

Natural Language Processing
Prof. Pushpak Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

Lecture - 20
HMM, Forward and Backward Algorithms, Baum Welch Algorithm

Today, we are going to discuss two important concepts with respect to hidden Markov model, the forward algorithm, the backward algorithm. The forward and backward probability values and their application in hidden Markov model training, this algorithm is also known as the Baum Welch algorithm or forward backward algorithm. Now, the training part is important, because the strings which are accepted by the hidden Markov model and we will discuss it in the context of bigram probabilities.

The states are bigram sequences, we will take two sequence states and with respect to that we find out how the parameters of the hidden Markov model can be found. We remember that, hidden Markov model is defined by its transition probability and the observation probability. The idea is to be able to learn these values, there is a very famous technique called the expectation maximization, which is at the heart of this algorithm and we will discuss this in detail.

(Refer Slide Time: 01:36)

Forward and Backward
Probability Calculation



So, continuing with the slides, forward and backward probability calculation is what we are going to do, and we make some very useful definitions.

(Refer Slide Time: 01:48)

Forward probability $F(k, i)$

- Define $F(k, i)$ = Probability of being in state S_i having seen $o_0 o_1 o_2 \dots o_k$
- $F(k, i) = P(o_0 o_1 o_2 \dots o_k, S_i)$
- With m as the length of the observed sequence
- $P(\text{observed sequence}) = P(o_0 o_1 o_2 \dots o_m)$
 $= \sum_{p=0, N} P(o_0 o_1 o_2 \dots o_m, S_p)$
 $= \sum_{p=0, N} F(m, p)$



We define $F(k, i)$ as the probability of being in state S_i having seen the observation sequence $o_0 o_1 o_2 \dots o_k$. So, this sequence is seen, this observed sequence and having seen that the system is in the state S_i . So, $F(k, i)$ can be written as probability of the string seen being $o_0 o_1 \dots o_k$ and the state after that being S_i , it is a joint probability; the probability of two events happening. With m as the length of the observation sequence, the probability of the observed sequence is the whole sequence is $P(o_0 o_1 o_2 \dots o_m)$.

And now, imply marginalization and this probability can be written as equal to probability $o_0 o_1 o_2 \dots o_m, S_p$. S_p is any state which comes after seeing the whole sequence of observations and we have to make this random variable, take all possible values. And so p is going from 0 to N , there are total number of $N + 1$ states and this equal to $\sum_{p=0, N} F(m, p)$. So, we apply straight forward, the definition of forward probability and $F(m, p)$ is the representation of P whole sequence of observations and the next state.

So, if we sum the forward probabilities of the observation sequence and the state the machine is in after the observation sequence then we will get the probability of the whole sequence. So, this is simply applying the law of probability, the marginalization law, and so we find that, the probability of the observation sequence is equal to \sum_p

going from 0 to N, F m p. So, this is the way, the forward probability is used to calculate the probability of the whole observed sequence.

(Refer Slide Time: 04:27)

Forward probability (contd.)

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

Now, the question that comes naturally is, how do we calculate this forward probability, the final forward probability is that of the whole observed sequence. Can we do it efficiently, can we sort of divide and conquer, can we build the whole computation from small intermediate computations on segments of the observed sequence. So, the picture shown here, clarifies lot of issues regarding the situation, so S_0 starting state, S_1 is the next state, which can be one of the states, which are the constituents of the hidden Markov model.

ϵ is typically epsilon where, we go into one of the starting states or if there is a unique starting state, we go there. Then $o_1 o_2 o_3 \dots o_k o_{k+1} \dots o_{m-1} o_m$ is the whole observed sequence and after o_m , the system enters a final state which can be one of the given states in the machine. So, when we do so we find that we are transiting from state to state with a output observation o_k . So, S_p to S_q , there is a state transition and o_k is the observed output symbol.

Now, this is a diagram which is capturing the relation between the observed sequence and the state sequence. So, for every state, there is an output symbol and a transition to the next state. So, when we have this picture in front of us, we can very easily see that, the forward probability $F(k, q)$ which means nothing but observing the sequence o_0

$O_1 O_2 \dots O_k$. And then being in state S_q , this is by definition and this is equal to $P(O_0 O_1 O_2 \dots O_{k-1}, I)$ isolate the symbol O_k , the last symbol on the output sequence and I also have this S_q .

So, the whole probability expression now has three parts, O_0 to O_{k-1} , the symbol O_k and then the state S_q . Now, I introduce the state S_p , so it is helpful to look at the picture, when we look at this expression. So, what was $F_{k,q}$, $F_{k,q}$ is referring to this part of the output sequence and this part of the state sequence. So, one sees this whole output sequence and comes into the state S_q . Now, when we have S_p then surely the output symbol is the previous one, which is O_{k-1} and S_p is the previous symbol.

Now, since we are introducing this variable here S_p , p has to again go from 0 to N , all possible states of the hidden Markov model then therefore, a sigma sign is introduced. This is the law of marginalization and this shows that, we have introduced S_p here, O_0 to O_{k-1} is the observation sequence, O_k is the symbol and, S_p and S_q are the states. So, one should particularly note this pattern $S_p O_k S_q$ where, S_p is the source state, S_q is the destination state and O_k is the output symbol.

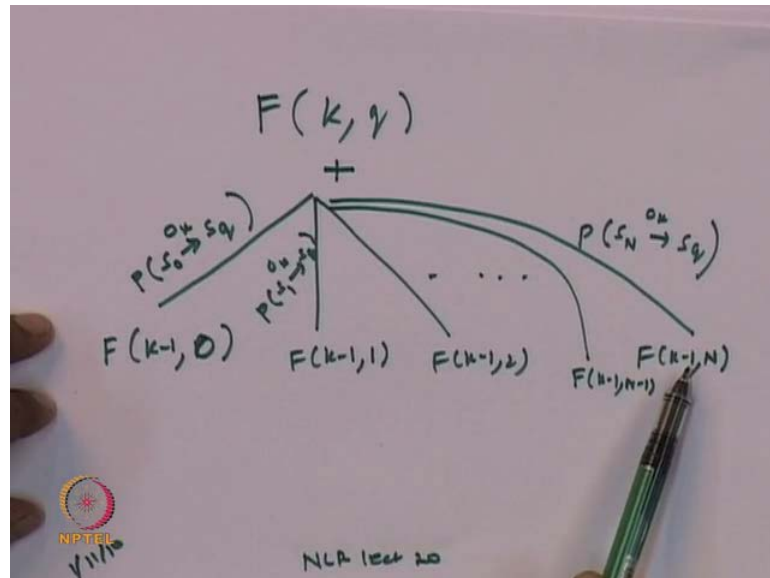
So, we are essentially referring to the pattern that obtains at the state and observation sequence as shown here, so this will be our pattern of interest. Now, what we can do is that, we can take this probability value and isolate the part $P(O_0 O_1 O_2 \dots O_{k-1}, S_p)$ here then we have symbols upto O . Then we have this symbol here, S_p and this should be O_k , so $P(O_k | S_p)$ and the condition part is $S_p O_0 O_1 O_2 \dots O_{k-1}$, so this is actually O_k .

So now, we have this part which is nothing $F_{k-1,p}$, so the probability of seeing O_0 to O_{k-1} and being in state S_p , so this $F_{k-1,p}$. And this particular probability here, we can ignore everything other than S_p , because the previous observation symbols do not influence this probability according to Markov assumption. So, $P(O_k | S_q)$ given S_p and this by definition is nothing but $P(S_p | O_k) \rightarrow S_q$.

So, this is nothing but this transition probability of going from state S_p to S_q as shown here in this whole sequence and $F_{k-1,p}$ is the forward probability up to $k-1$. So, we have this way of computing $F_{k,q}$ from $F_{k-1,p}$

and this transition. So, we can depict this forward probability calculation pictorially as follows, I would write this diagram.

(Refer Slide Time: 10:44)



So, $F(k, q)$ is nothing but a summation of $F(k-1, p)$ values, which is nothing but 0 then $F(k-1, 1)$, $F(k-1, 2)$ and so on $F(k-1, N)$ and let us also throw in one more term, which is $F(k-1, N-1)$. So, we take these forward probabilities, multiply them by the corresponding transition probability values, which is nothing but $F(p, s_0 \rightarrow s_q)$ with output symbol being O_k . And this probability here would be $S_1 \rightarrow S_q$ with output symbol being O_k and here, this probability will be $S_N \rightarrow S_q$ with output probability being O_k .

So, this very clearly shows, how the forward probability at any point in the sequence at the k th place can be computed in terms of, whatever probability value, whatever forward probability value has been accumulated so far upto $k-1$. We simply take those forward probability values for different states at $k-1$ and multiply each such forward probability with the transition probability involved and then sum them up, we get the forward probability. So, this means that, the computation of these $F(k, q)$ can be done by performing an addition of $N+1$ previous computations multiplied by a probability value. So, we have $N+1$ operations giving rise to the computation of $F(k, q)$.

(Refer Slide Time: 13:43)

Complexity expression for
Computing $F(k, q)$

$$T_k = \sum_{i=0}^N T_{k-1}$$

Recursive expression

NPTEL
NLP-lect 20

So, if we can write a complexity expression in the next slide, complexity expression for computing $F(k, q)$, let us call this T_k , T_k is clearly equal to i is equal to 0 to N , T_{k-1} , so this in terms of the recursive expression.

(Refer Slide Time: 14:45)

Boundary condition

$$F(0, q) = P_q$$

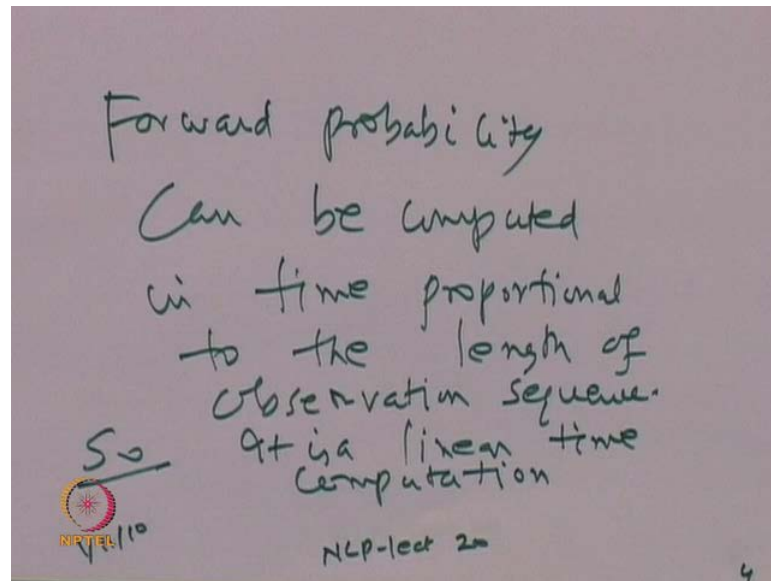
where P_q is the initial
probability of being in
state S_q

$S_0 \rightarrow S_q$

NPTEL
NLP-lect 20

And we can write the boundary condition as $F(0, q)$ is equal to P_q where, P_q is the initial probability of being in state S_q that is, S_0 to S_q . So, this shows us that, the forward probability can be computed very easily.

(Refer Slide Time: 15:29)



And it is not difficult to show that, forward probability can be computed in time proportional to the length of observation sequence, so it is a linear time computation. So, this expression we can take up in more detail later, in particular what we will emerge is that, the forward probability or the probability of the observation sequence, which in terms of the output, in terms of the forward probabilities. This forms a lattice like structure, so this what we would like to show ((Refer Time: 16:27)). Now, we will move on to the complimentary probability, which is called the backward probability.

(Refer Slide Time: 16:33)

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
- $P(\text{observed sequence}) = P(O_0 O_1 O_2 \dots O_m)$
 $= P(O_0 O_1 O_2 \dots O_m | S_0)$
 $= B(0,0)$

NPTTEL

Backward probability, we call it $B(k, i)$ and look at the slightly different definition of $B(k, i)$, $B(k, i)$ is the probability of seeing $O_k O_{k+1} O_{k+2} \dots O_m$ given that, the state was S_i . So, just compare this with the definition of the forward probability, forward probability says that, we begin from the start of the observation sequence and go to the k th observation symbol and then we note the state we are in, from there. In backward probability, we start from the k th observation symbol, go on upto the end of the string and this is the observed sequence given that, the state was S_i .

So, $B(k, i)$ can be written now as probability of seeing $O_k O_{k+1} O_{k+2} \dots O_m$ that is, the end of the sequence given that, the state before outputting O_k was S_i with m as the length of the observed sequence. So, $P(\text{observed sequence})$ therefore, can surely be written as $\sum_{i=0}^N P(O_1 O_2 \dots O_m | S_i)$, which is equal to $\sum_{i=0}^N P(O_1 O_2 \dots O_m | S_i)$ given that, the state was S_i , which is always true and that can be written simply as $B(0, 0)$.

So, we take the probability of observing O_1 to O_m given that, the starting state was S_0 , which simply is $B(0, 0)$. So now, you can compare that $P(\text{observed sequence})$ could be expressed as a sum of forward probabilities and probability of the same observed sequence is nothing but a single backward probability with arguments $(0, 0)$.

(Refer Slide Time: 19:02)

Backward probability (contd.)

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

O_1
 O_2
 $O_3 \dots$
 O_k
 O_{k+1}
 \dots
 O_{m-1}
 O_m

S_0
 S_1
 S_2
 S_3
 \dots
 S_p
 \rightarrow
 S_q
 \dots
 S_m
 S_{final}

Now, we try to see again just like forward probability, how the backward probability can be calculated and again this state sequence is, which was also used for forward probability calculation. So, we have the observation sequence $O_0 O_1 O_2 O_3$ upto O_m , which is the last observation symbol and $S_0 S_1 S_2 S_3$, etcetera are the state symbol, so with S_{final} as the final state. Now, each of these $S_1 S_2 S_3$, etcetera can be one of the constituents states of the hidden Markov model.

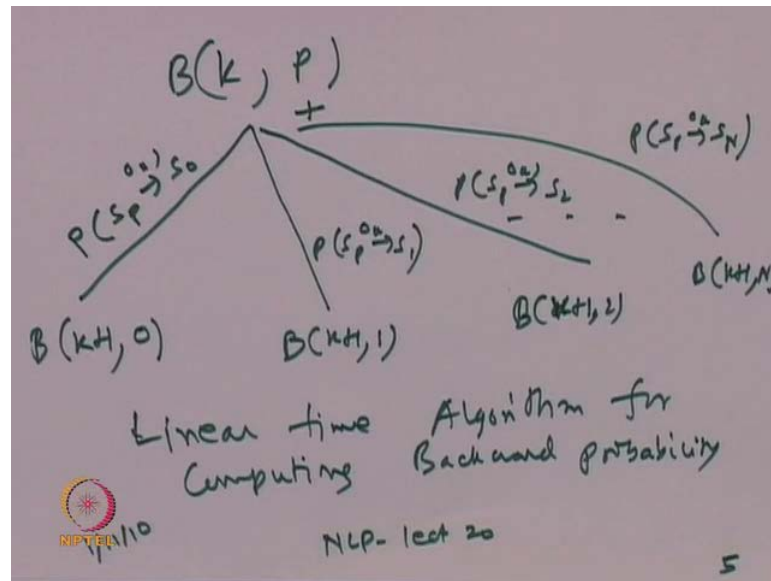
So, we take up these expression $P(O_k O_{k+1} \dots O_m | S_p)$, now this again can be broken up into two parts. We isolate the part O_k and leave this part O_{k+1} to O_m and the conditioning part remain same, which is S_p . And now, we introduce a margin variable, which is S_q and S_q you can see here, is the state at this point in the state sequence after O_k . So, this is the picture and we correspond this with the picture here.

Now, this is $\sum_{q=0}^N P(O_k | S_q, S_p)$, so this part I isolate, this particular pattern I isolate $O_k | S_q$ given S_p . So, we are doing nothing but trying to capture the probability of this pattern, which obtains at the output and observation sequence. So, $P(O_k | S_q)$ given S_p and then we have the other probability part $P(O_{k+1} O_{k+2} \dots O_m)$ and now, I apply chain rule, so O_k and S_q come as the conditioning part and S_p is of course, there.

So, this particular part $P(O_k | S_q)$ given S_p is written here and our part here, $P(O_{k+1} O_{k+2} \dots O_m | S_q, S_p)$ can simply be written as $P(O_{k+1} O_{k+2} \dots O_m | S_q)$. So, this becomes the probability expression, because this observation sequence is not influenced by anything, other than the state from which it started, which is S_q . So, look at the picture here, $O_{k+1} O_{k+2} \dots O_m$, the state before O_{k+1} was S_q and this is the only thing, which influences this whole observation sequence, this is by virtue of the Markov assumption.

So, this is what we have here and $P(O_k | S_q)$ given S_p is nothing but the probability of this transition S_p to S_q with the output symbol O_k . So therefore, we can write $B_{k,p}$ as equal to $B_{k+1,q}$, $P(S_p \rightarrow S_q | O_k)$ and q going from 0 to N on the sum here. So, from this again, we can write a small expression for the calculation of the backward probability $B_{k,p}$.

(Refer Slide Time: 22:53)



So, when we write on the paper $B(k, p)$ can be seen to be simply implemented by $B(k+1, 0) + B(k+1, 1) + B(k+1, 2) + \dots + B(k+1, N)$. So, this is what is the expression, which is used for computing $B(k, p)$. And we have to sum these probability values after multiplying each of these backward probabilities with the probability of S_p , $B(k, p)$, so probability of S_p to S_0 on $O(k)$, probability of S_p to S_1 on $O(k)$, probability of S_p to S_2 on $O(k)$, probability of S_p to S_N on $O(k)$.

So, we have these probability values, which are computed from a shorter string, because $k+1$ to N is a shorter string compared to k to N , the total length of the string starting from k and we multiply this by the probability values here. So, again the time complexity will be exactly like the forward probability and we have a linear time algorithm for computing backward probability. Thus, we have found two very important probability values, the forward probability and the backward probability.

So, both these probability values can be computed by making use of a recursive expression and the expression is such that, for both very efficient linear time algorithm exists. So, for the whole output sequence, we can get the forward probability in time proportional to length of the output string. Similarly, we can express this probability value in terms of backward probabilities and again, this is a linear time algorithm ((Refer Time: 25:46)).

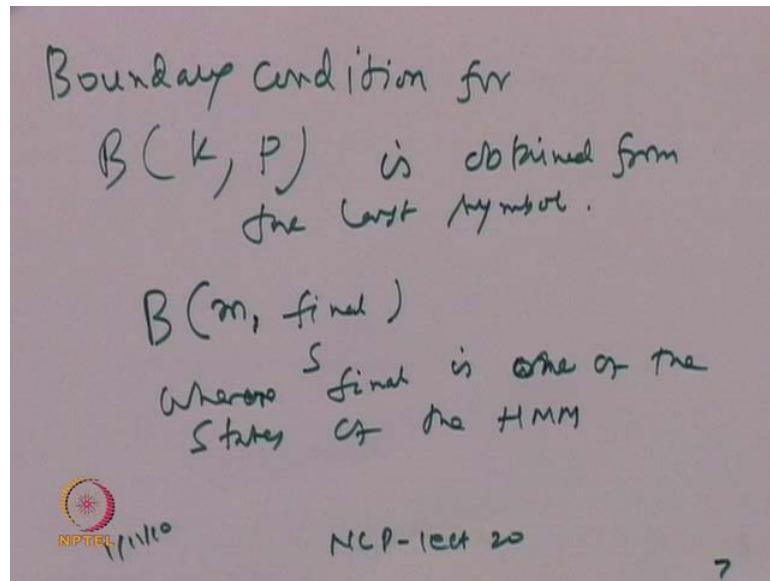
So, let us summarize what we have learned, you can look at the slide here and for any observation sequence and the corresponding state sequence, we have a notion of the k th place and we can compute the forward probability upto any point in the observation sequence. And we can compute the backward probability from any point to the end of the output sequence for any point in the string. So, let me just repeat this point, for any point in the output sequence and the corresponding state sequence, we can compute the forward probability upto that point and the backward probability from that point upto the end of the string. One point leads completion which is that, the backward probability needs a boundary condition.

(Refer Slide Time: 26:49)

The image shows a handwritten equation on a slide: $B(k, p) = \sum_{q=0}^N B(k+1, q) \cdot P(S_p \xrightarrow{O_k} S_q)$. Below the equation, there is a downward-pointing arrow and the text "goes on increasing towards end of observation sequence". In the bottom left corner, there is an NPTEL logo. In the bottom center, it says "NP-lect 20" and in the bottom right corner, there is a small number "6".

So, $B(k, p)$ we have seen is equal to $\sum_{q=0}^N B(k+1, q) \cdot P(S_p \xrightarrow{O_k} S_q)$ with O_k and this $k+1$ goes on increasing towards end of observation sequence. So, we must have a boundary condition, so we write this boundary condition here.

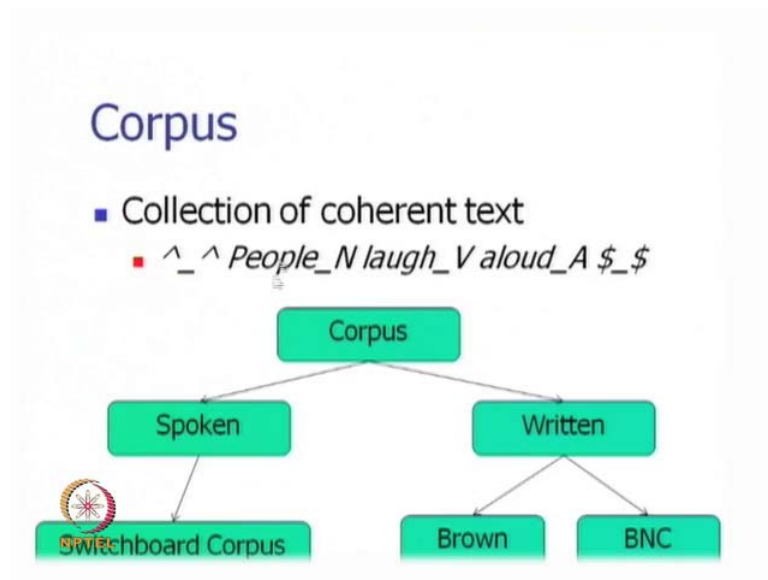
(Refer Slide Time: 27:36)



Boundary condition for $B(k, p)$, so if you look at the slide (Refer Slide Time: 27:50) the boundary condition is very clear. The boundary condition is that, having observed the last symbol in the whole sequence, the system can be said to be in a final state. So, this should give us the boundary condition, so we can say that, from S_m to the final state S_{final} on outputting O_m is a boundary condition, otherwise we can also propose an epsilon transition here and from there, we can say that, there is a final state with respect to one of the states of the hidden Markov model.

So, the probability of going to the final state from the last state S_m is the boundary condition. So, this would be for $B(k, p)$ is obtained from the last symbol, so we essentially have to write $B(m, s_{final})$ where, s_{final} is one of the states of the HMM. So, this gives us a complete picture about the forward and backward probabilities, which are useful for various calculations in hidden Markov model.

(Refer Slide Time: 29:52)



We now proceed to, how the training of the hidden Markov model is done, the problem of part of speech tagging. We have chosen a particular problem here, the problem of part of speech tagging, we have a sequence of words here, people laugh aloud and there is a symbol hat, which starts the sentence and a dollar symbol, which ends the sentence. So, for hat, the tag is hat itself, people is noun, so underscore noun, laugh is a verb, so underscore verb, aloud is an adverb, so underscore adverb.


So, these part of speech symbols are non standard non conventional, we simply use this for the purpose of illustration. So, our problem is to train a hidden Markov model from this kind of training data where, this is a supervise situation. But, it is also possible to have situations where, the output sequences of the machine are given and from there, we have estimate the parameters of the hidden Markov model. So, this is a bit of a degration, where we say that, one example where hidden Markov model is used, is in part of speech tagging.

There is a corpus, corpus can be either spoken or written, under spoken corpus we have many famous ones, the very well known corpus is the switchboard corpus, which records many hours of telephonic conversation. And under written corpora, we have famous things like brown corpus, which has about 1 million words, British national corpus which has about 10 million words with tags and so on.

(Refer Slide Time: 31:48)

Some notable text corpora of English

- [American National Corpus](#)
- [Bank of English](#)
- [British National Corpus](#)
- [Corpus Juris Secundum](#)
- [Corpus of Contemporary American English](#) (COCA)
400+ million words, 1990-present. Freely searchable online.
- [Brown Corpus](#), forming part of the "Brown Family" of corpora, together with [LOB](#), Frown and F-LOB.
- [International Corpus of English](#)
- [Oxford English Corpus](#)
- [Scottish Corpus of Texts & Speech](#)




So, here is a listing of famous text corpora of English, American national corpus, bank of English corpus, British national corpus, corpus Juris Secundum, corpus of contemporary American English, brown corpus international corpus of English, oxford English corpus, Scottish corpus of texts and speech, so these are well known corpora.

(Refer Slide Time: 32:11)

Penn Treebank Tagset: sample

- 1. CC Coordinating conjunction
- 2. CD Cardinal number
- 3. DT Determiner
- 4. EX Existential *there*
- 5. FW Foreign word
- 6. IN Preposition or subordinating conjunction
- 7. JJ Adjective
- 8. JJR Adjective, comparative
- 9. JJS Adjective, superlative
- 10. LS List item marker
- 11. MD Modal
- 12. NN Noun, singular or mass
- 13. NNS Noun, plural 1
- 14. NNP Proper noun, singular



Now, what about the tag set, tag set is a very important concern in any part of speech tagging. We have remarked in one of the previous lectures that, there is no point giving detail and intricate tags. If they cannot after all be assigned from a limited context around

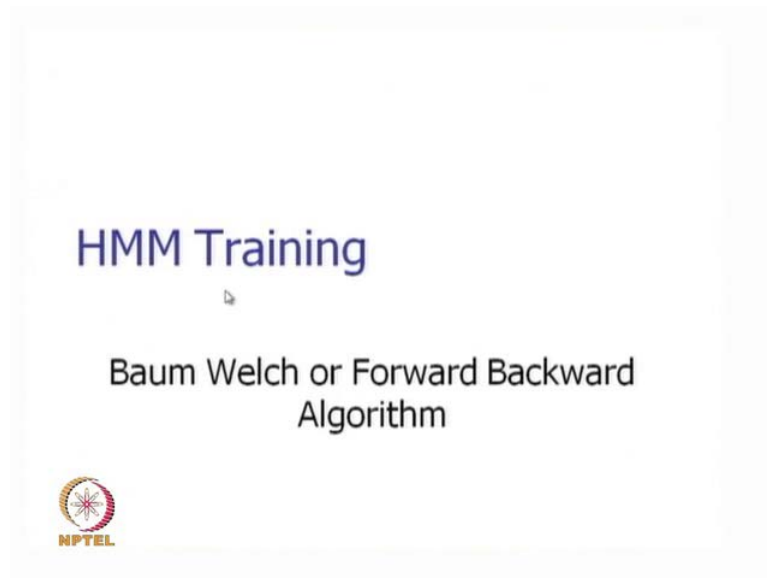
the word, which is the part of speech taggers task. So, here we have some tags from a very famous corpora called Penn tree bank tag set, CC is the coordinating conjunction, CD is a cardinal number, DT is a determiner, EX is an existential there, FW is foreign word, IN is preposition.

JJ is adjective, JJR is comparative adjective, JJS is superlative adjective, LS is list item marker, MD is modal, NN is noun, NNS is plural noun, NNP is proper noun, so a few of these tags in Penn tree bank are shown here. Now, all these tags have different examples for example, CC is the coordinating conjunction and it can be used for entities like, but and while and so on.

Now, how is this tag set created, the tag set is created by lot of insight into the way, the language operates and the words in the language are placed with respect to each other. So, some points of thought are wise it that, English has a tag for a singular noun and a separate tag for plural noun. Now, one of the important reasons for this is that, in English most nouns can be used as verbs, so the plural form of a noun is almost always same as the singular number third person present tense form of a verb.

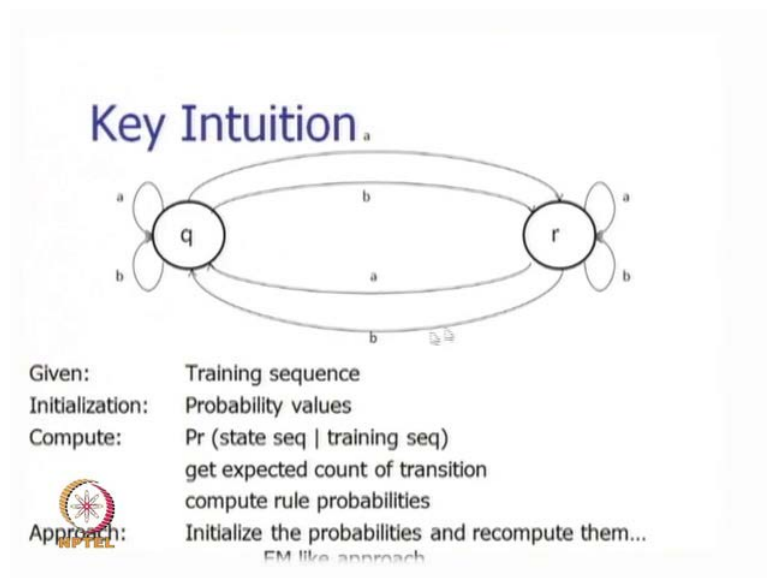
So, like John plays, plays is the third person singular number present tense form of play and we saw many plays, here plays is a noun. And both these categories that is, third person singular number present tense form of verb and plural noun, both of them are very every frequent in the language. And the system cannot afford to make mistake with respect to their tags and hence, it has become necessary to separate the noun singular tag and the plural noun tag. So similarly, each tag has lot of linguistic considerations and computational considerations behind it, which we can discuss at some point of time.

(Refer Slide Time: 35:06)



So, hidden Markov, we come back to hidden Markov model training and this is a very famously known by Baum Welch or forward backward algorithm.

(Refer Slide Time: 35:16)



So, the key intuition here is that, we have a hidden Markov model, we start with some probability values assigned on the arcs, capturing the probability of transition from state to state with a particular output symbol. And then we use an expectation maximization like algorithm for training the HMM.

(Refer Slide Time: 35:40)


Urn example revisited

Colored Ball choosing

Urn 1
of Red = 30
of Green = 50
of Blue = 20

Urn 2
of Red = 10
of Green = 40
of Blue = 50

Urn 3
of Red = 60
of Green = 10
of Blue = 30



So, if we take the urn example again then in the urn example, we have all the numbers of red balls and so on.

(Refer Slide Time: 35:51)

Example (contd.)

Transition Probability

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

Given :


Observation/output Probability

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

and

Observation : RRGGBRGR

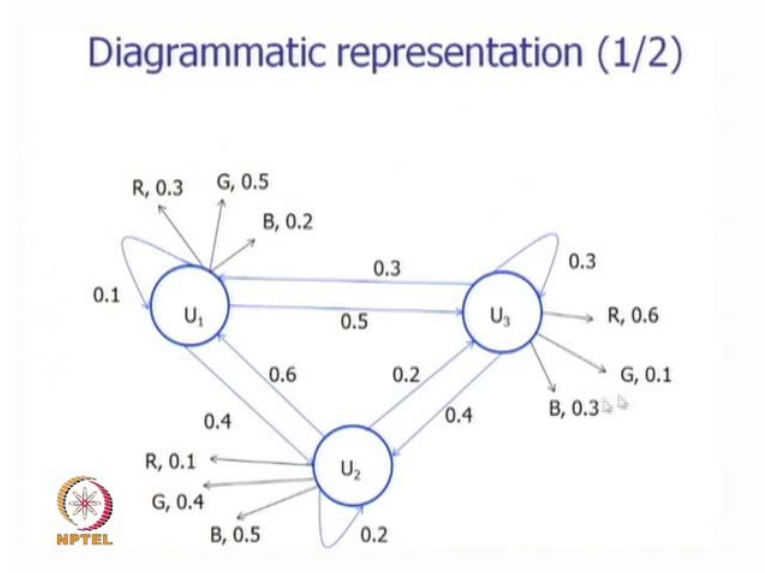
What is the corresponding state sequence ?



But, what is not given to us is this transition probability table and the observation probability table and we have seen before that, these two probabilities can be marched to have a probability value marked on the arcs. Now suppose, this observation is given, which is corresponding to the hidden Markov model R R G G B R G R then the viterbi question was, what is the corresponding state sequence. What we ask is that, if the state

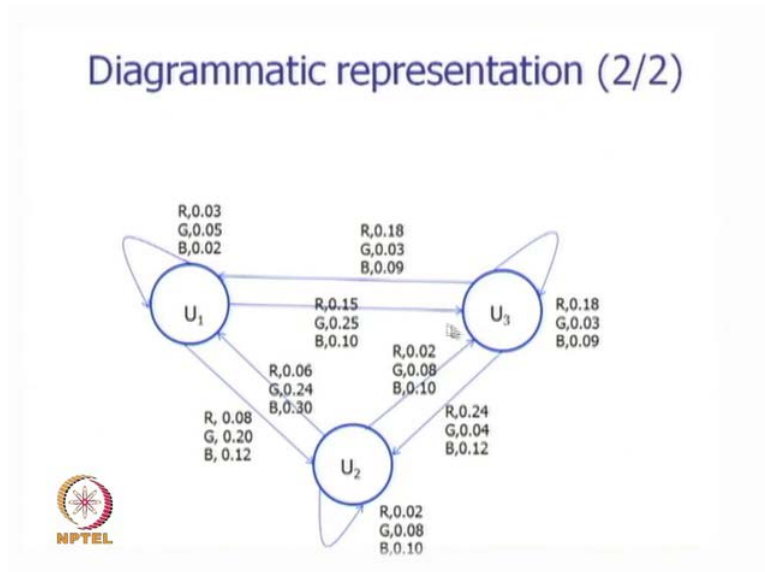
sequence is given, no if the state sequence is given that would be supervised situation. If only the observation sequences are given and the structure of the hidden Markov model is given then how can we estimate the transitional observation probabilities.

(Refer Slide Time: 36:39)



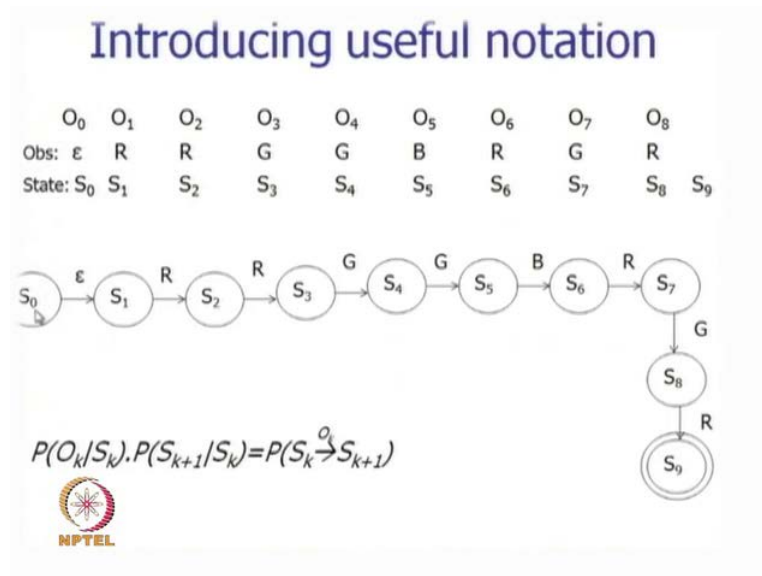
So, this is a question, in this Markov model we have hidden Markov model, we have given the observation and transition probabilities.

(Refer Slide Time: 36:46)



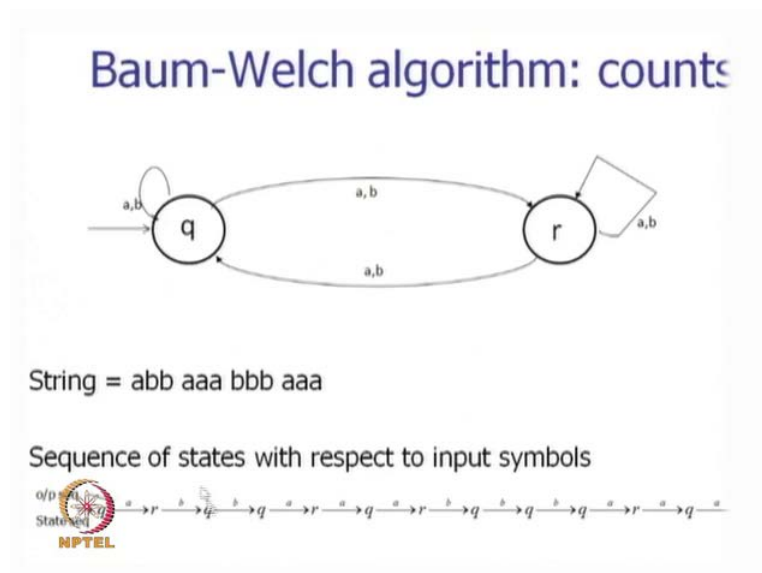
And they have been combined to produce these kinds of probabilities on the arcs and we have this kind observation sequences.

(Refer Slide Time: 36:58)



From the observation sequences, it is possible to go through either one single state sequence or a number of state sequences, but what is given is the observation sequence. And we can go through multiple state sequence for sequences for one particular observation sequence. When that happens, how do we get the parameters of the F S m.

(Refer Slide Time: 37:25)



So, why shall we discuss this question of the best possible state sequence given the observation sequence, we have seen how hidden Markov model and the Markov assumption can do this job. Now, we come to the Baum Welch algorithm and this is in

terms of counts of the transitions in the output observation sequence. So, to understand this point, let us take a sequence here abb aaa bbb aaa, so abb aaa bbb aaa this can be produced from a state sequence of this kind.

(Refer Slide Time: 38:16)

Calculating probabilities from table

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

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

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

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

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

And if this is the only state sequence, which produce the observation sequence then we can compute, we can make a good estimate of the probabilities of transitions and the output symbols just simply by counting the transitions and the outputs. So, in this particular observation sequence and the corresponding state sequence, we find that, q to r transition with output symbol a occurs five times, q to q transition with output symbol b happens three times, r to q transition with output symbol a happens three times, r to q with b happens two times.

So, from these observation, we can propose that the probability of q going to r on a is 5 by 8 and probability of q going to r on b is 3 by 8. So, the q to r transition is the only transition from state to state possible when the starting state is q and there are two possibilities of output being a or b. So, we have therefore, the probability of P q going to r on a is 5 by 8, there are eight cases of transition from q and then if we capture the next state as r and different symbols as a and b then this is what the probability comes out to be.


So, this can also be expressed by a formula here, probability of S i to S j with output symbol W k is nothing but the number of times S i goes to S j with W k. And this divided

by all the cases of S_i going to different states S_{i+1} with different output different output symbol W_m . That is why, there are two sigma symbols here, one is for all possible output symbols and the other is for all possible destination states. So, this is a simple formula, which will work in terms of exact frequency values, if an observation sequence produces a single state. But, problem is that, we have non deterministic transitions and then multiple state sequences are possible for a given output sequence. So, our aim is to find the expected count through this.

(Refer Slide Time: 40:36)

Interplay Between Two Equations

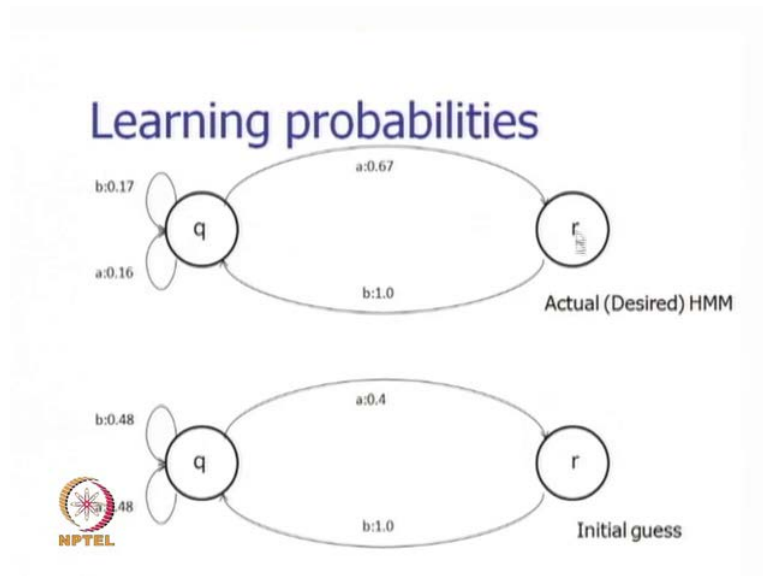
$$P(s^i \xrightarrow{W_k} s^j) = \frac{c(s^i \xrightarrow{W_k} s^j)}{\sum_{l=1}^T \sum_{m=1}^A c(s^l \xrightarrow{W_k} s^j)}$$

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


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

So, the whole computation happens by an interplay between a probability calculation in terms of counts and estimating the counts. Expected counts rather from the values of the transitions, the number of transition occurs and weighting it by the probability of state sequence given the observation sequence.

(Refer Slide Time: 41:03)



So, this is illustrated by an HMM here, this is the actual desired HMM and this is the initial guess of the HMM.

(Refer Slide Time: 41:12)

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>(0.01/(0.01+0.06+0.095))</small>	1.0	0.36	0.581
State sequences										

* ϵ is considered as starting and ending symbol of the input sequence string

This way through multiple iterations the probability values will converge.

And from this, we can find out the path probabilities and from this, we can make another estimate. So, this particular algorithm, we will discuss in detail in the next lecture and we will see that, this produces a very interesting algorithm for training the HMM.