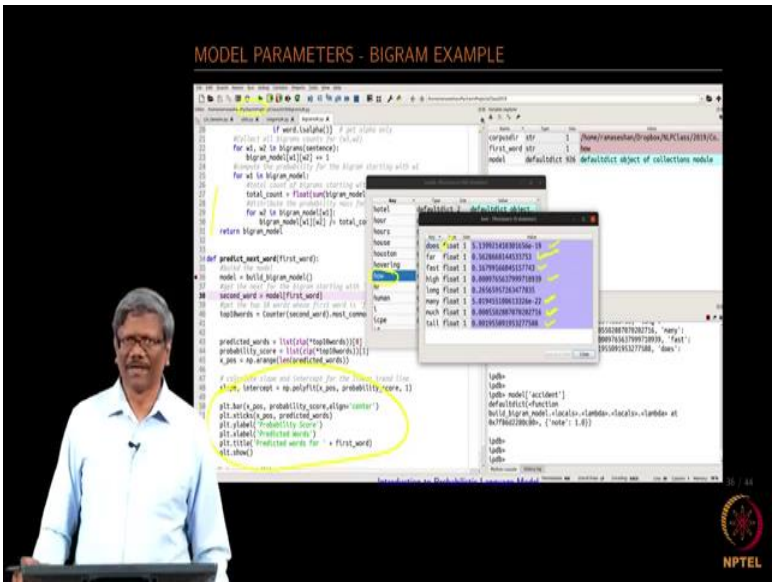
So, when I am pickup the first word and then take a model and then say give me all the second word with respect to the first one, it will give me a certain number of words. So, the combinations of the first and the second, I know you could have had 2 words together only in the entire corpus or 5 words or sometimes 50 words or 100 words and so on so forth ok.

So, there could be a combination of so many of them. So, I pick so, this model what does is, it gives me the list of all those words and the starting with the first word. So, instead of picking everything, I want only to take the top 10 OKs. So, I will only pick up the top 10 words for how ok. So, all these probabilities of the words would also be known ok. So, each one is a word, here ok. So, when you pick all of those and then find out which one has the highest probability in this, that is where we use the maximum likelihood right. So, when you pick the highest one, you will say that how is followed by are;

assuming that are has the highest probability in the list of words that we have found for how as the first word ok.

And let us see how it happens and then we can plot it. And I am sure you remember those plots right, like this and then we pick this particular word, these are the word right. And then the one which has the highest probability will be picked as the next word or predicted as the next word for how ok.

(Refer Slide Time: 08:53)

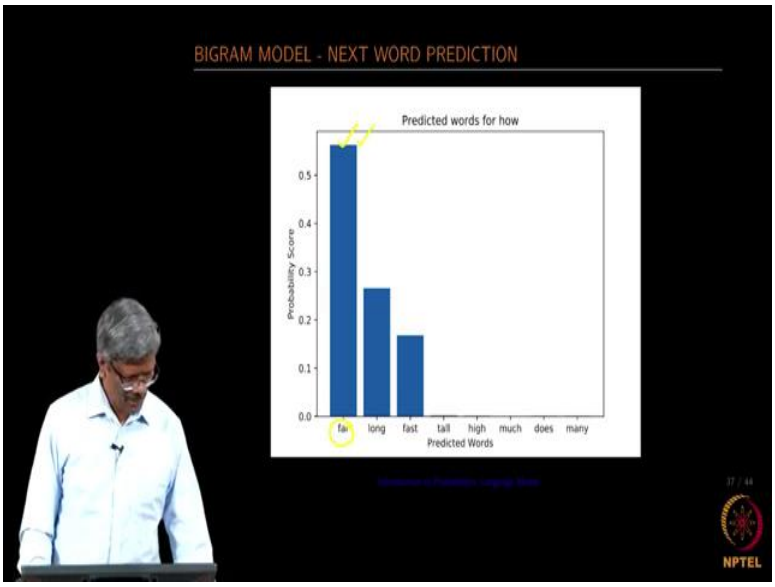


So, let me show you, how it really works. So, in this case I am using a which one I am using; I am using spider here, I am sorry I am using Pycharm as my IDE ok. So, if we use an IDE similar to Pycharm you will be able to debug the code and then run and then go step by step and see what is happening inside the bigram. Here I have done is, I have since we have already built the model ok, I am going to be predicting this second word ok; as I mentioned earlier, I going to be giving the first word as to how ok.

For the word how you see there are various combinations ok; how does; how far, how fast, how high, how long, how many, how much, how tall ok. So, and then it gives you all the probabilities of those 2 words occurring together. So, out of which, you pick the highest one ok. So, let us see what is the highest one; by looking at these numbers are pretty small for many of this, see this, ok. So, when you look at how does, probably it happened very few times in the entire corpus.

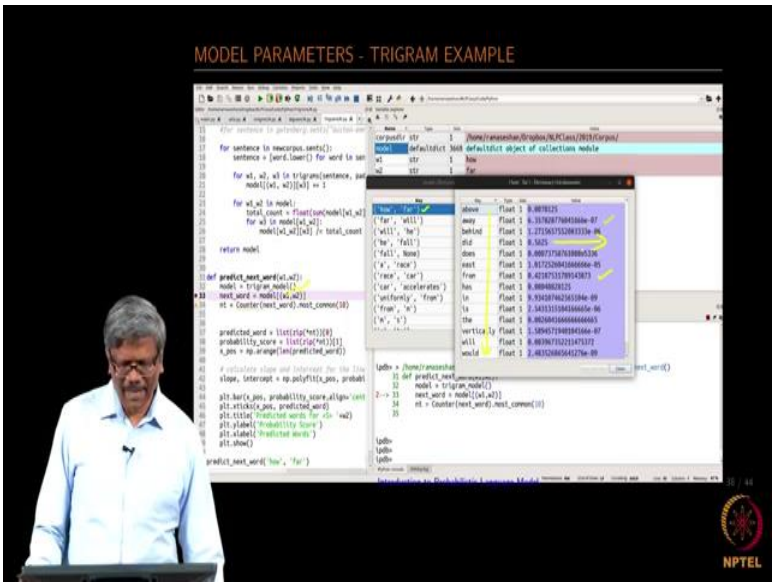
In the same way, how many occurred a lot less time than how does and then how much occurred lot more than how does and so on. You know there are some other once which are pretty small as well, how much also is pretty small, how tall is fine; but if you look at how far this has the highest probability ok. So, predicting the next word, you now know what is going to be the next word in this case, right. So, when you ask the system to predict the next word for how for the given corpus, it is going to be giving you how far.

(Refer Slide Time: 11:09)



Let us plot this to see it a lot more clearer. So, if you plot this using the plot functions, which I have listed here, actually you will find the same thing in the code that is available in the GitHub. So, you do not have to really write a code for this; you will see this, you know, you can see that the far has the highest probability there. So, we are going to be picking this one as our next word, ok. So, this is based on the bigram ok.

(Refer Slide Time: 11:49)



In the same fashion, we can also do a similar exercise for trigram, I am going to showing the application; the application again is available in the GitHub, you can go take a look at that. Again I have left the corpus space empty, so you have to go and then fill that space, so that you can run the exercise on the corpus. Again I am repeating take a smaller corpus to find out how it works, you know if you want to really try on a bigger one for a production sake, you can go ahead and then do it; but I have to warn you that the longer or a bigger corpus would take a lot of time for building the trigram, 4-gram, 5-gram models and so on ok.

So, for you to understand the concept, it is enough if you pick up a very small corpus that contains about 300 to 500 sentences. So, again in this case I am going to be showing the debugging exercise that I have done, to demonstrate what the model contains, and so on.

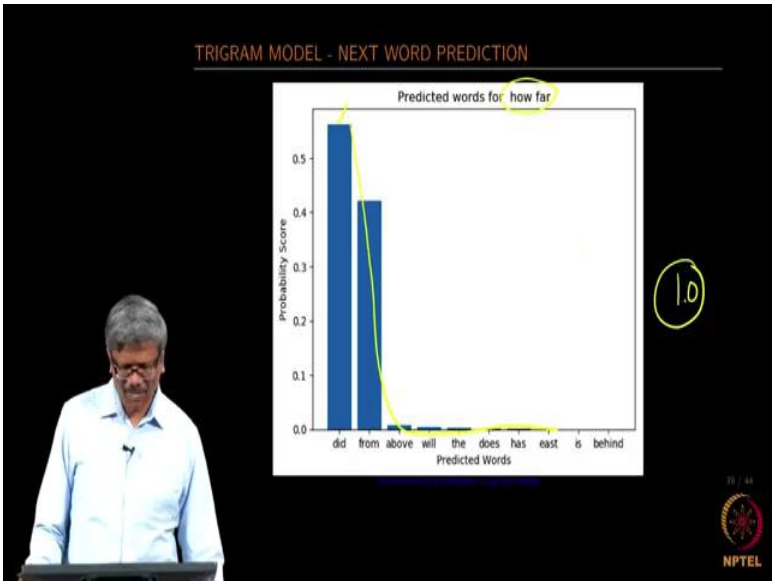
So, in this case of the trigram, we have 2 words that are given, we going to be predicting the third word correctly; that is how the trigram model works in terms of prediction ok. So, earlier we saw in the bigram model how and far occurred. So, we have picked up the first 2 words, what is going to be the next word ok. I can use the same bigram model to go ahead and then do it; but in this case I have picked up the trigram model to demonstrate how this can also be done using this.

Again in this case, you see that the keys are how far. The construction of the trigram is exactly similar to what we have done earlier. The model construction is very similar,

except that you will have 3 words and the count of 3 words and the probability of those 3 words would be computed instead of a bigram, that we had seen earlier ok. Again the prediction, we will be providing w 1 and w 2 and then I will let the model predict the third word and so on.

So, if we provide the keys, right let us say how and far. So, this is going to give us again the list of words, that follows how and far ok. So, you would see that how far above, how far away, how far behind, how far did, how far does and so on ok, there are so many that you will find. Again you will see the probabilities of each of these words that followed those 2 words; you will see very small ones and bigger ones again, in this case ok. Let us see what is our maximum likelihood word?

(Refer Slide Time: 15:01)



So, in this case for how far, the word does; so that means, this particular combination occurred several times in this corpus, that is why the word did has a very high probability. So, and you could look at this you know, this goes like this and if you sum all of this, it will you will have 1.0 at the end ok. So, I am only picked up the last rather the top 10 words that we have here. So, here this has a very high probability; that means, more than half the number of times this has occurred in the given corpus ok. So, I guess you have understood, what is the model in this and what the model contains, how we are really predicting the next word based on the bigram and trigram models.