Natural Language Processing

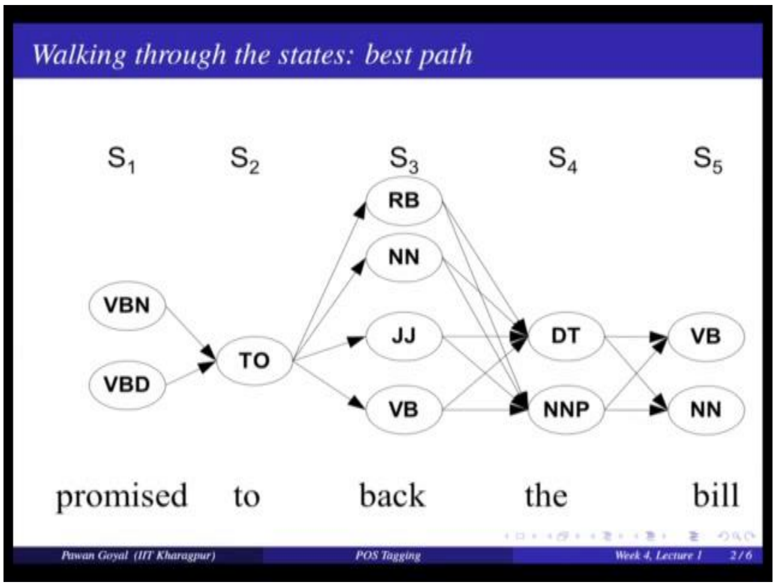
Prof. Pawan Goyal
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 16
Viterbi Decoding for HMM, Parameter Learning

Welcome back for the 4th week of the course. So, in the last week we had discussed a lot about part of speech tagging and what are the various algorithms as such one can use to solve this problem and we started with one particular classifier that is HMM model for doing part of speech tagging. So, we had gone through the basics of HMM and what are the different probabilities then one need to use to be able to come up with the best sequence given a new sentence. So, what do I mean by sequence given new sentence I want to find out what are the part of speech tags for each of the word in the sentence. So, there are multiple; there are many many possibilities and you need to find out what is the best probable tag sequence. So, we formulated this problem as noisy in the noise channel model framework and then we came up with the hidden Markov model by using certain assumptions.

So, in the last class so we had also discussed a bit about what we will be doing at run time when we are given a sentence, so let me start from there and we will see a particular algorithm on the Viterbi decoding to find out the actual tag sequences in a very very efficient manner and then later on I will go to the parameter learning part. So, we had talked about in the last week also, but we will devote sometime over that in this week.

(Refer Slide Time: 01:53)


Walking through the states: best path

$$\begin{tikzcd}
 & & RB \arrow[rd] \arrow[rdd] & & \\
 & & NN \arrow[r] \arrow[rd] & DT \arrow[r] \arrow[rd] & VB \\
VBN \arrow[r] & TO \arrow[ruu] \arrow[ru] \arrow[r] \arrow[rd] & JJ \arrow[ru] \arrow[rd] & NNP \arrow[ru] \arrow[r] & NN \\
VBD \arrow[ru] & & VB \arrow[ruu] \arrow[ru] & &
\end{tikzcd}$$

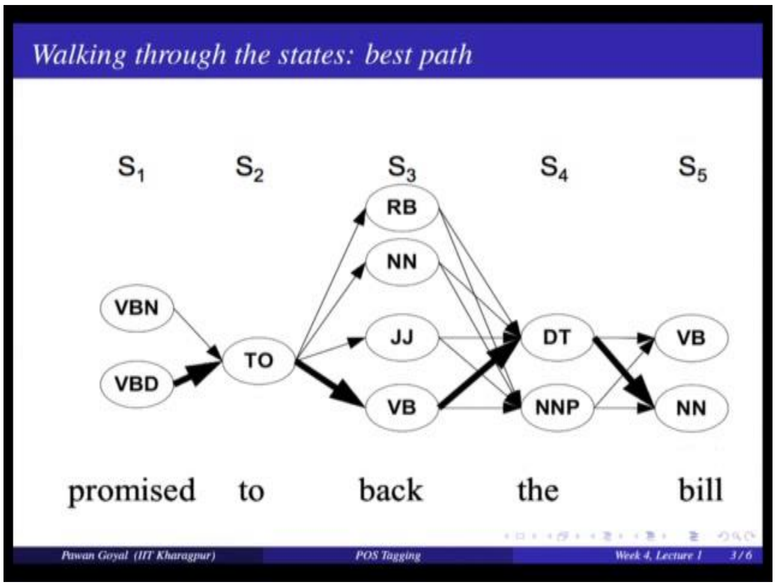
We had seen this figure in the last week itself. So, we have the sentence promised to back the bill and I want to obtain the part of speech tag sequence for the sentence. So, how do I start? I first start by finding out what are the probable part of speech tags for each of the individual words in the sentence. So, further word here promised the 2 possible tags are VBD and VBN, it can be past tense or the participle for the word to there is only one part of speech tag possible for the word back there are 4 part of speech tags possible, we have seen that in one of the lectures for the word the; there are 2 part of speech tags possible for the word bill there are 2 word of speech tags possible. So, I start by enumerating all the possibilities for each of the individual words in the sentence.

Now, I want to find out what is the actual part of speech tag sequence for this word for this sentence? Now looking at this picture, how many different possibilities can you see? So, if you start from any of the part of speech tag for the word promised and go to any of the participant tag for the word bill, any particular path denotes a possible tag sequence. So, if you try to count the number of possibilities so it will be 2 times, 1 times, 4 times, 2 times 2, it will be roughly 32 possible part of speech tag sequences and among the study to; I have to find out, what is the most probable tag sequence.

Now, how should I do that? So, 1 knife method would be that I enumerate all the possible sequences. So, all the 32 sequences I compute the probability for each sequence. So, now, you know how to compute the probability by using the HMM kind of model.

We did one simple example in the last class and find out which of the tag sequence has the highest probability that will be a naive method, you might even do that for if you have say only 32 or 50 possible text sequences, but think about the sentences that might have 10 to 15 words and some words might have 5 to 6 different tags. So, you will see that it will grow exponentially. So, enumerating all the sequences and computing the probabilities is not an efficient solution. So, you might want to do something more efficient.

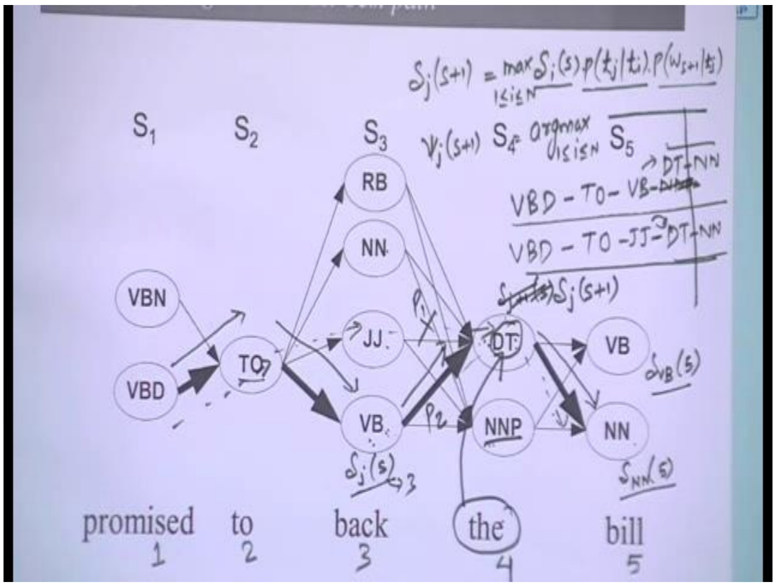
(Refer Slide Time: 04:27)



So that you can find out what is the most probable tag sequence in a very very efficient manner. So, what should be in good algorithm? So, let me take this figure; this example to give you the intuition on what kind of algorithm would be used for finding the actual tag sequence, to do it efficiently. So, the idea here is that if you are enumerated, all the 32 possibilities you are doing some computations again and again. So, can you try to reduce the time by doing it in a dynamic programming approach where you first store computations for the smaller paths and use that for the larger paths? So, we have seen one particular example of dining programming in the case of added distance.

We will try to use a similar concept here for decoding the actual tag sequence and this will be called Viterbi decoding. So, looking at the picture, so how you can save the computation time?

(Refer Slide Time: 05:42)



Let us take this sentence here, now let us; let me take 2 different paths. So, onw path that is there in the actual sequence S VBD TO VB DT and NN and let me take another path that is VBD TO then JJ then DT and then NN. So, there are 2 paths that we start with the same tag and end with the same tag. So, in VBD TO VB NNP sorry, DT NN and VBD TO JJ DT NN, So now, they are 2 different tag sequences and I want to find out probability of both of these to find out which one is better. Now what is the idea of Viterbi decoding? The idea is that. So, you are ending with the same sequence here DT and NM. So, let us look at only till that point till DT because after DT the computation will be the probabilities will be the same.

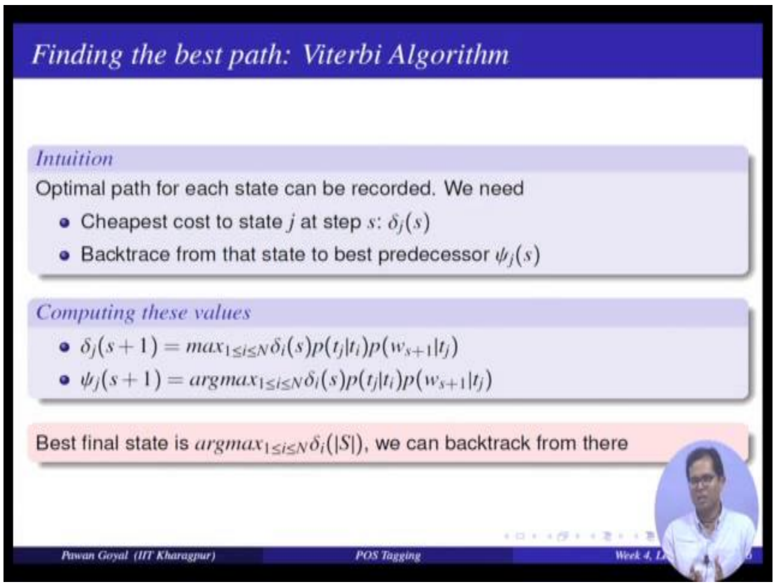
This is one way of saving computation in both of these the composition of DT will be the same. So, you can save on that, but what is more important here. So, when you arrive at the tech DT, in one case you are coming from VB, another case you are coming from JJ. So, there are 2 different ways you are coming to DT in these 2 sequences, let us suppose that at this point, I can store for the word or further tech DT, what is the best way of arriving at DT is it via this path or is it via this path. So, I suppose I can find out at this point what is the best way of arriving a DT is it this path of that path.

If I can store, if I can somehow find that the later computations will, so for the 2 paths the later computations will be the same. So, then I do not need to compute these probabilities again for the next sequence. So, if I can store at this point that is the best way is coming from VB and not from JJ. So, then I am doing it for the next steps that are going by DT, I do not need to explore this path. So, this is the idea suppose I write them as steps, step 1, 2, 3, 4 and 5 at each step for each part of speech tag if I can store, what is the best previous tag from which it is it should come? So, which previous tag will maximize the probability of arriving at this particular tag, then the computation of shared for the same path and I do not need to repeat the same computation again and again.

So, in this particular example, once I can store at step 4 for DT, the best way is coming from VB and not from JJ when I go further, I need not worry about this path at all, I can simply continue with this path because if this probability is P 1, this is P 2, the next probability will be shared, it will be times probability of NN given DT and probability of bill given NN that will be shared across the 2 paths. So, if I know at this point P 2 is higher than P 1, I can probably I can forget this path for going via DT and this is the idea this I will do for each of the speech step.

Similarly for step 3 for each of the 4 part; 4 part of speech tags like for VB I will. So here it is easy because for each part of speech tag it can come only via TO. So, it is important for this case because for DT it can come via 4 previous tags. So, I needed store which of the previous tags it is coming from which has the highest probability similarly fine and P which of the 4 previous tags has the highest probability. So, that is what I am going to do at this step for each part of speech take I will store what is the best probability of arriving at this part of speech.

(Refer Slide Time: 11:14)



Finding the best path: Viterbi Algorithm

Intuition

Optimal path for each state can be recorded. We need

- Cheapest cost to state j at step s: $\delta_j(s)$
- Backtrace from that state to best predecessor $\psi_j(s)$

Computing these values

- $\delta_j(s+1) = max_{1 \le i \le N} \delta_i(s) p(t_j|t_i) p(w_{s+1}|t_j)$
- $\psi_j(s+1) = argmax_{1 \le i \le N} \delta_i(s) p(t_j|t_i) p(w_{s+1}|t_j)$

Best final state is $argmax_{1 \le i \le N} \delta_i(|S|)$, we can backtrack from there

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, L

Let us look at the formulation, so intuition is that so as I have explained, we are recording what is the optimal path at each state for a given step, what we are storing, what is the cheapest cost of arriving at a particular state; that means, that has the highest probability in the cheapest cost. Now once we have found out what are the tags at different points we also want to write down the sequence. So, I also need to store batteries. So, this is the highest probability. So, what is the particular part of speech tag that gives the highest, this highest probability? So, I want to store this delta j S what that is the cheapest cost or the highest probability to step j at step S and also the back trace from that state to the best predecessor. So, for the previous example that we were seen for DT the best predecessor was VB and not j.

Now this I am doing at any step as for state j how do I use that to compute for the next step? So, let us again look at it. So, suppose I have stored delta j S. So, here S is 3 and j is my state, it can be VB JJ NN and RB. So, I have stored for all these part 4 different part of speech tags what is the cheapest cost of arriving at this point, what does that mean? I know at this point which path is better VBD TO VB or VB NT or VB. So, I know that for all these 4 different part of speech tags. So, I store the step for this step for this state what is the optimal cost other what is the most probable path.

So, this cost I have or this probability I have now how do I use this probability is to compute for the next step say want to compute delta j plus 1 S, how will I compute that?

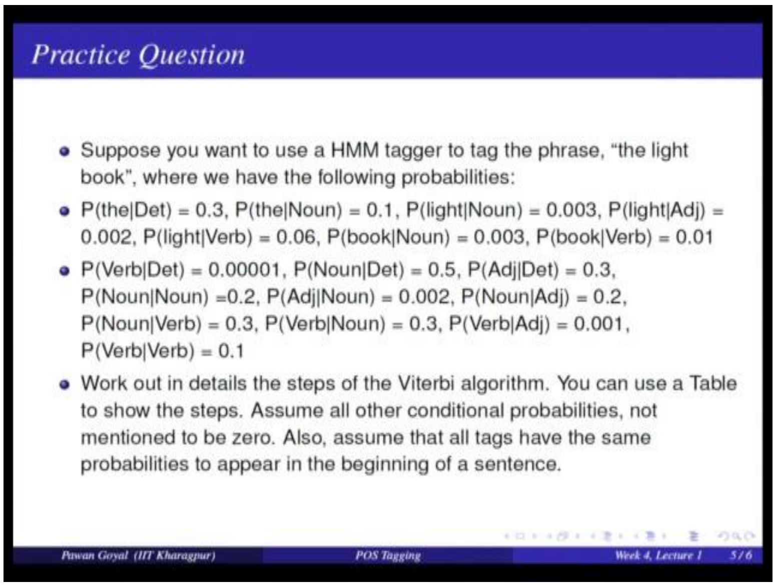
So, now, delta g S would have taken care of the word back already. So, what are the things that I added for going from? I am sorry; here this should be stepped yes. So, this is delta j S plus 1. So, this is a step S plus 1 I want to find out probability delta j.

I know the probabilities are step S, now what is the additional information the transition from this is state to this state and the emission probability of this word given is the state. So, I can compute it using delta j S plus 1 is delta i S times the probability of this tag sequence that is probability of tag t j given t i this is the probability of tag sequence transition and the emission probability that is probability of word edge step S plus 1 given t j.

This I am doing for some ith state, but there are 4 P, there can be some N Number of states. So, I will find out the max of those among all the states at the previous step I have delta S i multiplied with it the transition probability and the emission probability at step S plus 1 and the max gives me what is the best path at this is state j at step as for this 1. So, now, suppose out of these 4, I found out this is the best path. So, we will also like to store what is the best predecessor and this is nothing, but argmax of this function. So, I call that size S plus one that is argmax of the same function and that is what I am also storing and that is all this will continue. So, what will happen at the end I am at a step 5 and I know, what is the probability? So, let me write down delta NN at 5 and delta VB at 5, I know both these probabilities, what is the best way of reaching at this state, what is the best probability of reaching at this state at this fine I can just take the maximum of these 2 and this will give me the best probability of any possible part of speech tag sequence and then I can use the back traces from here to obtain the actual part of speech tag sequence and this is my vita by decoding algorithm.

So, if we see that formally, so yes, so I am storing at each step S, what is the cheapest cause of the best probability of reaching there also the predecessor and using that I am also computing the probabilities for the next step that is. So, that is similar to what we have seen here the probability at the previous step transition probability and emission probability then I take the max over all the previous possible states also show the predecessor for the next step and how do we end once we have gone through the sentence end I will have the probabilities for all the states possible at the end the ending word I will take the maximum of those n back trace from there and that will give me the optimal automat tag sequence for this particular sentence.

(Refer Slide Time: 18:26)



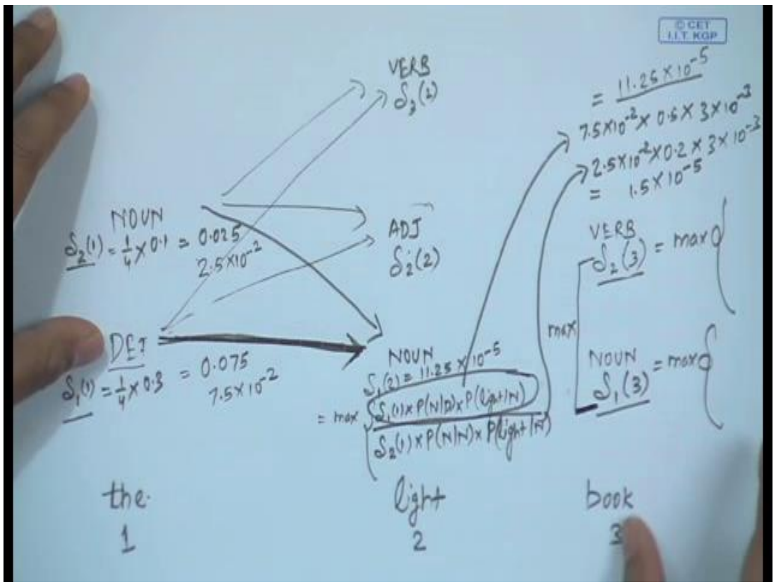
Practice Question

- Suppose you want to use a HMM tagger to tag the phrase, "the light book", where we have the following probabilities:
- P(the|Det) = 0.3, P(the|Noun) = 0.1, P(light|Noun) = 0.003, P(light|Adj) = 0.002, P(light|Verb) = 0.06, P(book|Noun) = 0.003, P(book|Verb) = 0.01
- P(Verb|Det) = 0.00001, P(Noun|Det) = 0.5, P(Adj|Det) = 0.3, P(Noun|Noun) =0.2, P(Adj|Noun) = 0.002, P(Noun|Adj) = 0.2, P(Noun|Verb) = 0.3, P(Verb|Noun) = 0.3, P(Verb|Adj) = 0.001, P(Verb|Verb) = 0.1
- Work out in details the steps of the Viterbi algorithm. You can use a Table to show the steps. Assume all other conditional probabilities, not mentioned to be zero. Also, assume that all tags have the same probabilities to appear in the beginning of a sentence.

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 1 5 / 6

I hope this algorithm is clear. So, let us take a simple example. So, that you can become much more familiar with that how do you use this video decoding algorithm finding problem. So, what do you see?

(Refer Slide Time: 18:47)


© CET I.I.T. KGP
VERB
$\delta_3(2)$
$= 11.25 \times 10^{-5}$
$7.5\times10^{-2} \times 0.5 \times 3\times10^{-3}$
$2.5\times10^{-2}\times0.2\times3\times10^{-3}$
$= 1.5\times10^{-5}$
NOUN
$\delta_2(1) = \frac{1}{4}\times0.1 = 0.025$
2.5×10^{-2}
ADJ
$\delta_2(2)$
VERB
$\delta_2(3) = \max$
DET
$\delta_1(1) = \frac{1}{4}\times0.3 = 0.075$
7.5×10^{-2}
NOUN
$\delta_1(2) = 11.25\times10^{-5}$
$= \max \delta_1(1)\times P(N|D)\times P(light|N)$
$\delta_2(1)\times P(N|N)\times P(light|N)$
max
NOUN
$\delta_1(3) = \max$
the
1
light
2
book
3

In this problem I have a sentence the light book and you are given some sort of emission probabilities first that the word the can come from the tag determiner or noun. So, they are 2 possibilities determiner noun the word light can come from noun adjective and verb possibility by using vita b decoded algorithm. So, how do we start? So, I have 3 steps 1 2

3. So, I want to store at each step for each state. So, let me call it delta one at step one similarly delta 2 as step one. So, what is this probability? So, this will be what is the probability that the tag determiner occurs in the start of the sentence multiply with the probability of emitting the from the tag determiner.

Let us see how these probabilities given in this case probability of determine occurring in the start of the sentence. So, in this question in the last sentence, it is written assume that all the tags that have the same probability stop in the beginning of the sentence. So, probabilities are given implicitly. So, now, the question is it how many tags would you assume. So, here you can take any assumption. So, in this question you are shown only 4 tags determiner noun verb an adjective. So, assume there are only 4 part of speech tags and each one has equal probability of occurring at the start of the sentence. So, what will the probability of determine occurring at the start of sentence it will be one by 4 similarly for noun one by 4 times probability of the given determiner and that is given here h point 3 I am probability of the given noun is given as point 1. So, this is your delta 2 1 and delta 1 1. So, you can also further write them oh 0.025 and this will be 0.075. So, better write them as 2.5 into 10 to power minus 27.5 into 10 to the minus 2.

You have delta 1 1, del 2 1, now you go to the next step. So, now, here you want to find out delta 1 2 in step 2, what is the best probability of reaching noun? You can reach via either here or here. So, you need to compute both the possibilities and take the max over that. So, let us compute these probabilities. So, what will be that so, first will be delta 1 1 times probability of getting noun from determine times probability of light given noun S and second one will be delta 2 one times probability noun given noun times probability light given noun. So, let us compute these 2 probabilities. So, you will take the max of these.

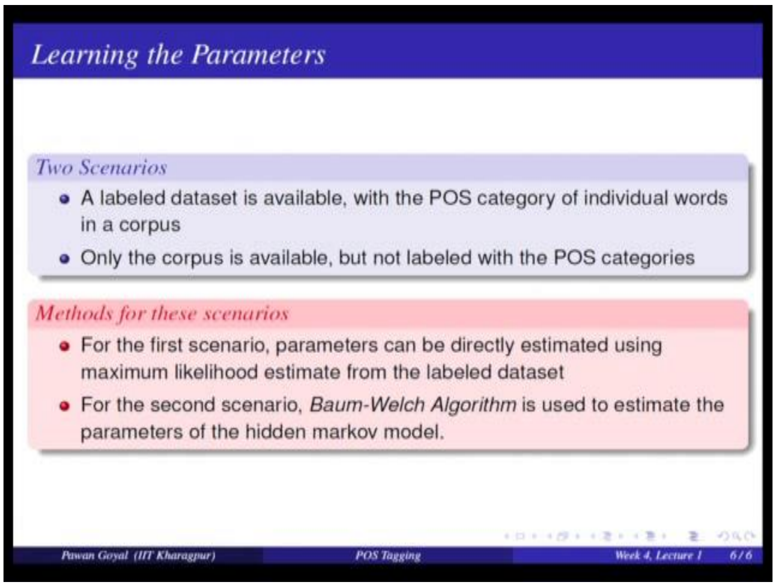
So, let us compute these probabilities. So, this would be. So, let me take it here seven point 5 into 10 to the power minus 2 probability of noun given to determiner. So, that is that we can see from the table given to us noun given determiner point 5 and light given noun each 3 into 10 to power minus 3 and this gives me 3.75 into 3 11.25 into 10 to the minus 5 and the second one becomes 2.5 into 10 to power minus 2 times probability of n; given n, this is 0.2 into probably at given and that will be 3 into 10 to the minus 3 and that is 0.5 1.5; 1.5 into 10 to power minus 5 and from these 2 you can say that this one is higher than the next one. So, this is what you will take. So, you can store delta 1 2 H

11.25 into 10 to power minus 5 and you will also store the back trace that is coming from this path same thing you will do for adjective and verb adjective again there will be 2 possibilities coming from determiner and noun for work there will be 2 possibilities again you compute the probabilities using the formula and store what is the best probability. So, it will be delta 3 at step 2 delta 2 h step.

This is what you will store; also you will store the back trace, once you have done that now we will go to the final step. So, where you will store delta 1 step 3 1 means noun here and delta 2 at the step 3 again you will compute that by taking all the 3 possibilities times the transition and all. So, this will be max over 3 quantity this will be max over 3 quantities again you will find out the 2 values delta one 3 and delta 2 3 and how will you actually determine the finest sequence you will take the max of this suppose max comes out to be this one then you will go to the predecessor suppose the predecessor was this one you will go to hear suppose predecessor of as this one and so on. So, you will find out this sequence assuming this is these are the best fervent.

But I hope the idea is clear that how do you actually do this computation? So, I would encourage that you complete this particular exercise on your own.

(Refer Slide Time: 27:26)



Learning the Parameters

Two Scenarios

- A labeled dataset is available, with the POS category of individual words in a corpus
- Only the corpus is available, but not labeled with the POS categories

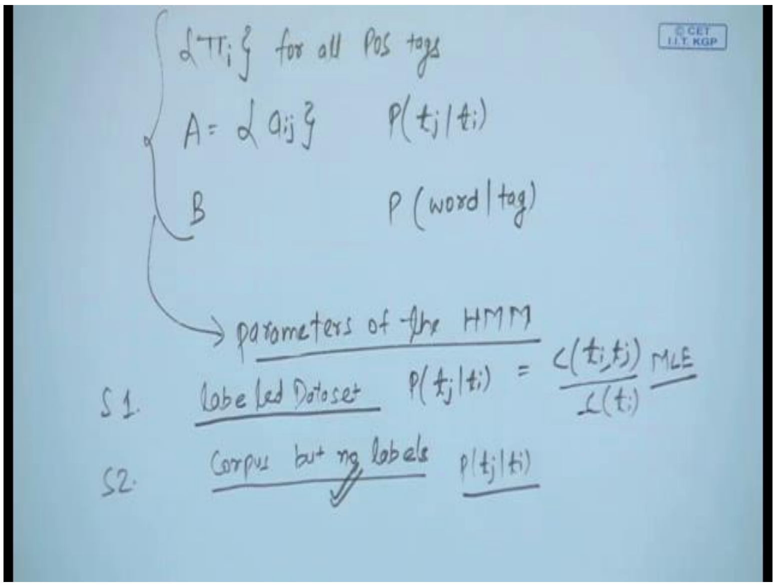
Methods for these scenarios

- For the first scenario, parameters can be directly estimated using maximum likelihood estimate from the labeled dataset
- For the second scenario, *Baum-Welch Algorithm* is used to estimate the parameters of the hidden markov model.

Pawan Goyal (IIT Kharagpur) POS Tagging Week 4, Lecture 1 6 / 6

Now, coming to the point of learning the parameters so, what are the parameters do we need in the; so, we need 3 different parameters. So, we need the probability of starting the sentence.

(Refer Slide Time: 27:50)



$\{\pi_i\}$ for all POS tags

$A = \{a_{ij}\}$ $\quad P(t_j|t_i)$

B $\quad P(\text{word}|\text{tag})$

$\rightarrow$ parameters of the HMM

S1. Labeled Dataset $\quad P(t_j|t_i) = \frac{c(t_i,t_j)}{c(t_i)}$ MLE

S2. Corpus but no labels $\quad P(t_j|t_i)$

That is let me call this in the language of HMM the pi that is what is the probability that a particular tag or estate start the sequence. So, I want this probability pi i for all posh tags then I need the probability of transition from one part of speech at another. So, this is my transition matrix I need all this a i j transiting from probability T j given T i S and i need my emission probabilities that is probability of the word given a tag. So, you can use it using a matrix b. So, I need all these 3 probabilities to actually use this Viterbi decoding algorithm at runtime if I do not have these probabilities I cannot teach you Viterbi decoding.

The question is so; these are my parameters of the HM. So, how do we actually find out these parameters of my HM? So, here I am saying there are 2 different scenarios that you might have. So, first scenario is where a label data set is available what do you mean by a label data set you have a set of sentences available to you for each sentence you also know what is the actual part of speech tags sequence. So, what is the part of speech tag for individual world somebody is manually labeled it for you. So, this is a label data set this one scenario second scenario is you only have the corpus, but you do not have any data where the sentences are labeled with the part of speech categories. So, I have the tooks that is a clear. So, scenario one where you have a label data set a scenario to only corpus, but no labels. So, what I am saying for these 2 scenarios you can find the parameters in different manner. So, suppose you have scenario 1 you have a label data set. So, what you can do for finding the parameters of HMM. So, this is something we also took them the in the last week.

For example, if the label data set, I want to find a parameter like them the transition probabilities probability of tags a given tag I, how will you find that the label data set you will have all the possible tag sequences that actually occurred in the corpus now from that you will find out number of times t i n t j occur together in the corpus divided by number of times only t i occurs. So, this was what we said as maximum likelihood estimator you can use that to find this probability same with all the other probabilities pi i as well as P you can find using maximum likelihood estimate.

Now, problem comes when you have the corpus, but no labels. So, now, there are no labels. So, how would you actually find out probability of t j given t i when in the in the data there is no label on which word got a tag t I or which one got it at t j. So, you cannot compute it directly from any labels. So, what is the method that you will use for learning the parameters when the corpus is available, but labels are not available and that is what we will see in the next lecture.

We will be using Baum Welch algorithm to estimate the parameters of the hidden Markov model when the labels are not available. So, this is similar to expectation maximization algorithm and you will see what is the addition for this algorithm and how we actually apply this. So, hope in this lecture you understood how to apply with Viterbi decoding when the parameters of HMM are given and so you can do that for (Refer Time: 32:32) and in the next lecture we will discuss how do we run the parameters when the labels are not available.

Thank you.