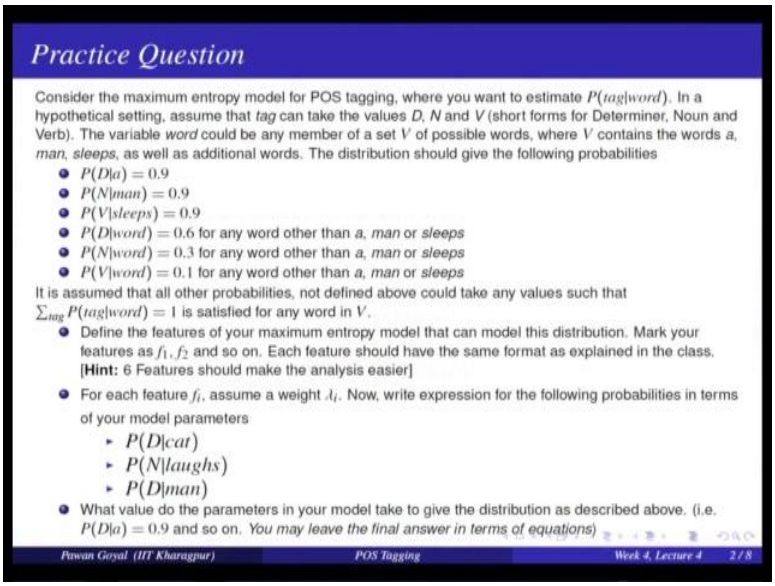
**Natural Language Processing**
**Prof. Pawan Goyal**
**Department of Computer Science and Engineering**
**Indian Institute of Technology, Kharagpur**

**Lecture – 19**
**Maximum Entropy Models – II**

So, welcome back for the fourth module of this week. So, in the last module, we had started talking about the maximum entropy classifiers what is the basic principles for using this maximum entropy classifiers. And we saw that we can use any hidden new set of features that were not allowed in the earlier model like HMM. So, today I will start by taking a simple example that will give you some idea on how maximum entropy classifier actually works for a simple classification problem. And then we will see how it can be used for a sequence tagging problem that is I am given a sentence. So, this is my problem for part of speech tagging and given the sentence for each of the words, I have to find out what is the actual part of speech tag for that.

So, right now what we had seen in maximum entropy classifier, it gives me a probability on the class y given a context. So, for each individual word for the context around that word, I can predict what is the appropriate part of speech category, but once I am given the sequence how this model will be actually utilized. So, there it has a special name called MEMM - maximum entropy Markov model and we will see what is the algorithm for using the classifier that we have discussed for a sequence tagging problem.

(Refer Slide Time: 01:40)




So, now starting with a simple problem on how to use maximum entropy classifier, the practice problem, so you can try it on your own but I will try to give you certain hints. So, problem says, so you want to use maximum entropy model for part of speech tagging, and you are trying to estimate probability tag given word. And in this hypothetical setting, you can assume that there are only three possible tags determiner, noun and verb. And the word can belong to any member of a set V that is your vocabulary; and vocabulary contains the words a, man, sleeps and some additional words. So, in this problem it does not matter how many words are there in your vocabulary.

Now, the constraint given is as follows. So, the distribution that you will find probability tag on word should give the following probabilities. Probability of determiner given the word a is 0.9, probability of noun given the word man is 0.9, tag V given sleeps is 0.9, D given any word other than a, man or sleeps in the vocabulary is 0.6 and same for that two constraints that are given here. So, remember that is what we did in the case of maximum entropy classifier. You are given the data; and from the empirical distribution, you extract certain facts and now you want to find out a distribution that resembles that empirical distribution.

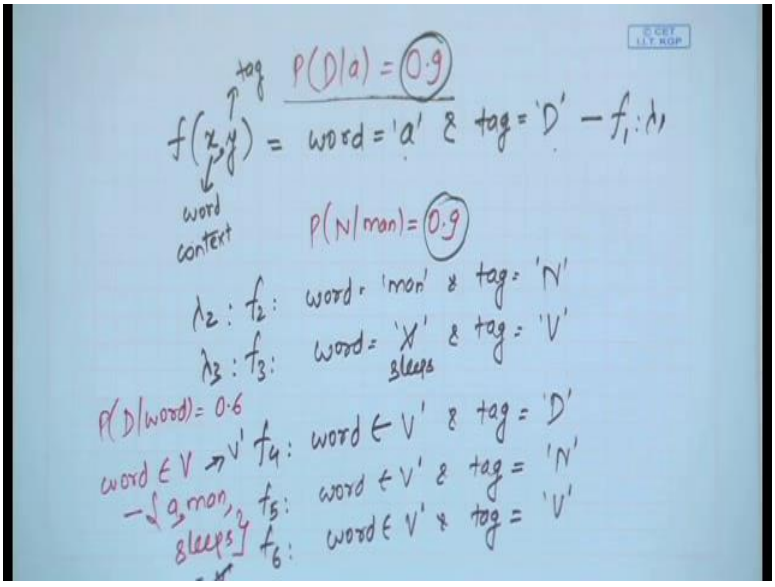
So, now I want to find out probability I given word such that these constraints are followed. So, now how do I go about building my maximum entropy classifier this is the problem. So, now it is said that all other probability that are not given suppose that take

some values such that probability I given word is one for all possible tags, this is the normalization constraint, the probability for all the possible tags for a given words should add up to 1.

Now, there are certain questions here. So, the first question says is that so you are given this problem settings. Now, try to define what will be the features option maximum entropy model given these constraints. And the problem says that Markov features as some f 1 and f 2. And each feature should have the same format as we explained in the last class, so that is each feature your function of x and y x is the context around the word and why is the particular class or tag.

Then the next question is for each feature f i assume a weight of lambda I now write down the expression of the following probabilities. So, what is the probability of D given cat in terms of your model parameters? So, the parameters are mainly lambda 1 to lambda 6. Similarly, find out probability of N given laughs probability of D given man. Finally, the problem is what should be the values of the parameters in your model lambda 1 to lambda 6 such that you have to the distribution as shown in the question. So, this is by problem. So, let me try to solve it n-steps. So, this is the first problem what should be the features of my model, such that they resemble this distribution. So, these are the actually certain facts from the data.

(Refer Slide Time: 05:01)



So, let me write down the first fact probability D given a is 0.9. So, now I want to find out some features f x, y such that it can try to match this particular constraint. So, x is on my data. So, this was my word and the context and y is the tag. So, now what am I observing in this particular example I am observing that when the word is a, then the tag that I gives is D with the probability of 0.9. So, what kind of feature will try to model this? So, my feature should be affects y. So, I can use a feature word that is my x is equal to a, and tag that is my y is equal to D that encodes both a and D that are that I am observing in this empirical data. In this, we can call as your f 1 what is a and tag is D. So, this is a binary feature. So, whenever you will see word as a and tagged D, you will put it as 1. So, to express it you can say f 1 is equal to 1; if this happens is equal to 0; if this does not happen I am not writing it in this paper.

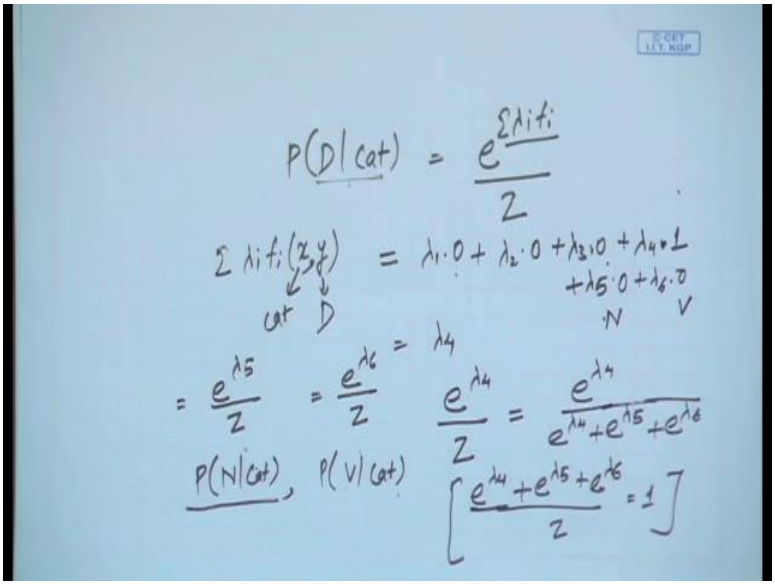
So, now I have to define some other features right and the hint is that six feature will make my task easy, so that is like sufficient hint you have six possible constraints and you have to define six feature one for each for each constraint. So, let us take the second constraint. Second constraint is probability N given man is 0.9. So, for this constraint, what will my feature, again so f 2 would be word is equal to man and tag is equal to N – noun. Same way you can do for your third case. So, f 3 will come out to be word is equal to sleeps and tag is equal to V.

Now, what would you do for your fourth observation? What is that probability D given word is equal to 0.6, for word in vocabulary minus a, man and sleeps, for any word other than a minus sleeps. So, what would your feature, again you will have a similar feature f 4 word is in, so let me call this as V prime V minus a man sleep, let me call it as V prime this is my V prime. So, word is in V prime and tag is equal to D. Similarly, my f 5 will become word in V prime and tag is N. F 6word in V prime and tag is V, and these become my six features that will try to model the distribution. So, this is how you will select the features given certain facts about the data.

So, see here I am not doing anything from matching this, these numbers right now, so that I will try to do using my lambdas that the feature weights set that I have to choose. So, the next part of the question says is that now give weights. So, suppose this is lambda 1, it has a weight of lambda 1, lambda 2, lambda 3, and so on. So, each feature f i has a weight of lambda i. Now, try to find out some probabilities in terms of the model

parameters. So, let me take out the first question itself that is probability of D given cat, I want to write that in terms of model parameters.

(Refer Slide Time: 10:04)




So, what will be this probability D given cat? So, now in terms of my maximum entropy model, how do I write this particular probability, it should be e to the power sigma lambda i f i divide by a normalization constant; and Z should be such that this will add up to one for all the three tags. So, probability D given cat plus probability N given cat plus probability V given cat should be 1. So, let me take this particular y, let me focus only on this and I will talk about Z later. So, now so this is sigma lambda i f i x y, x here is cat and y is D. So, now f x, y, so each of the six features are binary features, so they will take either 1 or 0; and lambda ones are just given to me and then the values are not given. So, I can just take lambda 1, lambda 2 and so on.

So, now, what will be this value? So, let me take the first feature. So, first feature is here word is a, and tag is D. In this case, my word is cat, so clearly this feature is not one. So, for f 1, x 5 it will be 0. So, lambda 1 times 0 plus lambda 2 times; what is my second feature second feature says that word is man and tag is n again this is 0; f 3 word is sleeps tag is V again 0. F 4 so my feature f 4 is word is in v prime, is this word in v prime cat is not either a man and sleep so it is in v prime, and tag is d. So, this is actually one. So, word is in v prime and tag is d, so lambda both 4 times 1 is I will not write f 4, f 4 is 1 in this case plus lambda 5. Now, you will see in feature 5 and feature 6, the tags

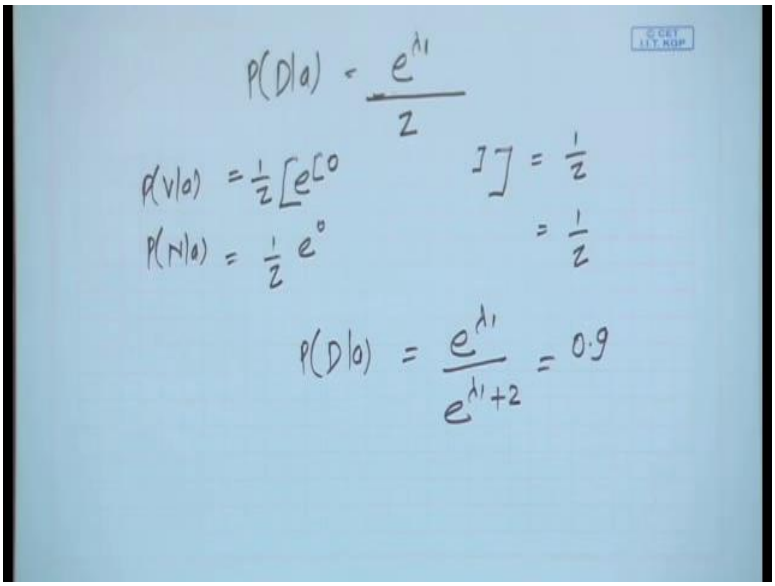
are different they are N and V. So, they will again be 0 plus lambda 6 0, this comes out to be lambda 4. So, now what is my probability? It will be e to the power lambda 4 divided by Z.

Now how do you find out Z? Now have to find out Z, you have to actually compute the other two probabilities also. So, you have to compute probability N given cat, probability V given cat and then you can make this plus this plus this is equal to 1, so let me quickly do that. So, what will happen for when you compute probability N given cat? The first three features again use a word a, man and sleeps that is not here. From fourth features, they are using the words in V prime, so the word cat is in V prime. So, the features can be 1 from f 4, f 5, f 6, but now I have to see the tag. For f 4, the tag was D; for f 5, the tag was N; and for f 6, the tag was V; this was N and this was V. So, for N given cat f 5 will be 1; and V given get f 6 will be 1. So, can I write probability and given cat, this will be e to the power lambda 5 given Z, and this will be e to the power lambda 6 divided by Z.

Now, you can compute what is P D given cat because this c will add up to 1. So, put in the constraint e to the power lambda 4 plus e to the power lambda 5 plus e to the power lambda 6 divided by Z is equal to 1, so that gives you Z is equal to e to the power lambda 4 plus e to the power lambda 5 plus e to the power lambda 6. So, this we can write as e to the power lambda 4 divided by lambda 4 plus e to the lambda 5 plus e to the power lambda 6 that is your probability of D given cat in terms of your model parameters.

Now, in a very similar manner, I will suggest that you try out the other two cases, what is the probability of N given laughs, and what is the probability of D given laugh. So, again we have not use of the probabilities 0.9 and etcetera that were given in the constraint of discussion, so that I have to use in the last point. So, what values with the parameters in your model take to give the distribution as described above, so that is what values they should takes us that probability D given a is 0.9. So, need to you may leave your final answer in terms of equations. So, let me try to take only the first try to satisfy only the first constraint what should be the constraint on my lambda such that the first constraint of probability D given a is 0.9 satisfied and the idea is very, very easy.

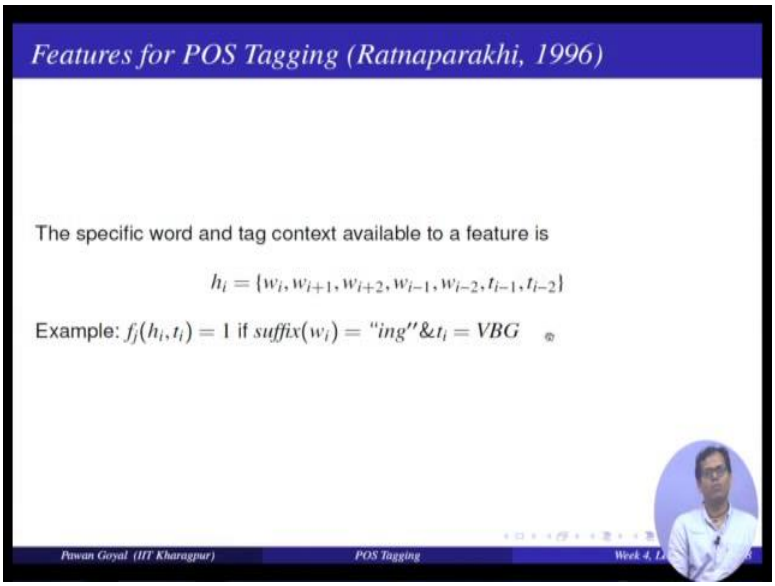
(Refer Slide Time: 15:53)




You will just that like we found out probability D given cat, you will now try to find out probability D given a, and that will be e to the power lambda 1, yes because first feature is 1 for this case divided by Z. Now, what is Z? So, for finding out Z, I have to find out probability V given a, and probability N given a, now brought this probability V given a. So, again I will write 1 divide by Z into something e to the power lambda 1 times f 1 and so on. Now the word a is given only in the first feature, yes, first feature says the word is a and tag is D. So, the word is a is given; it is not given in feature 2, feature 3.

Now, what about the other features f 4, f 5, f 6, they all talk about words in V prime where word is not there. So, any of the feature cannot be one for this case, so all of them will be 0. So, it will be simply e to power 0, all the features are 0. Same happens with N given a, so they are 1 by Z, 1 by Z, and this tells me that probability D given a should be e to the power lambda 1 divided by e to the power lambda 1 plus 2, and there should be 0.9 to satisfy the constraint. And that will give me the value of lambda 1. Similarly, you will do for all the six cases and write down the questions. So, hope that gives you a good idea on how do we actually start building a maximum entropy classifier from (Refer Time: 17:50) in the data and how do we compute the probabilities.

(Refer Slide Time: 17:54)

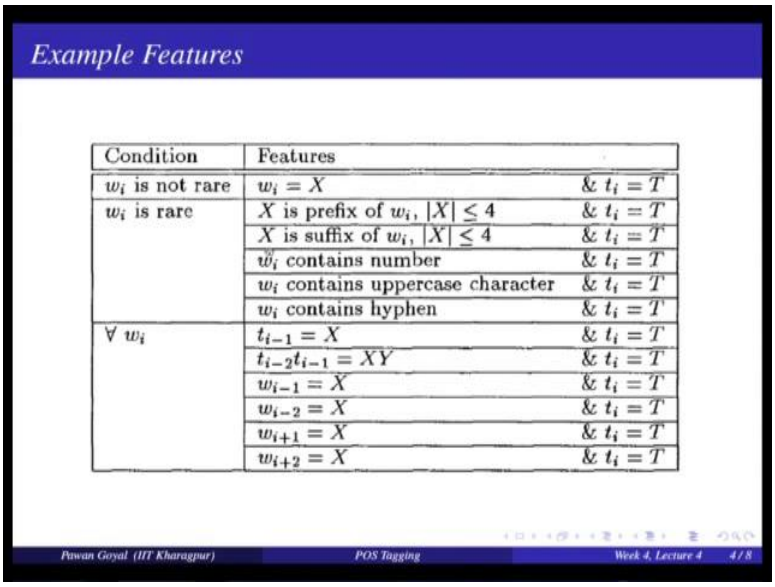



Now, I will take a very specific case. So, that is by the paper written in 1996 by Ratnaparakhi on using maximum entropy model for part of speech tagging. So, what I will show is that what kind of features they are used for part of speech tagging. So, now, this model is there now what is important to understand is that given a problem, what sort of features I should use to get a good performance, model will work well if the features that you have chosen you have chosen make sense for this particular problem. So, for part of speech tagging problem, what are the kind of features that he had used.

Now, given a new problem again you have to go back and find out what should be the interesting features you know now the format of features how you should define your features, but what are important features we will always depend on the particular problem that you are solving. So, as we had said in the maximum entropy model you choose a context around the word from where you can choose your features. So, in this case, what is the context they have chosen? They have chosen the current word, the next two words, the previous two words, and the tags given to the previous two words. And that is interesting how do you use this kind of features at the time of decoding when given a sequence you are trying to find out the tags. Because you do not know what is the best tag for the previous word. So, we will look at this problem, when we talk about the search algorithm the beam search algorithm that how do we make use of previous tag and previous tag here are to be assigned that are not yet assigned fully, but here use always features in all this context for defining this features.

Now, what are the form of features? So, yes just to give you another example, features will depend on the history or the context and the current tag. So, it is one if the current word has a suffix of ing and the tag is VBG that is one kind of feature; the suffix of the current word and the tag. So, what are the other features?

(Refer Slide Time: 20:02)

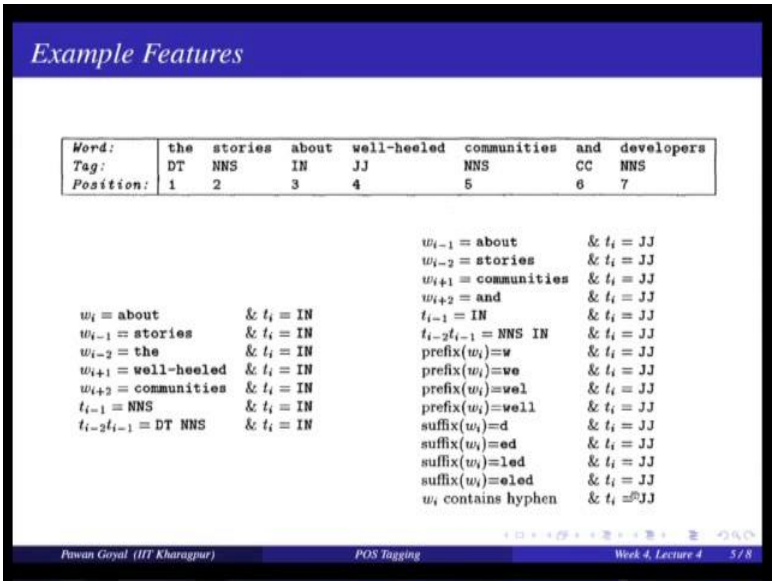



So, what is one interesting thing here? We divided it features into three parts; some features are only for those words that are not rare that a very, very common. Second sort of features are those words that are rare. Why does it define separate features for the words that are rare? Because these words may not be seen again in my training data, but I might see some other real words, so can I use some of the properties of these words instead of using the word directly? So, further words that are not rare, they occur many times uses a word directly w i is equal to x, and its current tag this is the form of feature. But if the word is rare, he does not use the word directly, he says x is a prefix of w i length of x is less than equal to 4. So, he uses all the prefixes of the word is starting from 1, 2, 3 and 4, same for suffix.

Now, how will that be helpful using the suffix? So, for example, I have words that ends in ed, but overall the word is rare. So, can I use the fact that the word ends in ed ok that is why we are using the suffix here similarly for prefix, I can have something like in for making the opposites for an adjectives and so on. So, this can be captured by using this kind of features. Whether the word contains a number or in uppercase character or in

hyphen, this is for rare words. Now, he just some other features for all the words. So, what are the features? The previous tag is x and the current tag is t, this is very, very generic feature. The previous two tags are x and y, current tag is t; previous word is x and current tag is t; previous-to-previous word is x, current tag is t then similarly for the next for the next, next words.

(Refer Slide Time: 22:19)



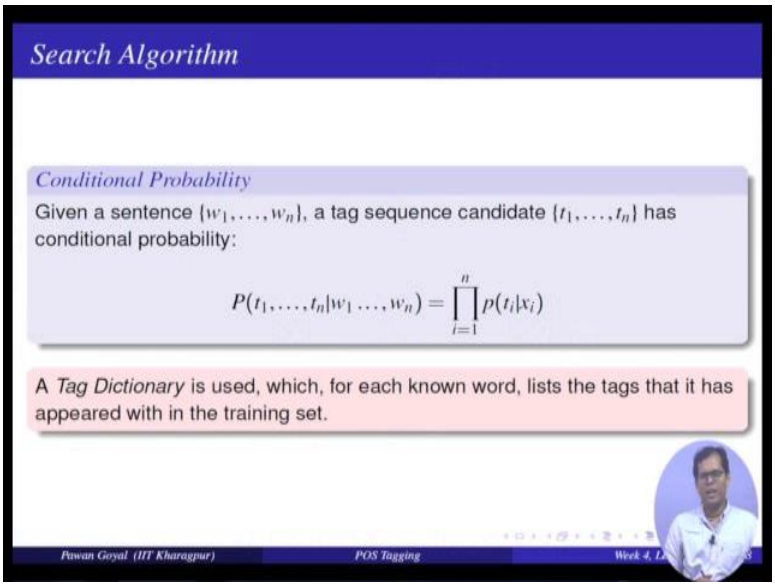

So, now let us see what will be the form these features will take. So, this is the generic form of the features. Now, let us take a simple context and see what are the forms these features are going to take. So, this is one context, I have a sentence the stories about well heeled communities and developers, and the tags are DT, NNS all the part of speech tags are given to me. So, this is for my label data.

Now, what are the values these features will take? So, firstly, I will differentiate between the rare words and not so rare words. So, what can be the rare words here? For example, the word well-heeled here is a rare word; and other words like stories, communities, developers are not rare. So, for the words that are not rare what feature am I using the current word and the tag? So, my feature could be the word is a stories, and tag is NNS and so on. So, for the word well heeled I will use the suffix and prefix. So, W is a prefix of the word and tag is J J; W is a prefix and tag is J J; similarly ed is a suffix and tag is J J and so on. And then there will be some features for all the words like for about and well-heeled, what would be the feature the previous tag is IN and the current tag is JJ.

The previous two tags are NNS and IN, and the current is JJ. The previous words or the word stories and well-heeled about and well-heeled and so on, so you can use all these features. So, here are all these features. So, this is prefix and suffix for the rare word well-heeled.

And there are other things like w i contains hyphen that is in the case of well heeled and tag is JJ. So, now that is how you will write down all your features from your data. And you will learn your parameters lambdas and all; and at run time you will use again these features in the parameter to finding the probability of the tags. So, hope this gives you some idea, and how can you choose your features for a given task.

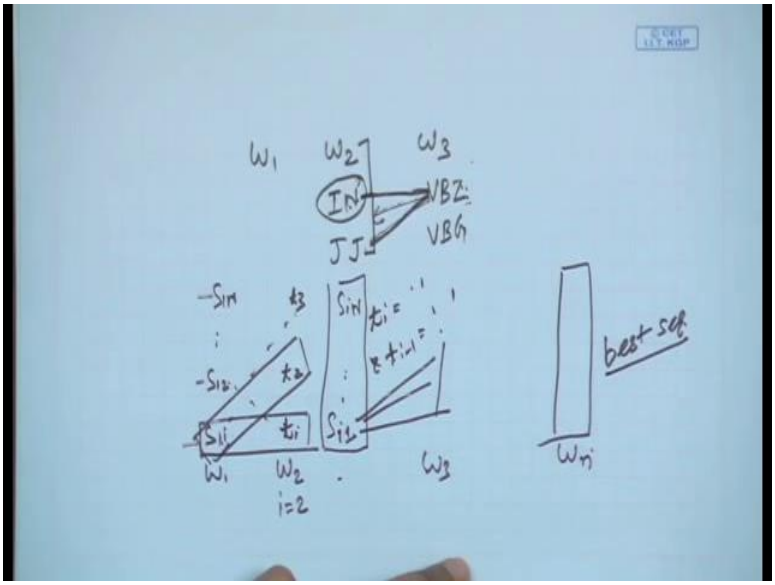
(Refer Slide Time: 24:37)




Now, let me go to the search algorithm for this maximum model. So, this we had discussed earlier also, but we did not discuss in full details. Some given a sentence w 1 to w n these are the words, and I want to find out the probability of tag sequence t 1 to t n. And we said that we can write it in this form multiplication of probability t i given x i, x i is the context for each individual word. Now, the problem here is and you can use some tag dictionary that says for each individual word what are the possible tags similar to what we did in the case of Viterbi decoding for HMM. Now, what is one particular problem in this case? So, as such it looks as if you can do it independently for each word right; for the first word I can find out what is the best tag multiplied with best tag for the

second word and so on. Now, what is one problem with this approach let me just give you one simple example.

(Refer Slide Time: 25:36)

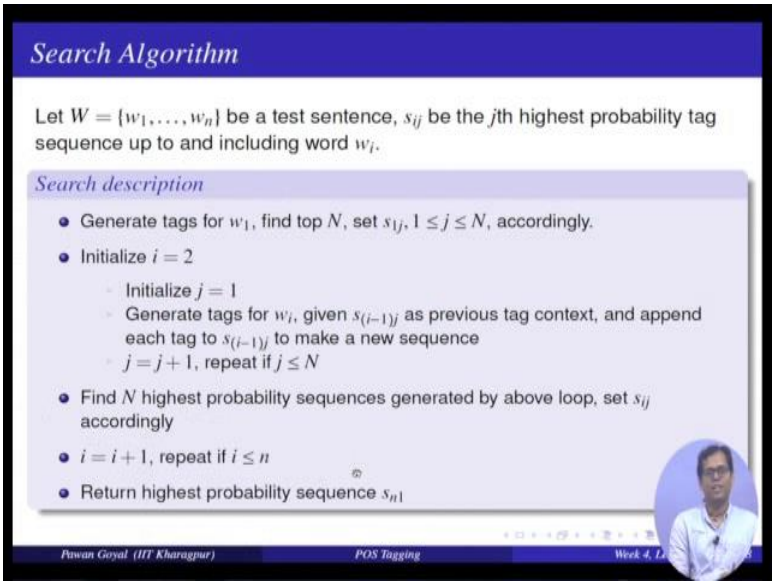



So, this is a hypothetical example. So, let us say that we have word 1, word 2, word 3. And word 2 can take two tags IN and JJ; and word 3 can take lets to keep it simple that it can take like VBZ and maybe something else like VBG maybe they may not be possible, but just a hypothetical case. Now, suppose by using this algorithm or doing it independently, we found out that for w 2, IN give some probability and JJ gives some probability. Now, in w 3 what might what could happen some of the features might depend on the previous assigned tag. So, for VBZ you might want to use what is the tag assigned in the previous word so that means, choosing the probabilities the best probability at this point may not be the best.

So, choosing the tag at this point may not be the best because at the next step you are using the tags and the probabilities can change this can happen that IN is giving the better probability at this point, but when combined with VBZ, JJ gets a better probability this can happen. So that means, I can find out the probabilities, but I have to use the probabilities and the tags in the next step also. So, this is not fully independent. I cannot just choose IN and forget about it I have to remember what is the probability next time I will choose IN, VBZ, JJ VBZ and try to compute the probability for VBZ multiplied by

the previous probabilities and that is how exactly we do that I will show you in the beam search algorithm.

(Refer Slide Time: 27:34)




So, what is the beam search algorithm? So, your test sentences containing words like w 1 to w n and let say that for the ith word as s i j denotes the jth highest probability tag. This is probability tag sequence up to the word w i. Now, how does the algorithm work? So, you have the words w 1 to w n. For word w 1, you have s 1 1, s 1 2 up to s 1 N top and highest probability tags. So, we will come to the probabilities and this is easy again you will use the features here which feature is 1, which feature is 0 then you will do the normalization you will compute all these probabilities for the each of these tags.

Now, the important thing is how do you go to w 2. So, one thing to understand is that you can probably not keep all the possible tag sequences, because assume that each word can take on an average k tags. So, if you have N-words, the tag would be of the order of k to the power n, yes k to the power. So, this will be exponential in the length of the sentence. So, you cannot keep all the tag sequence. So, you might have to make the choice of the best sequence is available at this point that is what we do in bean search. So, at each point, I will select what is the top set up capital N sequences at this point and how do we how do we select that, so that is the next part of the algorithm.

So, initialize i is equal to 2. So, now you have to second word, generate tags for w i given s i minus 1 j as previous tag context and append each tag to s i minus j to make a

new sequence and then continue for all the tags. Now, what does that mean? Initialize i is equal to 2 that is your second word. Now, generic tags for w i s given s i minus j as the previous tag context what is i minus j, i minus 1 is s 1 and j. So, let me take this as the previous context. And suppose this has again some tags like I will just write some tags t 1, t 2, t 3. So, s 1 1 and t 1 becomes the first sequence; s 1 1 and t 2 becomes the second sequence; and s 1 1 t 3 becomes the third sequence. Similarly, s 1 2 t 1 becomes forth sequence and so on. So, in this case, there will be three times n sequences. So, this you can take in general case this may not be one tag this will be a sequence up to this point. So, we have all the first sequence at this point.

Now, for each of the sequence, you know what are the previous tags that have been assigned? Yes, so I know what are tags assigned at this point. So, I can use all the features. So, if a feature says t i is equal to something and t i minus 1 is equal to something else. So, I can check if t i matches this, and the previous tag matches this then only it will do 1, other wise 0, because I am choosing the context, I can compute all these feature. I can compute these features; I can compute the probability for all the sequences. And now what I will do I will select at this point what are the s i 1 to s i N, what are top end sequences at this point. And again I will go to the third word, again its tags combine with all the sequences and choose top end from here. So, what will happen use your space will not do exponentially, you will always have top end sequence is at any given point.

So, finally, when you go to the final word, you will get the top end probabilities or all the probabilities, and you will select the best sequence. And now because the sixth is sequence contains the tag from by starting, you will find out the part of speech tag just from the choosing the best sequence at the last word and that is your algorithm. So, find N highest probability sequences by above loop set this accordingly and repeat if i less than equal to n. And return to highest probability sequences x n 1 for the last word, and this is your maximum entropy this you can call as maximum entropy Markov model. Instead of maximum entropy model is generate classifier that can do for each word individually, but when you are trying to applied for a sequence labeling problem for a sequel, you use this maximum entropy Markov model. So this was maximum entropy Markov model for you.

Now, in the next lecture, I will take one simple example of this search, and I will quickly cover the model of conditional in the fields that is one of the state of the arts in sequence labeling task. And what we will show, there is some problem with the maximum entropy model that condition of fields try to handle.

Thank you, I will see you in the next class.