

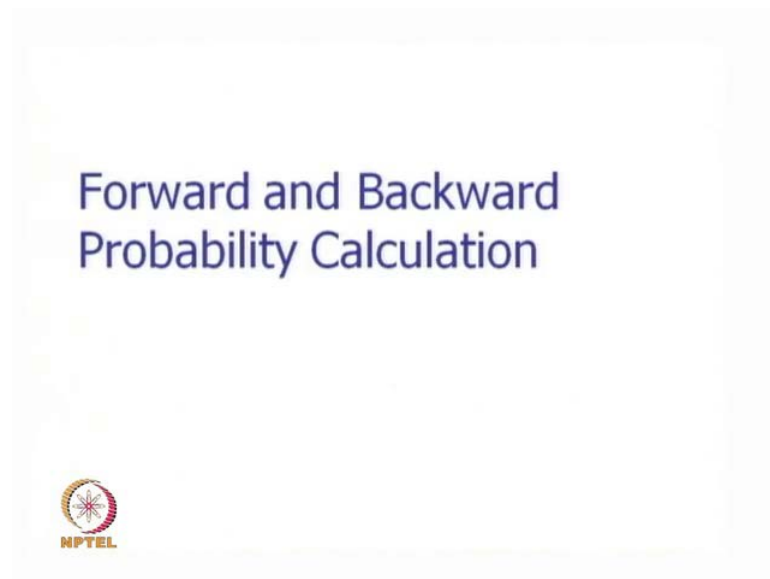
**Natural Language Processing**  
**Prof. Pushpak Bhattacharyya**  
**Department of Computer Science & Engineering,**  
**Indian Institute of Technology, Bombay**

**Lecture - 21**  
**HMM, Forward and Backward Algorithms, Baum Welch Algorithm**

Today, we would like to discuss the training of the hidden Markov model, this is an supervised learning, the strings which are output by the machine or observed from the machine are given. And because each output sequence can be the result of more than one state sequence, we need an algorithm for training the hidden Markov model, whereby we obtain the transition and observation probabilities of the machine or a combinations thereof.

So, in the last lecture, we had discussed forward algorithm and backward algorithm and how to compute them efficiently; these two algorithms are used in the training of the hidden Markov model. That is why the training algorithm is also known as the forward backward algorithm or more famously possibly as the Baum Welch algorithm. So, we proceed with the material on the slide. And I have written here, HMM Forward and Backward Algorithm, Baum Welch algorithm.

(Refer Slide Time: 01:38)



So, just to remind ourselves of the forward and backward probability calculation.

(Refer Slide Time: 01:44)

## Forward probability $F(k,i)$

- Define  $F(k,i)$  = Probability of being in state  $S_i$  having seen  $o_0o_1o_2\dots o_k$
- $F(k,i) = P(o_0o_1o_2\dots o_k, S_i)$
- With  $m$  as the length of the observed sequence
- $P(\text{observed sequence}) = P(o_0o_1o_2\dots o_m)$   
 $= \sum_{p=0,N} P(o_0o_1o_2\dots o_m, S_p)$   
 $= \sum_{p=0,N} F(m, p)$

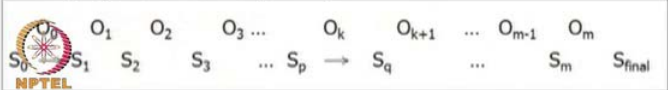


We remember we defined  $F(k,i)$  as the probability of being in state  $S_i$  having seen  $o_0o_1o_2$  up to  $o_k$ , which is the observation sequence up to the  $K$ th time step. So, this particular expression was defined as the forward probability and the expression for that is  $P(o_0o_1o_2 \dots o_k, S_i)$ . So, the machine has seen up to this symbols or the machine might have output, this symbols  $o_0$  up to  $o_k$  and then enters the state  $S_i$ .  $m$  is the length of the observed sequence, so one can write the probability of the observed sequence as  $P(o_0o_1o_2 \dots o_m)$  and which is equal to  $\sum_{p=0,N} P(o_0o_1o_2 \dots o_m, S_p)$ , the state introduced, this is marginalization and it can be written as the sum of forward probabilities  $F(m, p)$  with  $p$  varying from 0 to  $N$ .

(Refer Slide Time: 03:03)

### Forward probability (contd.)

$$\begin{aligned}
 F(k, q) &= P(o_0 o_1 o_2 \dots o_k, S_q) \\
 &= P(o_0 o_1 o_2 \dots o_k, S_q) \\
 &= P(o_0 o_1 o_2 \dots o_{k-1}, o_k, S_q) \\
 &= \sum_{p=0, N} P(o_0 o_1 o_2 \dots o_{k-1}, S_p, o_k, S_q) \\
 &= \sum_{p=0, N} P(o_0 o_1 o_2 \dots o_{k-1}, S_p) \cdot P(o_k, S_q | o_0 o_1 o_2 \dots o_{k-1}, S_p) \\
 &= \sum_{p=0, N} F(k-1, p) \cdot P(o_k, S_q | S_p) \\
 &= \sum_{p=0, N} F(k-1, p) \cdot P(S_p \xrightarrow{o_k} S_q)
 \end{aligned}$$



$O_0 \quad O_1 \quad O_2 \quad O_3 \quad \dots \quad O_k \quad O_{k+1} \quad \dots \quad O_{m-1} \quad O_m$   
 $S_0 \quad S_1 \quad S_2 \quad S_3 \quad \dots \quad S_p \quad \rightarrow \quad S_q \quad \dots \quad S_m \quad S_{final}$

Then, we also found a method to compute the forward probability recursively, from its definition itself. So,  $F(k, q)$  is a probability of seeing,  $k + 1$  symbols  $O_0$  up to  $O_k$  and then being in state  $S_q$ . So, this then can be broken into 2 parts, the symbol of the sequence itself, so  $O_0$  up to  $O_{k-1}$ , the last symbol is isolated, which is  $O_k$  and  $S_q$  remains.

So, we now refer to the whole situation where,  $O_0$  up to  $O_m$  is a complete observation sequence, after seeing  $O_m$ , the machine enters the final state  $S_{final}$ , which is one of the given states, in the hidden Markov machine and  $S_0$  is the starting state. So, as is shown in the output and state sequence, the transitions are state to state with a particular output symbol shown on top of the arrow. So, we see that, this  $O_k S_q$  is there, in the expression and then we introduced  $S_p$  and this is marginalisation  $P$  going from 0 to  $N$  sum is also introduced.

So, now we have this pattern here,  $S_p O_k S_q$ . So, this is the pattern, which exists in the observation sequence state sequence. So, we isolate this part probability  $O_0$  up to  $O_{k-1}$  with  $S_p$ , this part is isolated, this is then multiplied by the probability of  $O_k S_q$  given  $O_0$  up to  $O_{k-1}$  and  $S_p$ . This nothing but the chain rule and in the next step, the only conditioning factor  $S_p$  is written nothing else affects, this probability of the observed symbol  $O_k$  and the next state  $S_q$  and therefore, we drop this whole sequence  $O_0$  up to  $O_{k-1}$ .

Now,  $P(o_k | S_{k-1})$  with  $S_{k-1}$  nothing but the forward probability  $F_{k-1}$  and  $P(o_k | S_k)$  is written as the transition  $S_{k-1}$  to  $S_k$  with  $o_k$  as the output symbol. So, there is whole probability  $F_k$  is nothing but  $\sum P_{k-1}$  going from 0 to N and each  $F_{k-1}$  is multiplied, by the transition probability  $P_{k-1}$  to  $S_k$  with  $o_k$ . So, this whole deduction is completed by applying marginalisation here and then chain rule here and the Markov assumption after that, marginalisation chain rule Markov assumption.


And this produces the recursive expression, for  $F_k$  and it is easy to see that  $F_k$  can be looked upon as, the sum of the feeding probabilities  $F_{k-1}$  multiplied by the transition probability. And each of this can be computed from  $F_{k-2}$  and another index, let us say  $t$  varying from 0 to N and this way the whole computation can be completed.

The boundary condition for this comes from the initial probability  $F_0$  and  $P$  where,  $o_0$  is the symbol seen, which is nothing but  $\epsilon$  and having seen  $\epsilon$ , what is the state. So, these are the initial probability values of the states, the probability that the machine is one of these initial states, so this is the way,  $F$  is computed from output sequence, state sequence and the expression for it.

(Refer Slide Time: 07:31)

### Backward probability $B(k,i)$

- Define  $B(k,i)$  = Probability of seeing  $o_k o_{k+1} o_{k+2} \dots o_m$  given that the state was  $S_i$
- $B(k,i) = P(o_k o_{k+1} o_{k+2} \dots o_m | S_i)$
- With  $m$  as the length of the observed sequence
- $P(\text{observed sequence}) = P(o_0 o_1 o_2 \dots o_m)$   
 $= P(o_0 o_1 o_2 \dots o_m | S_0)$   
 $= B(0,0)$



Similarly, we could compute the backward probability  $B_k$ , we remember  $B_k$  is defined to be the probability of seeing  $o_k o_{k+1}$  up to  $o_m$ , given that

the state was  $S_i$ . So,  $B_k$  comma  $i$  is nothing but probability of  $O_k$  to  $O_m$  given  $S_i$  with  $m$  is the length of the observed sequence.


So, from this definition, it is clear that probability of the observed sequence is nothing but the probability of the observed sequence given the initial state as  $O_0$ , so this is nothing but  $B_0 O_0$ . So, the first  $O_0$  is the starting index of the sequence, we will observe until the end and the second  $O_0$  is the state, in which the machine is before this observation sequence is seen.

(Refer Slide Time: 08:23)

### Backward probability (contd.)

$$\begin{aligned}
 B(k, p) &= P(O_k O_{k+1} O_{k+2} \dots O_m | S_p) \\
 &= P(O_{k+1} O_{k+2} \dots O_m, O_k | S_p) \\
 &= \sum_{q=0, N} P(O_{k+1} O_{k+2} \dots O_m, O_k, S_q | S_p) \\
 &= \sum_{q=0, N} P(O_k, S_q | S_p) \\
 &\quad P(O_{k+1} O_{k+2} \dots O_m | O_k, S_q, S_p) \\
 &= \sum_{q=0, N} P(O_{k+1} O_{k+2} \dots O_m | S_q) \cdot P(O_k, S_q | S_p) \\
 &= \sum_{q=0, N} B(k+1, q) \cdot P(S_p \xrightarrow{O_k} S_q)
 \end{aligned}$$

$O_0$	$O_1$	$O_2$	$O_3$	...	$O_k$	$O_{k+1}$	...	$O_{m-1}$	$O_m$
$S_1$	$S_2$	$S_3$	...	$S_p$	$\rightarrow$	$S_q$	...	$S_m$	$S_{final}$



So, again we have a recursive way of computing  $B_k p$  where,  $k$  is any index within the observation sequence and we can recursively compute this. So, since this is backward probability, we will gradually look into the smaller and smaller segment of the observation sequence towards the end.

So, naturally  $k$  will now increase in its value 1 by 1, so let us see how the process is  $P O_k O_{k+1}$  up to  $O_m$  given  $S_p$ , this is isolated now, we take up  $O_k$  separately and  $O_{k+1}$  to  $O_m$  is the shorter sequence. So, this is  $O_{k+1}$  to  $O_m$  with  $O_k$  given  $S_p$ , we introduce the variable state  $S_q$  as is seen in the picture here,  $S_p$  was the starting state before  $O_k$  to  $O_m$  was observed  $S_q$  is the starting state before  $O_{k+1}$  to  $O_m$  was observed.

So, now we have  $P(O_k + 1 \text{ to } O_m \text{ with } S_q)$  and this can now be used and  $P(O_k | S_q \text{ given } S_p)$  is isolated, now on marginalisation, now on chain rule, we have to have the term  $P(O_k + 1 \text{ to } O_m \text{ given } O_k | S_q \text{ and } S_p)$ . Now, the only conditioning factor for this term is  $S_q$ . So,  $O_k$  and  $S_p$  can be dropped, which is dropped. And the term that, we have now  $S_p | O_k + 1 \text{ to } O_m \text{ given } S_q | P(O_k | S_q \text{ given } S_p)$ , so this can be written as  $B_k + 1 \text{ comma } Q_p | S_p \text{ to } S_q$  with  $O_k$  as the upward symbol. So, therefore, we have again like forward probability a backward probability expression recursively computed from shorter segments up to the end of the sequence.

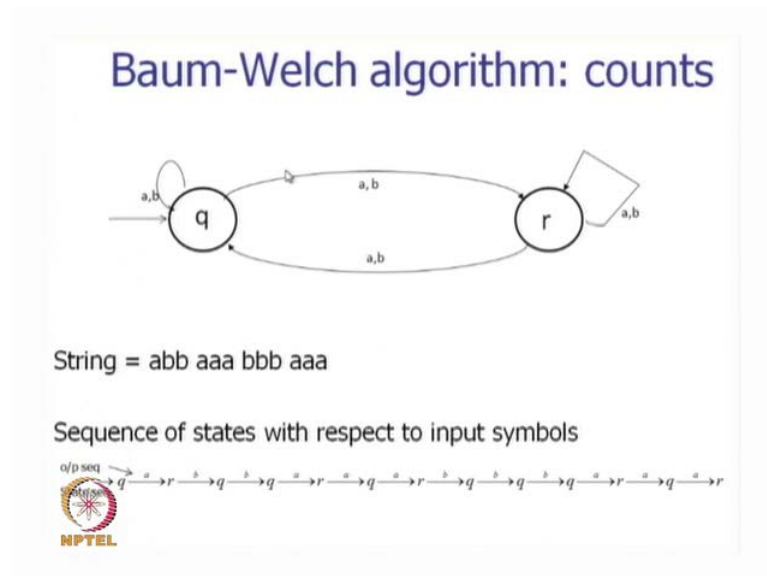
Here also the boundary condition is obtained from the final symbol  $O_m$ , the state before  $O_m$  and the final state. So, these probability values have to be given again these backward probability can be computed in linear time proportional to the length of the sequence, for which the probability backward probabilities calculated. So, both backward and forward probability can be calculated in linear in the length of the sequence. Now both these probability values are very useful for various kinds of computation involving hidden Markov model and we will now proceed to see how these 2 probability values are used for training the hidden Markov model.

(Refer Slide Time: 11:28)



So, we come to this topic of H M M training the algorithm known as Baum Welch or forward backward algorithm.

(Refer Slide Time: 11:36)



Now, to understand this algorithm, we will first clear up a notion of counts of transitions with output symbol within the sequence, which is produced by the machine. Let us suppose, that a b b a a b b b a a is the sequence, which is produced by the machine shown above, now this is a machine where all transitions are possible. And therefore, given this observation sequence many many states are possible, which produce these output sequence.

So, one of the output sequence is shown here q to r a, so the machine goes from q to r on a and from r on b, it comes back to q, then from q it takes b and remains in b, it could have gone to r also and then from q to a, it goes to r q to r it goes with a and then r it comes back to q with a q to r again with a r to q again with b and so on. So, this is the whole state sequence, which produced corresponding to the observation sequence all of a symbols are written on top of the arrows.

(Refer Slide Time: 12:58)

Calculating probabilities from table

$$P(q \xrightarrow{a} r) = 5/8$$

$$P(q \xrightarrow{b} r) = 3/8$$


$$P(s^i \xrightarrow{w_k} s^j) = \frac{c(s^i \xrightarrow{w_k} s^j)}{\sum_{l=1}^T \sum_{m=1}^A c(s^i \xrightarrow{w_m} s^l)}$$

$T = \#states$   
 $A = \#alphabet\ symbols$

Now if we have a non-deterministic transitions then multiple state seq possible for the given o/p seq (ref. to previous slide's feature). Our aim is to find expected count through this.

Table of counts

Src	Dest	O/P	Count
q	r	a	5
q	q	b	3
r	q	a	3
r	q	b	2



Now, when this sequence is given, we can make a frequentist approach to probability calculation, so for example, the probability of q going to r on a or with a is found out to be 5 by 8, how do we get this value. So, we make a record of the transitions and the corresponding output symbols, as they obtain in the output sequence. So, we find that in the output sequence and the corresponding state sequence, we have the transition source to destination q to r with the symbol a, this happens 5 times.

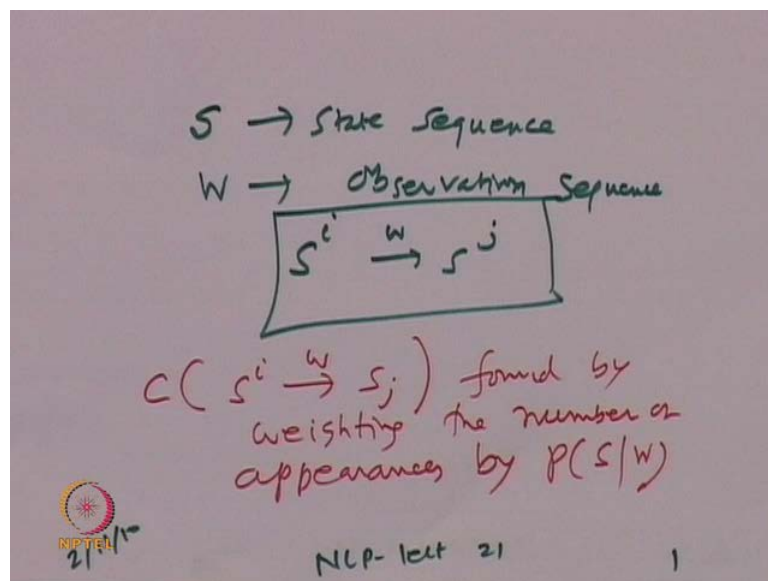
Now, to get the probability value of q to r on a, we have to find out what other transitions from q are there in the observation sequence. So, we find that, there is transition from q to q with the symbol b, so these are the only 2 cases of transition from q, with q as the source. So, therefore, it makes the total count of transition where, q is the source state as 8 5 plus 38 and out of that due to r with a is 5, that is why the probability of q to r with a is nothing but 5 by 8.

Similarly probability of q to r with b, it should be q to q with b is 3 by 8, so in general the probability of a transition from  $S_i$  to  $S_j$  with symbol  $W_k$  is the count of  $S_i$  to  $S_j$  with  $W_k$  number of times, this occurs divided by all the total number of transitions from  $S_i$  to any state with any symbol. And here, we have shown the index to be going from 1 to a where a is the alphabet symbol and this  $W_m$ , all possible symbols and l from 1 to t where,  $S_l$  is the destination state and t is the number of states alright.



So, now, the point that is being made is that, if we have a single output sequence, single state sequence corresponding to an observation sequence, then this kind of taking the count is alright, this is valid. But, if there are multiple state sequences for an observation sequence, what do we do, we have a number of state sequences for an observation sequence. So, the idea is to weight, these counts to prorated or weight, these counts with the probability of the state sequence given the observation sequence. So, you would like to write this expression and explain with the formula.

(Refer Slide Time: 16:26)



So, suppose there is a state sequence  $S$  and the observation sequence is  $W$  and suppose a pattern  $S_i$  to  $S_j$  on  $W$  appears in this, so the count of this, we are seeing will be as follows. So, the count of  $S_i$  to  $S_j$  on  $W$  will be found by waiting the number of appearances by probability of  $S$  given  $W$ , which sort of make sense, because now we have the number of these patterns, obtaining in the observation sequence and the state sequence.

And if it if this pattern appears in multiple state sequences, resulting from the observation sequence, then it makes sense to give weight age to these counts and the weight comes from the probability value of  $S$  given  $W$ ,  $P(S|W)$  and if the pattern appears in  $s$  then that count will be multiplied by the probability of  $P(S|W)$ . So, this is intuitively acceptable and this is the way, we would like to obtain the count.


(Refer Slide Time: 18:26)

### Interplay Between Two Equations

$$P(s^i \xrightarrow{w_k} s^j) = \frac{c(s^i \xrightarrow{w_k} s^j)}{\sum_{l=0}^T \sum_{m=0}^A c(s^i \xrightarrow{w_m} s^l)}$$

$$C(s^i \xrightarrow{w_k} s^j) = \sum_{s_{0:n+1}} P(S_{0:n+1} | W_{0:n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0:n+1}, W_{0:n})$$

No. of times the transitions  $s^i \xrightarrow{w_k} s^j$  occurs in the string



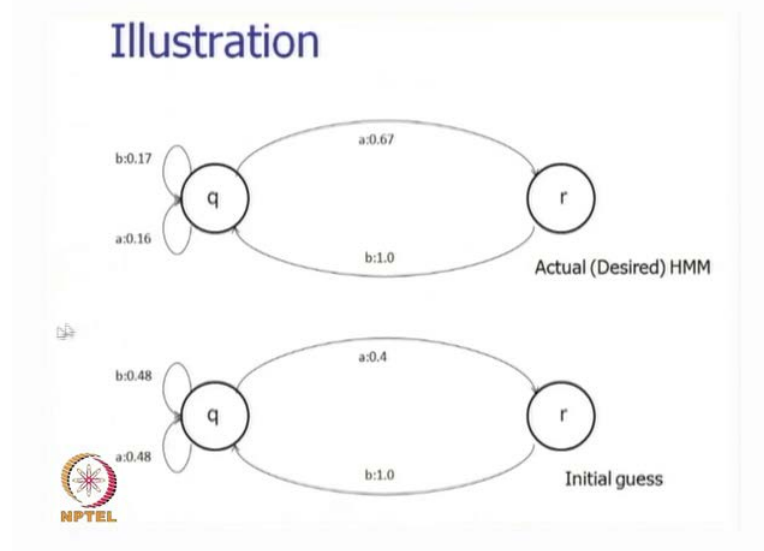
So, going to the slides once again, let's illustrate this procedure of computation, so  $P(s^i \rightarrow s^j \text{ on } W^k)$  is nothing but  $C(s^i \rightarrow s^j \text{ on } W^k)$  divided by total number of count of transitions from  $S^i$  to  $S^l$  on symbol  $W^m$ . Where, this is the count of all those patterns where,  $S^i$  is the source and any  $W^m$  and any  $S^l$  are the corresponding, output symbol and destination state respectively.  $C(s^i \xrightarrow{w_k} s^j)$  that is the count of this pattern, is obtained by finding out the number of these pattern  $S^i \xrightarrow{w_k} s^j$  in the context of the 8 sequence  $0$  to  $n+1$  and the observation sequence  $0$  to  $n$ . So, this is the context, in which we are seeing this pattern and this is the number, we weight this number by the probability of the state sequence given the observation sequence. And we have to take this expression for all possible states, which result from this observation sequence, that is why there is a sigma of all possible state sequences  $S_0$  to  $N+1$ , given the observation sequence.

So, this is a nice intuitive formula the count multiplied by the probability value and these the probability is found from this count, the count is found from the probability value. So, the way the method starts, that we first assume some probability values of these transitions from there, we obtain these count values from there, we obtain these count values, from this probability of transition, we can very easily compute  $P$   $s$  given  $W$ , this is nothing but the Viterbi algorithm.

And for bigram sequences, this is nothing but the multiplication of the transition probabilities. So, we obtain this count and from the count, we obtain new probability value from the new probability value, we obtain the new count, we obtain the new probability value. This is the way the computation proceeds from probability to count to probability.

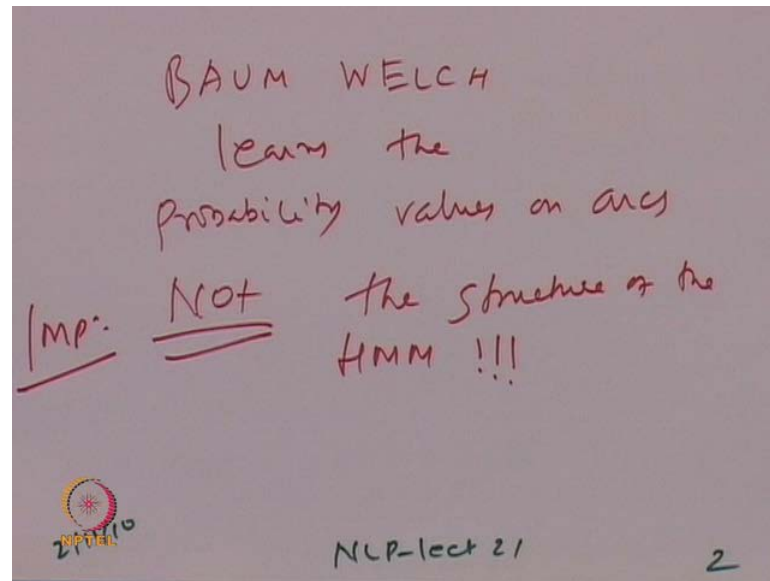
And eventually after sometime the algorithm terminates when, we see that there is no appreciable change, in the probability values, this algorithm is called expectation maximisation. So, we expect a value for the count and then maximise the probability of the observation sequence, through this alright. So, this is the basic idea and these are the 2 controlling equations.

(Refer Slide Time: 21:33)



We illustrate this procedure with this example here, here is an H M M where, the 2 states are q and r, the transitions are on a and b, it is shown that from q to r a transition can happen with a probability of 0.67 where, the output symbol is a. And from r to q the transition is on b only and this a certain transition, there is no non determinants here, so the probability is 1.0. From q v find that apart from transition to r on a, we also have transition back to q on b with probability 0.17 and back to q on a with probability 0.16. So, this is the machine, which is given to us, we have to learn this machine, now let us be careful about the meaning of learning here, you would like to write certain important comments here.

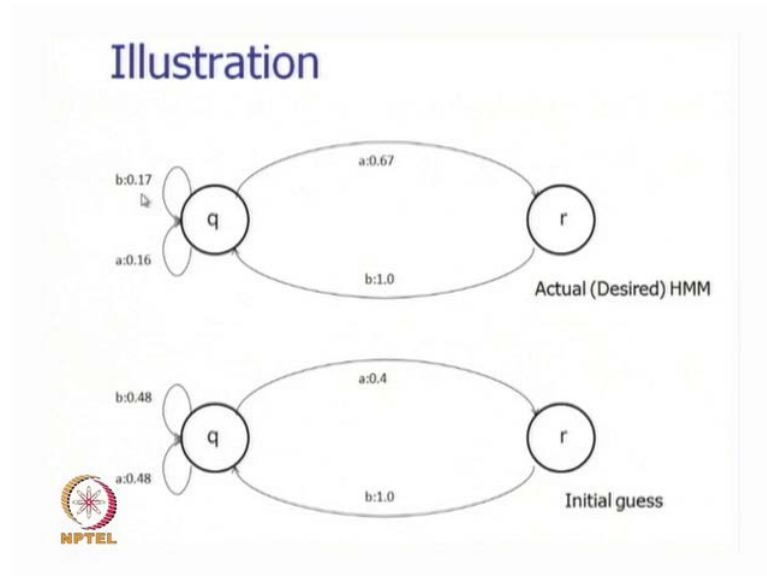
(Refer Slide Time: 22:43)



So, the Baum Welch algorithm, Baum Welch learns the probability values on arcs, not the structure of the H M M, this is important to remember, this is H M M, so we never learnt the structure of the machine, we learnt the probability values on the arcs. So, this is an important point in whole of machine learning, it is very rare, that we learn the structure of an automaton or the contestry rules of a probabilistic, contestry grammar.

What, we learnt are the probability values corresponding to the elements of the machine and the structure is a priori given, learning thus the structure. And then learning the probability that is known to be a difficult problem, that is seem to be difficult problem the structure learning does not happen in such cases, typically we learn the probability values.

(Refer Slide Time: 24:03)



And the way, we learn this is being shown here, so then it is clear that, we are given this machine, the transitions are given a b a b and so on. For example, we know that, from  $r$  a transition is only possible to  $q$  and that to only with symbol  $b$  alright. But, we do not know the exact probability values, this is our problem, we need to learn this. So, initially we start with a guess, for the probability values, so we see that from  $r$  to  $q$  only one transition is possible, which is on  $b$  and we give it a value of 1.0. We know nothing about the transition probabilities on  $b$  and  $a$  here and this  $a$  here, so the probability values are assumed to be 0.48 and this should be actually 0.04 only, that will make the total probability value equal to 1. This is 0.0.04, I would like to correct it sometime and these probability values are the initial estimates.




(Refer Slide Time: 26:25)

### Interplay Between Two Equations

$$P(s^i \xrightarrow{w_k} s^j) = \frac{c(s^i \xrightarrow{w_k} s^j)}{\sum_{l=0}^T \sum_{m=0}^A c(s^i \xrightarrow{w_m} s^l)}$$

$$C(s^i \xrightarrow{w_k} s^j) = \sum_{s_{0,n+1}} P(S_{0,n+1} | W_{0,n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0,n+1}, W_{0,n})$$

No. of times the transitions  $s^i \xrightarrow{w_k} s^j$  occurs in the string




So, this gives us the first factor of this expression  $P(S_{0,n+1} | W_{0,n})$   $n$  is the count.

(Refer Slide Time: 26:39)

### One run of Baum-Welch algorithm: *string ababa*

$\epsilon \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \epsilon$	P(path)	$q \rightarrow a \rightarrow r$	$r \rightarrow b \rightarrow q$	$q \rightarrow a \rightarrow q$	$q \rightarrow b \rightarrow q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Rounded Total $\rightarrow$						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) $\rightarrow$							0.06 <small>= (0.01 / (0.01 + 0.06 + 0.095))</small>	1.0	0.36	0.581
State sequences										

\*  $\epsilon$  is considered as starting and ending symbol of the input sequence string. Through multiple iterations the probability values will converge.



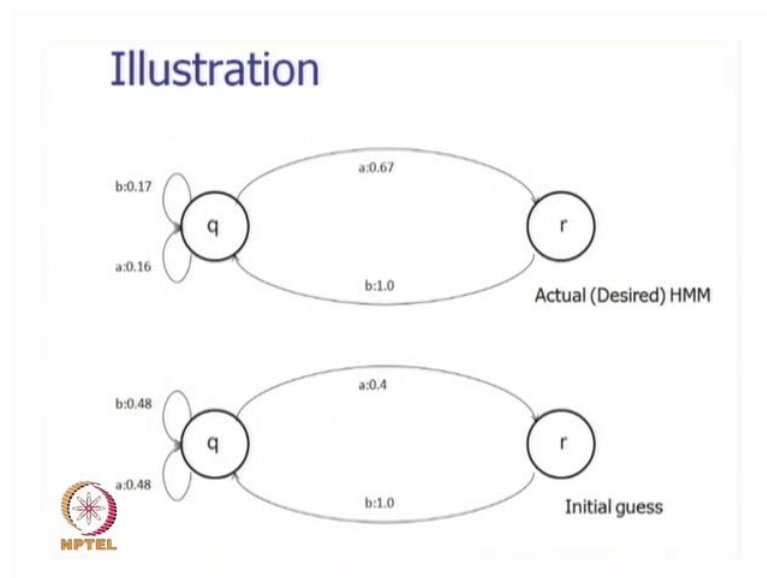
Now, we can see that, the number of times, this transition happens  $q \rightarrow r$  on  $a$ , so  $q \rightarrow r$  on  $a$  happens here also and no more, so there are 2 such appearances of  $q \rightarrow r$  on  $a$ . So, we have to multiply the probability of the path given observation sequence, which is 0.00077 by 2 and get a new probability of  $q \rightarrow r$  on  $a$  as in this state sequence with respect to this state sequence and this observation sequence as 0.00154.

We have to do get this expected count, through the computation of all the states. So, 4 such state sequences are possible, here the probability value is 0.00442, the probability of the path and q to r on a appears only once. So, this is multiplied by 1 and we have 0.00442, in this case also, we have the path probability as 0.00442 and q to r on a appears only once. So, multiplied by 1.00442 the probability of this path sequence is 0.2548 q to r on a does not appear here. So, this is 0 therefore, path probability into count gives me 0.

I have to sum all these probabilities and having taken the sum and round and having rounded it to 2 places of decimal, we obtain the new probability value as 0.01 or the new count expected count then expected count of q on r q to r on a as 0.01. Similarly, we can find out the expected count of r to q on b from here 0.01, then q to q on a as 0.06 q to q on b as 0.095. So, these expected counts of various patterns have found out corresponding to this observation sequence.

So, next what we have to do is that, we have to normalise these counts to get the probability values as given in the formula to get the new probability values. So, this is nothing but q to r on a and then q to q on a and q to q on b, so these counts are taken and put in the denominator, it is a 0.01 divided by 0.01, which is itself 0.06, which is nothing but this and 0.095, which is here. So, when we divide 0.01 by these normalising counts, we get the probability value as 0.06.

(Refer Slide Time: 29:59)






So, we started with the initial probability  $q$  to  $r$  on a  $q$  to  $r$  on a as 0.48, we have to reach a value of 0.16. So, this has to reduce  $q$  to  $r$  on a sorry,  $q$  to  $r$  on a, we started with 0.04 and we have to reach a value of 0.67.

(Refer Slide Time: 30:18)

One run of Baum-Welch algorithm: *string ababa*

$\epsilon \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \epsilon$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Rounded Total $\rightarrow$						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) $\rightarrow$							0.06 <small><math>= (0.01 / (0.01 + 0.06 + 0.095))</math></small>	1.0	0.36	0.581
State sequences										

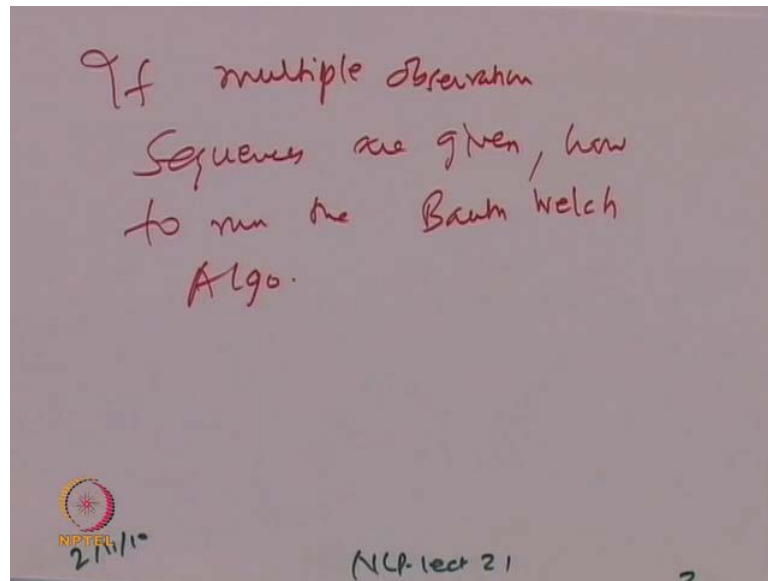
\*  is considered as starting and ending symbol of the input sequence string. Through multiple iterations the probability values will converge.

So, it has to increase and we find that, it has increased, it was 0.04, now it has become 0.06, the probability of  $r$  to  $q$  on  $b$ , there is no other transition from  $r$  and therefore, it has to be divided by 0.01, itself and we get the value 1.  $Q$  to  $q$ , similarly is normalised and we get the new the probability as 0.36 and the new probability for  $q$  to  $q$  on  $b$  0.581.

So, this is the new probability value, after we have seen, the observation sequence  $a b a b b$  and if this was the only observation sequence, then after the first iteration, these are the new probability values. And from these new probability values, we will again be able to compute the new path probabilities, from the new path probabilities, we will get new expected count here and then we will obtain the new normalised values, so which will be new probabilities again.

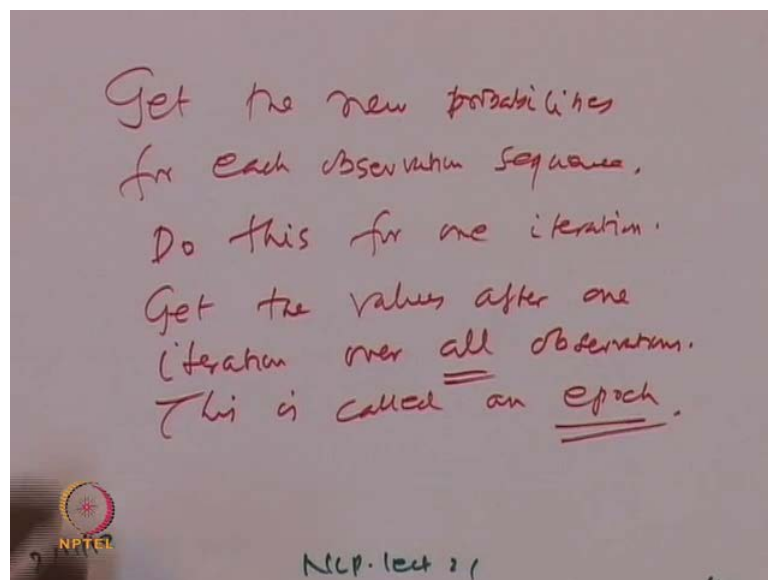
So, the this is the main idea of the algorithm you must have observed, how probability of the sequence, gets recomputed from the new probabilities and these gave give rise to new expected counts, new expected counts give rise to new probabilities. So, what has been illustrated is just one iteration over one observation, the question that arises now, which I would like to write is.

(Refer Slide Time: 32:05)



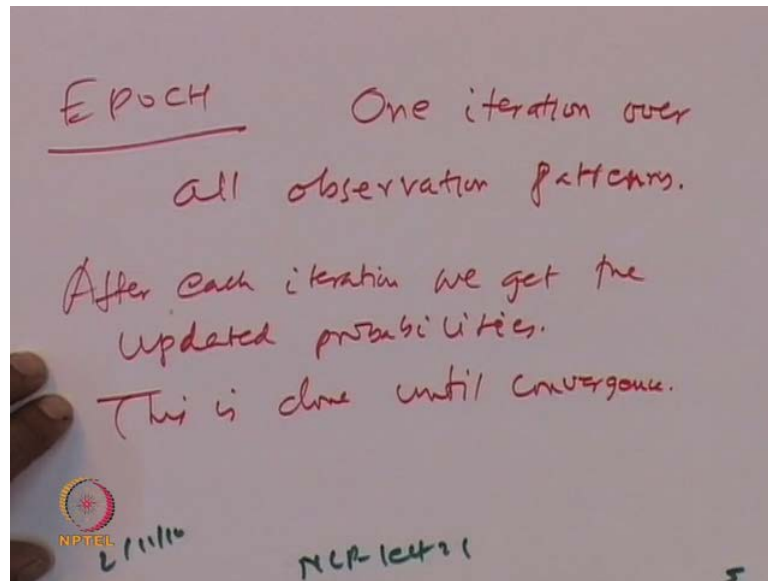
If multiple observation sequences are given, how to run the Baum Welch algorithm, so what we do is that, we find the values of new probabilities over each observation sequence.

(Refer Slide Time: 32:39)



Get the new probabilities, for each observation sequence do this for one iteration get the values, after one iteration over all observations, this is called an epoch. So, let's write this definition here.

(Refer Slide Time: 33:23)



Epoch is nothing but one iteration over all observation patterns, one iteration over epoch is nothing but the what one iteration over all observation patterns. So, after each iteration, we get the updated probabilities, this is done until convergence.

(Refer Slide Time: 34:03)

One run of Baum-Welch algorithm: *string ababa*

$\epsilon \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \epsilon$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.0044	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.0044	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Rounded Total $\rightarrow$						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) $\rightarrow$							0.06	1.0	0.36	0.581
State sequences							$\frac{0.06}{0.01+0.06+0.095}$			


\* considered as starting and ending symbol of the input sequence string. Through multiple iterations the probability values will converge.

So, I hope this gives you a correct idea about, how the computation proceeds, now lets understand the theory of the Baum Welch algorithm, the procedure was illustrated by this example, lets understand the theory.

(Refer Slide Time: 34:20)

### Computational part (1/2)

$$\begin{aligned}
 C(s^i \xrightarrow{w_k} s^j) &= \sum_{t_0, n+1} [P(S_{0, n+1} | W_{0, n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t_0, n+1} [P(S_{0, n+1}, W_{0, n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t=0, n} \sum_{t_0, n+1} [P(S_t = s^i, W_t = w_k, S_{t+1} = s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t=0, n} [P(S_t = s^i, W_t = w_k, S_{t+1} = s^j, W_{0, n})]
 \end{aligned}$$



$$\begin{matrix}
 w_0 & w_1 & w_2 & \dots & w_t & \dots & w_{n-1} & w_n \\
 \rightarrow & S_1 & S_1 & \dots & S_t & S_j & \dots & S_{n-1} & S_n & S_{n+1}
 \end{matrix}$$

This is a computational part and we will understand the mathematical expressions involved in this, so term of interest  $C(S_i \rightarrow S_j \text{ on } W_k)$ ,  $S_i$  is the source state,  $S_j$  is the destination state and  $W_k$  is the observed symbol. We are supposed to get the count of this pattern  $S_i$  to  $S_j$  on  $W_k$ , in the context of the state sequence and the observation sequence. So, this is a number, we have to multiply it or weigh it by the probability of the state sequence given the observation sequence as has been understood.

And this task has to be done for all possible states that results from, the observation sequence. Now, this part  $P(S \text{ given } o)$  can be converted into  $P(S, W)$ , can be converted to  $P(S, W)$ , it is a joint probability and we have to divide it by probability of  $W$ , the observation sequence. And this  $W$  does not depend on  $S$  and there is taken out of the summation symbol.

Now, we have the expression where, we have the state sequence observation sequence, state sequence observation sequence and the pattern that obtains. Now, one can see that, this whole number of the patterns can be converted into, an equivalent expression of the probability of the  $t$ th state being  $S_i$   $t+1$ th being  $S_j$  and the  $t$ th observation symbol being  $W_k$ .

So, this whole pattern  $S_i$  to  $S_j$  on  $W_k$  is equivalent to  $S_t$  being  $S_i$ , the  $t$ th state being  $S_i$ ,  $t+1$  state being  $S_j$  and the  $t$ th observation being  $W_k$  and this happens in the context of the state sequence  $S$  and observation sequence  $W$ . So, previously itself, there


was this summation over all possible state sequences. Now, we introduce the summation over all possible  $t$  where,  $t$  is varying from 0 to  $n$ , which is nothing but a position,  $t$  records a position in the output, in the observation sequence.

So, you can see that, this number this expression with the count with the number of the pattern is equivalent to this expression where, the pattern is searched for at every location of the output sequence or the observed sequence. This 2 expressions are equivalent, because wherever wait I do not see this pattern, the probability will be 0, otherwise there is some probability, which is equivalent to this expression.

Now, we what we can do is that, since  $S_0$  to  $n+1$  is present here as a joint probability expression and since there is a some overall possible state sequences, this particular variable can be dropped by the law of marginalisation. And having dropped, it we have the pattern have the probability expression here with source state  $S_i$  at  $t$  th location  $S_{t+1}$  state is  $S_j$ , that  $t$  th observation symbol is  $W_k$  and the whole observation sequence is  $W_0$  to  $n$ . And we have a multiplying factor of 1 by  $P(W_0$  to  $n$ ), so this is the expression, we have and this can be pasteurised by means of the observation sequence here,  $W_0$  to  $W_n$  and state sequence  $n_0$  to  $S_{n+1}$ .

(Refer Slide Time: 38:54)

### Computational part (2/2)

$$\begin{aligned}
 & \sum_{t=0}^n P(S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{0:n}) \\
 &= \sum_{t=0}^n P(W_{0:t-1}, S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{t+1:n}) \\
 &= \sum_{t=0}^n P(W_{0:t-1}, S_t = s^i) P(S_{t+1} = s^j, W_t = w_k | W_{0:t-1}, S_t = s^i) P(W_{t+1:n} | S_{t+1} = s^j) \\
 &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\
 &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\
 &= \sum_{t=0}^n F(t-1, i) P(s^i \xrightarrow{w_k} s^j) B(t+1, j)
 \end{aligned}$$


$$\begin{matrix}
 w_0 & w_1 & w_2 & \dots & w_{t-1} & w_t & w_{t+1} & w_n \\
 \rightarrow & S_1 & S_1 & \rightarrow \dots & S_t & S_j \dots & S_{n-1} & S_n \rightarrow S_{n+1}
 \end{matrix}$$

Now, we look at the expression  $P(S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_0$  to  $n$ ). So, here again, we can make use of isolating parts of the string, that are useful to us and applying marginalisation chain rule Markov assumption etcetera etcetera

to get a useful expression. So, the way we work is that, we isolate the part 0 to  $t - 1$  as the observed sequence and with the state  $S_t$  being  $S_i$ .

So, this part is isolated, so that we can have a forward probability up to  $t$ th location and then  $S_{t+1}$  being  $S_j$  and the observation sequence from  $W_{t+1}$  to  $n$ , this will be useful to us, in terms of the backward probability and  $W_t$  is equal to  $W_k$ . We have also done, 1 thing that, we have isolated this pattern here,  $S_t$  equal to  $S_i$ ,  $S_{t+1}$   $S_j$  and  $W_t$  equal to  $W_k$ , which should give us something like a transition probability.


Now, we apply chain rule and we see this expression here, the summation remains  $t$  equal to 0 to  $n$ , so  $P(W_{0:t-1} | S_t = S_i)$  is isolated, this will give us the forward probability, when I isolate this, I have to apply the chain rule and say that  $P(S_{t+1} = S_j | W_t = W_k \text{ and } W_{0:t-1} | S_t = S_i)$ . So, this is nothing but a destination state observation symbol and source state and this is a string up to this particular observation symbol and can be ignored, because of the Markov assumption, this part will give us the transition probability.

What remains is this part  $W_{t+1:n}$ , which is the observation sequence, starting from  $t+1$ th location to the end of the string and this again by chain rule has to be conditioned by  $S_{t+1}$ ,  $S_{t+1} = S_j$ ,  $W_t = W_k$ ,  $S_t = S_i$  and  $W_{0:t-1}$  all these will come as conditioning variable for  $W_{t+1:n}$ . And we know that,  $W_{t+1:n}$ , which is the observation sequence from  $t+1$ th location to the end of the string is dependant only on the state, which existed before this observation sequence.

And that is nothing but the  $t+1$ th state as  $S_j$ , so in the conditioning part, we retain only this, so from this expression, we see the first part gives us the forward probability  $F_{t-1,i}$ , the second part gives us the transition probability  $P(S_i \rightarrow S_j | W_k)$ . And the remaining part gives me the backward  $B_{t+1,j}$ . So, this is the forward probability up to the  $t$ th location in the string, this is backward probability from  $t+1$ th location to the end of the string and in between is the transition probability  $P(S_i \rightarrow S_j | W_k)$ . So, this is a neat expression, which is obtained the expected count of a particular transition, so the expected count.

(Refer Slide Time: 42:39)

### Computational part (1/2)


$$\begin{aligned}
 C(s^i \xrightarrow{w_k} s^j) &= \sum_{t_0, n+1} [P(S_{0, n+1} | W_{0, n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t_0, n+1} [P(S_{0, n+1}, W_{0, n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t=0, n} \sum_{t_0, n+1} [P(S_t = s^i, W_t = w_k, S_{t+1} = s^j, S_{0, n+1}, W_{0, n})] \\
 &= \frac{1}{P(W_{0, n})} \sum_{t=0, n} [P(S_t = s^i, W_t = w_k, S_{t+1} = s^j, W_{0, n})]
 \end{aligned}$$


$w_0 \rightarrow S_1 \xrightarrow{w_1} S_1 \xrightarrow{w_2} \dots S_i \xrightarrow{w_k} S_j \dots \xrightarrow{w_{n-1}} S_{n-1} \xrightarrow{w_n} S_n \rightarrow S_{n+1}$

If you remember is C S i to S j with W k.

(Refer Slide Time: 42:43)

### Computational part (2/2)

$$\begin{aligned}
 &\sum_{t=0}^n P(S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{0, n}) \\
 &= \sum_{t=0}^n P(W_{0, t-1}, S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{t+1, n}) \\
 &= \sum_{t=0}^n P(W_{0, t-1}, S_t = s^i) P(S_{t+1} = s^j, W_t = w_k | W_{0, t-1}, S_t = s^i) P(W_{t+1, n} | S_{t+1} = s^j) \\
 &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\
 &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\
 &= \sum_{t=0}^n F(t-1, i) P(s^i \xrightarrow{w_k} s^j) B(t+1, j)
 \end{aligned}$$


$w_0 \rightarrow S_1 \xrightarrow{w_1} S_1 \xrightarrow{w_2} \dots S_i \xrightarrow{w_k} S_j \dots \xrightarrow{w_{n-1}} S_{n-1} \xrightarrow{w_n} S_n \rightarrow S_{n+1}$

So, in the context of the observation sequence and the state sequence, we have the expression as expected count is nothing but t going over the whole string and we have to sum these quantities. And F t minus 1 comma i, this is the forward probability, the transition probability and the backward probability, the whole thing of course, has to be divided by the probability of the observation sequence. Now, probability of the observation sequence also can be expressed, as the product of forward probability and

backward probability with t bearing over the whole string. So, you should try to show this, I right down the expression here. So, it can be shown.

(Refer Slide Time: 43:28)

$$P(W_{0,n}) = \sum_{t=0}^n F(t-1, i) \cdot B(t+1, j)$$

$S^i \xrightarrow{w_k} S^j$

NPTEL 2/11/10 NLP-lect 2 r 6

That the probability of the observation sequence 0 to n is nothing but t equal to 0 to n, F t minus 1, i into B t plus 1 j and the picture is S i to S j on W k, so this means that, we can completely work with backward and forward probabilities. And these backward and forward probabilities can be computed in time proportional to the length of the sequence in question.

(Refer Slide Time: 44:19)

### Computational part (2/2)

$$\begin{aligned} & \sum_{t=0}^n P(S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{0,n}) \\ &= \sum_{t=0}^n P(W_{0,t-1}, S_t = s^i, S_{t+1} = s^j, W_t = w_k, W_{t+1,n}) \\ &= \sum_{t=0}^n P(W_{0,t-1}, S_t = s^i) P(S_{t+1} = s^j, W_t = w_k | W_{0,t-1}, S_t = s^i) P(W_{t+1,n} | S_{t+1} = s^j) \\ &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\ &= \sum_{t=0}^n F(t-1, i) P(S_{t+1} = s^j, W_t = w_k | S_t = s^i) B(t+1, j) \\ &= \sum_{t=0}^n F(t-1, i) \mathbb{P}(s^i \xrightarrow{w_k} s^j) B(t+1, j) \end{aligned}$$

NPTEL  $0 \rightarrow S_1 \xrightarrow{w_0} S_1 \xrightarrow{w_1} \dots S_t \xrightarrow{w_t} S_j \dots \rightarrow S_{n-1} \xrightarrow{w_{n-1}} S_n \xrightarrow{w_n} S_{n+1}$




So, this is a neat expression where, we get the probability value where, we get the expected count from this formula of course, dividing it by the probability of the observation sequence and from that count, we can obtain, the new probability values.

(Refer Slide Time: 44:36)

### Interplay Between Two Equations

$$P(s^i \xrightarrow{w_k} s^j) = \frac{c(s^i \xrightarrow{w_k} s^j)}{\sum_{l=0}^T \sum_{m=0}^A c(s^i \xrightarrow{w_m} s^j)}$$

$$C(s^i \xrightarrow{w_k} s^j) = \sum_{s_{0,n+1}} P(S_{0,n+1} | W_{0,n}) \times n(s^i \xrightarrow{w_k} s^j, S_{0,n+1}, W_{0,n})$$



$w_k$   
 No. of times the transitions  $s^i \rightarrow s^j$  occurs in the string


Then the probability values can be used, in the forward backward formula to obtain the new count, this will give rise to new probability value and this will go on. So, as this goes on, we finally, come to a situation where, the probability value does not appreciably change and that would be the termination of the algorithm. So, there is some interesting points about this algorithm.

(Refer Slide Time: 45:06)

## Discussions

- Symmetry breaking:**  
Example: Symmetry breaking leads to no change in initial values  

The 'Desired' diagram shows three states (S) in a line. Transitions are: S1 to S2 (a:0.5), S2 to S1 (b:1.0), S2 to S3 (a:1.0), and S3 to S2 (b:0.5). The 'Initialized' diagram shows the same three states. Transitions are: S1 to S2 (a:0.5), S2 to S1 (b:0.25), S2 to S3 (a:0.25), S3 to S2 (b:0.5), S3 to S1 (a:0.25), and S1 to S3 (b:0.5).
- Struck in Local maxima
- Label bias problem  
Probabilities have to sum to 1.  
Values can rise at the cost of fall of values for others.



Some of these issues pertain to, the behaviour of the algorithm when it, it gets stuck in a local maxima for example, so in this particular algorithm, the computation stops when there is no appreciable change in the probability values. Now, this particular state of probability values can be a case of local maximum, given this particular initialisation, we could, we come to a sort of dead end, I would say because the probability values do not change any more. But it is very possible these probability values are not the desired probability values, it is a case of a local maximum or local minimum however, way you have to you want to look at it.

And this can be changed only by reinitialising the machine and starting the computation all over again, so this phenomenon is the phenomenon of getting stuck in the local minimum, looking at the slide once again, we see there are 2 other points, which can sort of handicap the algorithm, 1 is the issue of symmetry breaking. So, the symmetry breaking issue is that, if we do not randomly initialise the weights, but a kind of symmetry exists, which is shown here, you can see that, a value b value and a value b value here.

So, forward and backward transitions with a and b values, they have equal starting probabilities. Now, if the incrementation happens with equal amount, then these values will never be different and if the target machine has different probability values, it will not be possible to reach them.

And the final problem is the label bias problem, which we will discuss later, if one of the probability values dominates, and then it can lead to problems in the computation. So, this is the Baum Welch algorithm, which is an important algorithm in this area, a probabilistic models hidden Markov model in particular and this is a very useful development.