ENCS5341
Machine Learning and Data Science

Hidden Markov Model

Slides are based on slides by Xing, Precup, Rabusseau, Govindaraju, Jurafsky, and Martin

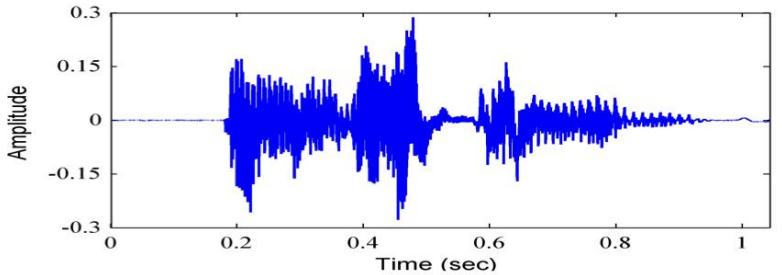
STUDENTS-HUB.com

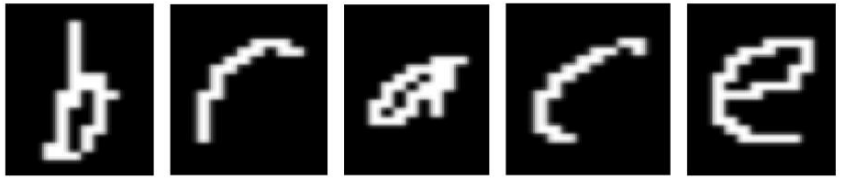
Uploaded By: Jibreel Bornat

i.i.d to Sequential Data

- So far we assumed independent, identically distributed data.

$$\{X_i\}_{i=1}^{n} \overset{iid}{\sim} p(\mathbf{X})$$





- In practice, many applications are based on sequential data
  - Time-series data. E.g. Speech
  - Characters in a sentence
  - Base pairs along a DNA strand

- Does this fit the machine learning paradigm as described so far?
  - The sequences are not all the same length (so we cannot just assume one attribute per time step)
  - The data at each time slice/index is not independent
  - The data distribution may change over time

Markov Model

- The HMM is based on augmenting the Markov chain.

- A Markov chain is a model that tells us something about the probabilities of sequences of random variables, states, each of which can take on values from some set.
  - These sets can be words, or tags, or symbols representing anything, like the weather.

- A Markov chain makes a very strong assumption that if we want to predict the future in the sequence, all that matters is the current state.
  - E.g. to predict tomorrow's weather you could examine today's weather but you weren't allowed to look at yesterday's weather.

- Markov Assumption:

$$P(S_t|S_1, S_2, \dots, S_{t-1}) = P(S_t|S_{t-1})$$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Markov Model Definition

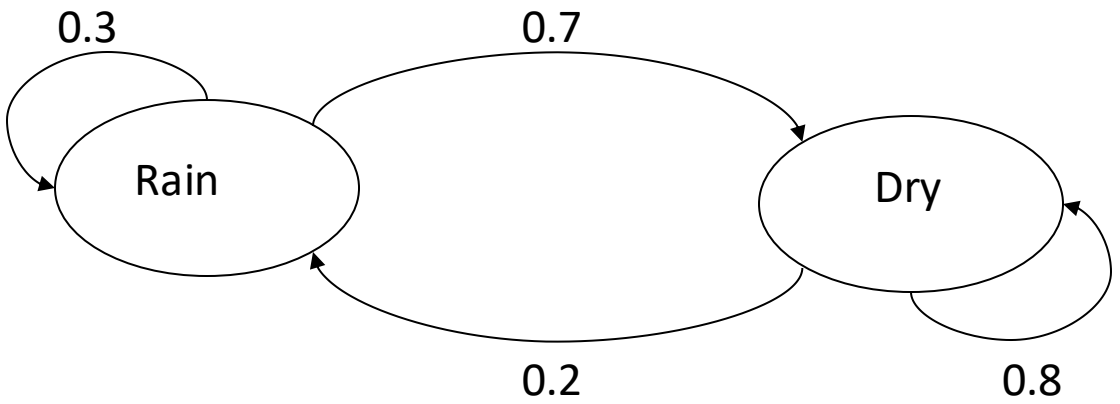
A Markov model is defined by the followings:

- Set of states: $\{s_1, s_2, \ldots, s_N\}$
- Process moves from one state to another generating a sequence of states : $s_{i1}, s_{i2}, \ldots, s_{ik}, \ldots$
- Markov chain property: probability of each subsequent state depends only on what was the previous state:

$$P(s_{ik} \mid s_{i1}, s_{i2}, \ldots, s_{ik-1}) = P(s_{ik} \mid s_{ik-1})$$

- To define Markov model, the following probabilities have to be specified:
  - transition probabilities $a_{ij} = P(s_i \mid s_j)$
  - and initial probabilities $\pi_i = P(s_i)$

Example of Markov Model



- Two states : ‘Rain’ and ‘Dry’.
- Transition probabilities: P(‘Rain’|‘Rain’)=0.3 , P(‘Dry’|‘Rain’)=0.7 , P(‘Rain’|‘Dry’)=0.2, P(‘Dry’|‘Dry’)=0.8
- Initial probabilities: say P(‘Rain’)=0.4 , P(‘Dry’)=0.6 .

STUDENTS-HUB.com Uploaded By: Jibreel Bornat

Calculation of sequence probability

- By Markov chain property, probability of state sequence can be found by the formula:

$$P(s_{i1}, s_{i2}, \ldots, s_{ik}) = P(s_{ik} \mid s_{i1}, s_{i2}, \ldots, s_{ik-1}) P(s_{i1}, s_{i2}, \ldots, s_{ik-1})$$
$$= P(s_{ik} \mid s_{ik-1}) P(s_{i1}, s_{i2}, \ldots, s_{ik-1}) = \ldots$$
$$= P(s_{ik} \mid s_{ik-1}) P(s_{ik-1} \mid s_{ik-2}) \ldots P(s_{i2} \mid s_{i1}) P(s_{i1})$$

- Suppose we want to calculate the probability of the following sequence of states {'Dry','Dry','Rain',Rain'}.

P({'Dry','Dry','Rain',Rain'}) = P('Rain'|'Rain') P('Rain'|'Dry') P('Dry'|'Dry') P('Dry')= 0.3*0.2*0.8*0.6

Hidden Markov Model (HMM)

- Hidden Markov Models (HMMs) are used for situations in which:
  - The data consists of a sequence of observations
  - The observations depend (probabilistically) on the internal state of a dynamical system
  - The true state of the system is unknown (i.e., it is a hidden or latent variable)

- There are numerous applications, including:
  - Speech recognition
  - Robot localization
  - Gene finding
  - User modelling
  - Fetal heart rate monitoring
  - …

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Hidden Markov Model Definition

An HMM is specified by the following components:

$Q = q_1 q_2 \ldots q_N$	a set of N **states**
$A = a_{11} \ldots a_{ij} \ldots a_{NN}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
$\pi = \pi_1, \pi_2, ..., \pi_N$	an **initial probability distribution** over states. π_i is the probability that the Markov chain will start in state i. Some states j may have $\pi_j = 0$, meaning that they cannot be initial states. Also, $\sum_{i=1}^{n} \pi_i = 1$

Two simplifying assumptions for first-order HMMs:

Markov Assumption: $P(q_i|q_1 ... q_{i-1}) = P(q_i|q_{i-1})$

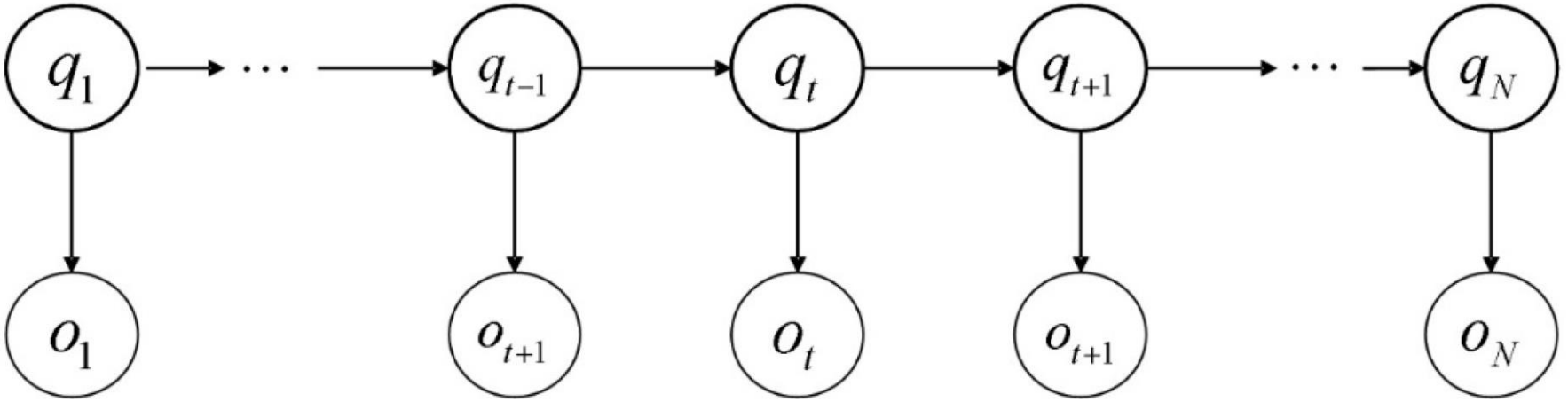
Output Independence: $P(o_i|q_1 \ldots q_i, \ldots, q_T, o_1, \ldots, o_i, \ldots, o_T) = P(o_i|q_i)$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

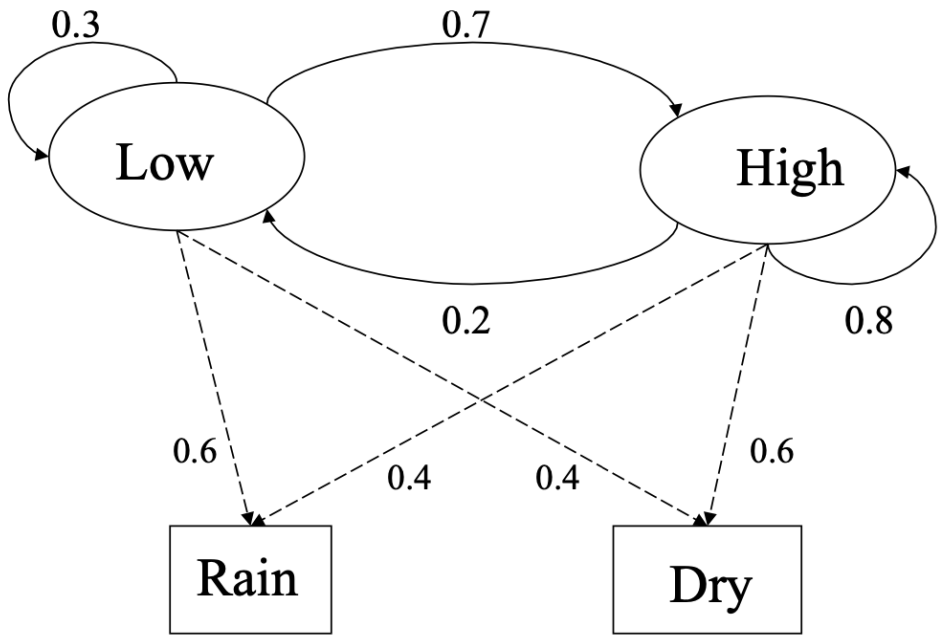
How an HMM works

- Assume a discrete clock t = 0,1,2,...
- At each t, the system is in some internal (hidden) state q_t and an observation o_t is emitted (stochastically) based only on q.
- The system transitions (stochastically) to a new state q_{t+1}, according to a probability distribution $P(q_{t+1} \mid q_t)$, and the process repeats.
- This interaction can be represented as a graphical model



Example of Hidden Markov Model

- Two states : ‘Low’ and ‘High’ atmospheric pressure.
- Two observations : ‘Rain’ and ‘Dry’.
- Transition probabilities: P(‘Low’|‘Low’)=0.3 , P(‘High’|‘Low’)=0.7 , P(‘Low’|‘High’)=0.2, P(‘High’|‘High’)=0.8
- Observation probabilities : P(‘Rain’|‘Low’)=0.6 , P(‘Dry’|‘Low’)=0.4 , P(‘Rain’|‘High’)=0.4 , P(‘Dry’|‘High’)=0.6 .
- Initial probabilities: say P(‘Low’)=0.4 , P(‘High’)=0.6 .

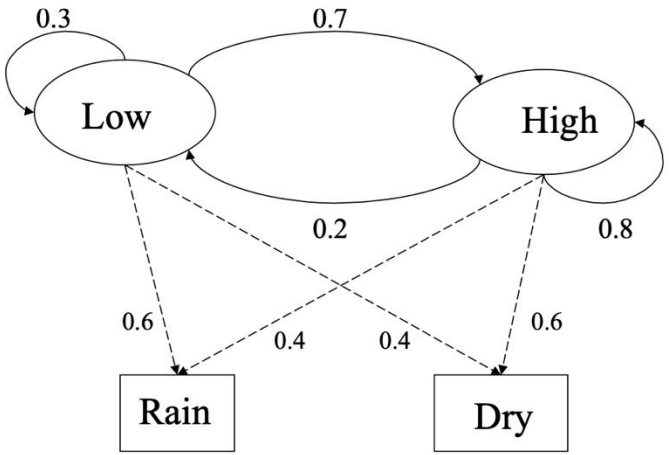


Calculation of observation sequence probability

- Suppose we want to calculate a probability of a sequence of observations in our example, {‘Dry’,’Rain’}.

- Consider all possible hidden state sequences:
  P({‘Dry’,’Rain’}) = P({‘Dry’,’Rain’} , {‘Low’,’Low’}) + P({‘Dry’,’Rain’} , {‘Low’,’High’}) + P({‘Dry’,’Rain’} , {‘High’,’Low’}) + P({‘Dry’,’Rain’} , {‘High’,’High’})

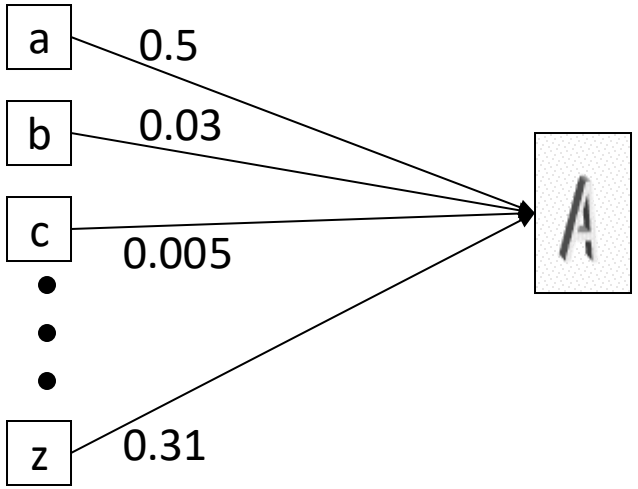
- where first term is :
  P({‘Dry’,’Rain’} , {‘Low’,’Low’})= P({‘Dry’,’Rain’} | {‘Low’,’Low’}) P({‘Low’,’Low’})
  = P(‘Dry’|’Low’)P(‘Rain’|’Low’) P(‘Low’)P(‘Low’|’Low) = 0.4*0.4*0.6*0.4*0.3



Word recognition example(1).

- Typed word recognition, assume all characters are separated.

- Character recognizer outputs probability of the image being particular character, P(image|character).



Word recognition example(2).

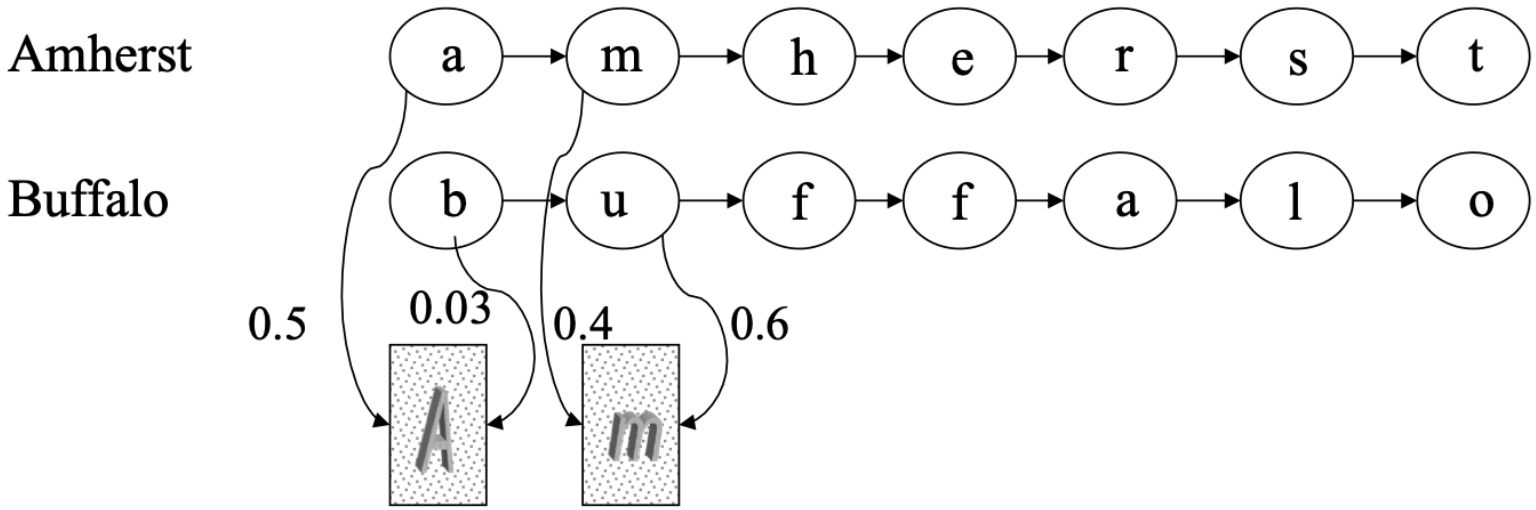
- Hidden states of HMM = characters.
- Observations = typed images of characters segmented from the image v_α .
  Note that there is an infinite number of observations
- Observation probabilities = character recognizer scores.

$$B = \left(b_i(v_\alpha)\right) = \left(P(v_\alpha \mid s_i)\right)$$

- Transition probabilities will be defined differently in two subsequent models.

Word recognition example(3).

- If lexicon is given, we can construct separate HMM models for each lexicon word.

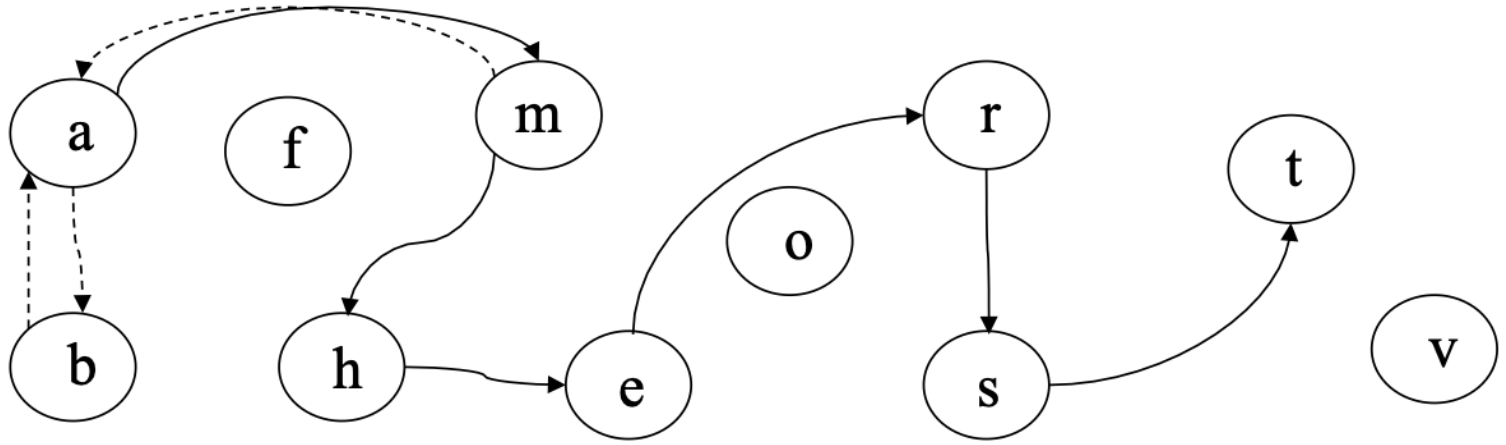


- Here recognition of word image is equivalent to the problem of evaluating few HMM models.

- This is an application of Evaluation problem.

Word recognition example(4).

- We can construct a single HMM for all words.
- Hidden states = all characters in the alphabet.
- Transition probabilities and initial probabilities are calculated from language model.
- Observations and observation probabilities are as before.



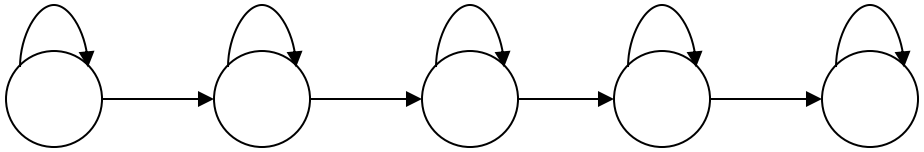
- Here we have to determine the best sequence of hidden states, the one that most likely produced word image.
- This is an application of Decoding problem.

STUDENTS-HUB.com

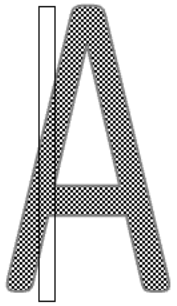
Uploaded By: Jibreel Bornat

Character recognition with HMM example.

- The structure of hidden states is chosen.



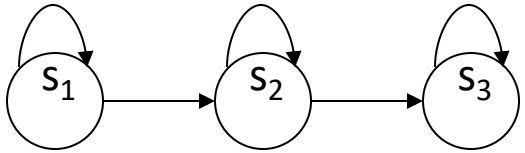
- Observations are feature vectors extracted from vertical slices.



- Probabilistic mapping from hidden state to feature vectors:
  - use mixture of Gaussian models
  - Quantize feature vector space.

Exercise: character recognition with HMM(1)

- The structure of hidden states:

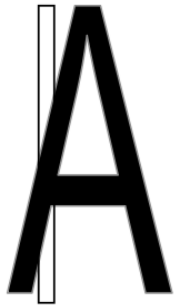


- Observation = number of islands in the vertical slice.

- HMM for character 'A' :

$$\text{Transition probabilities: } \{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$$

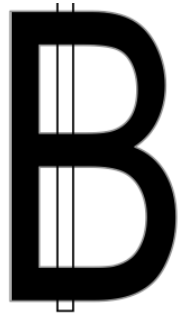
$$\text{Observation probabilities: } \{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ .1 & .8 & .1 \\ .9 & .1 & 0 \end{pmatrix}$$



- HMM for character 'B' :

$$\text{Transition probabilities: } \{a_{ij}\} = \begin{pmatrix} .8 & .2 & 0 \\ 0 & .8 & .2 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{Observation probabilities: } \{b_{jk}\} = \begin{pmatrix} .9 & .1 & 0 \\ 0 & .2 & .8 \\ .6 & .4 & 0 \end{pmatrix}$$



STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Exercise: character recognition with HMM(2)

- Suppose that after character image segmentation the following sequence of island numbers in 4 slices was observed: { 1, 3, 2, 1}

- What HMM is more likely to generate this observation sequence , HMM for ‘A’ or HMM for ‘B’ ?

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Exercise: character recognition with HMM(3)

- Consider likelihood of generating given observation for each possible sequence of hidden states:

- HMM for character 'A':

Hidden state sequence	Transition probabilities		Observation probabilities
$S_1 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$	.8 * .2 * .2	*	.9 * 0 * .8 * .9 = 0
$S_1 \rightarrow S_2 \rightarrow S_2 \rightarrow S_3$	.2 * .8 * .2	*	.9 * .1 * .8 * .9 = 0.0020736
$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_3$	.2 * .2 * 1	*	.9 * .1 * .1 * .9 = 0.000324
			Total = 0.0023976

- HMM for character 'B':

Hidden state sequence	Transition probabilities		Observation probabilities
$S_1 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$	.8 * .2 * .2	*	.9 * 0 * .2 * .6 = 0
$S_1 \rightarrow S_2 \rightarrow S_2 \rightarrow S_3$	.2 * .8 * .2	*	.9 * .8 * .2 * .6 = 0.0027648
$S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_3$	.2 * .2 * 1	*	.9 * .8 * .4 * .6 = 0.006912
			Total = 0.0096768

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Three main problems in HMMs

- Evaluation: Given HMM parameters & observation sequence find prob of observed sequence.

  Given $\{O_t\}_{t=1}^{T}$ and model parameters, find $p(\{O_t\}_{t=1}^{T})$

- Decoding: Given HMM parameters & observation sequence find most probable sequence of hidden states

  Given $\{O_t\}_{t=1}^{T}$ and model parameters, find $\arg \max_{s_1,\ldots,s_T} p(\{S_t\}_{t=1}^{T} | \{O_t\}_{t=1}^{T})$

- Learning: Given HMM with unknown parameters and observation sequence find parameters that maximize likelihood of observed data

  Given $\{O_t\}_{t=1}^{T}$, find $\arg \max_{\theta} p(\{O_t\}_{t=1}^{T} | \theta)$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Evaluation Problem - Likelihood Computation

- Computing Likelihood: Given an HMM $\lambda = (A, B, \pi)$ and an observation sequence $O = (o_1, o_2, \ldots, o_T)$, determine the likelihood $P(O|\lambda)$.

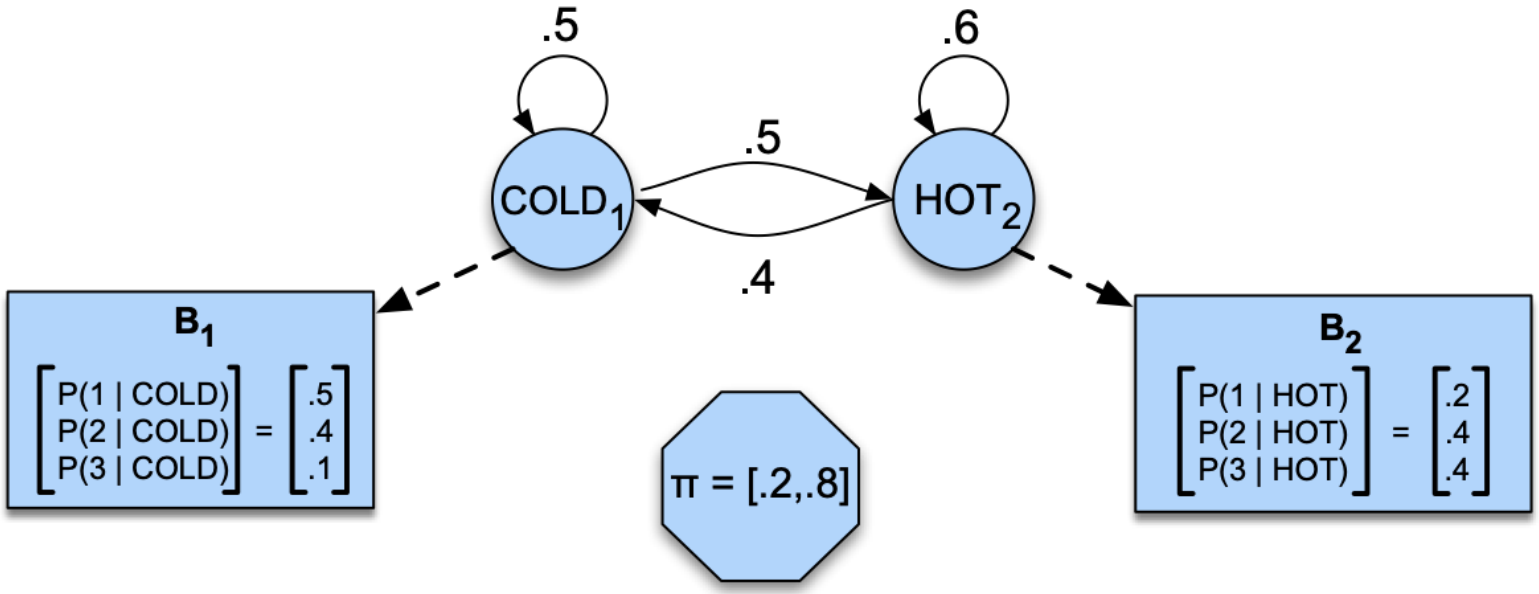
- Trying to find probability of observations $O = (o_1, o_2, \ldots, o_T)$ by means of considering all hidden state sequences (as was done in the example) is impractical.

- N^T hidden state sequences - exponential complexity.

- Solution: Instead of using such an extremely exponential algorithm, we use an efficient $O(N^2T)$ algorithm called the forward algorithm.

Example: The Ice Cream Task (1)

- The two hidden states (H and C) correspond to hot and cold weather.
- observations (drawn from the alphabet O = {1, 2, 3}) correspond to the number of ice creams eaten.



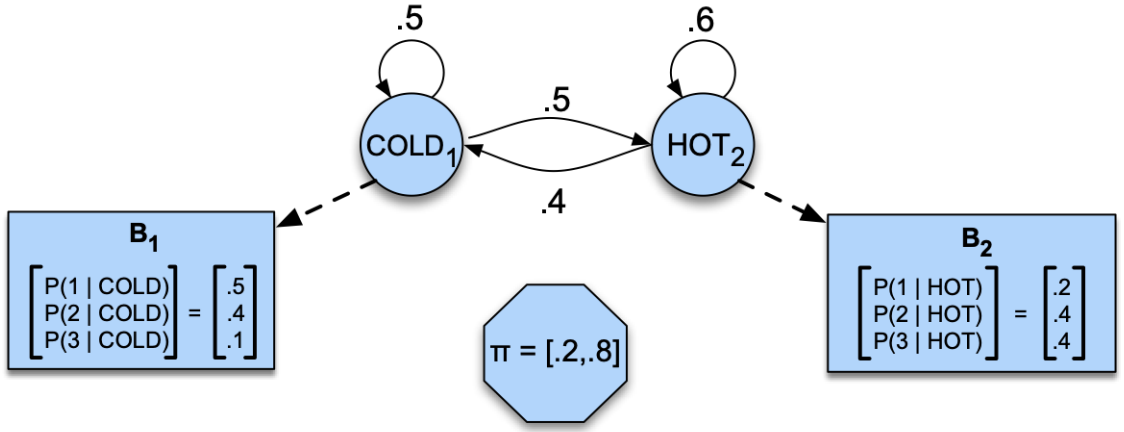
- Question 1: What is the probability of observing the sequence 3 1 3 given that the weather was *hot hot cold* ?
- Question 2: What is the probability of observing the sequence 3 1 3 ?

STUDENTS-HUB.com
Uploaded By: Jibreel Bornat

Example: The Ice Cream Task (2).

- Question 1: What is the probability of observing the sequence 3 1 3 given that the weather was *hot hot cold* ?

- **Solution**:



$$P(O|Q) = \prod_{i=1}^{T} P(o_i|q_i)$$

$$P(3\ 1\ 3|\text{hot hot cold}) = P(3|\text{hot}) \times P(1|\text{hot}) \times P(3|\text{cold})$$

$$= 0.4 * 0.2 * 0.1$$

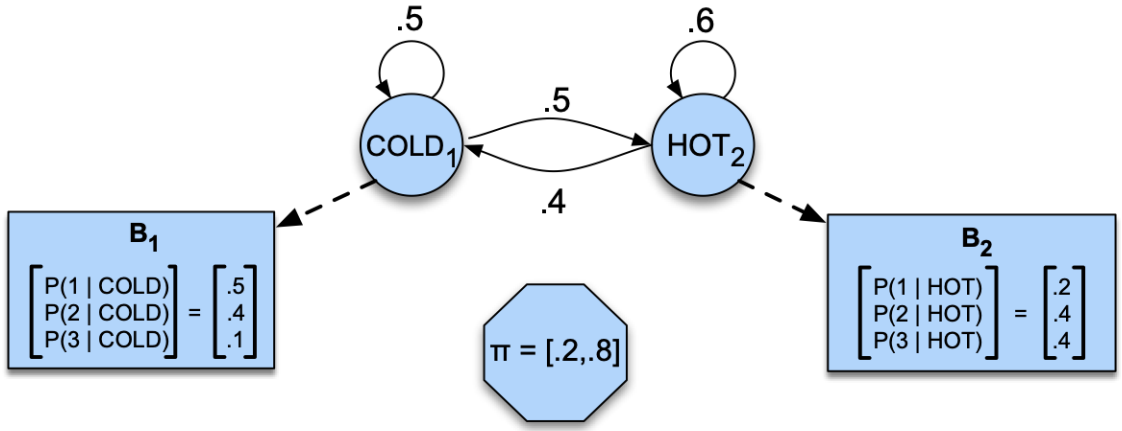
$$= 0.008$$

Example: The Ice Cream Task (3).

- Question 2: What is the probability of observing the sequence 3 1 3 ?

- **Naïve Solution**:

$$P(O) = \sum_{Q} P(O,Q) = \sum_{Q} P(O|Q)P(Q)$$



Such that $P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{T} P(o_i|q_i) \times \prod_{i=1}^{T} P(q_i|q_{i-1})$

- For our case, we would sum over the eight 3-event sequences cold cold cold, cold cold hot, …, etc. that is: $P(3\ 1\ 3) = P(3\ 1\ 3, \text{cold cold cold}) + P(3\ 1\ 3, \text{cold cold hot}) + P(3\ 1\ 3, \text{hot hot cold}) + \ldots$

- For example, $P(3\ 1\ 3, \text{hot hot cold}) = P(\text{hot}|\text{start}) \times P(\text{hot}|\text{hot}) \times P(\text{cold}|\text{hot}) \times P(3|\text{hot}) \times P(1|\text{hot}) \times P(3|\text{cold})$

- *Note*: We can solve the problem in an efficient way using the Forward Algorithm

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Forward Algorithm (1)

- For an HMM with N hidden states and an observation sequence of length T, there are N^T possible hidden sequences.
- For real tasks, where N and T are both large, N^T is a very large number, so we cannot compute the total observation likelihood by computing a separate observation likelihood for each hidden state sequence and then summing them.
- Instead of using such an exponential algorithm, we use an efficient Forward $O(N^2T)$ algorithm called the forward algorithm.
- The forward algorithm is a kind of dynamic programming algorithm. The algorithm uses a table to store intermediate values as it builds up the probability of the observation sequence.
- The forward algorithm computes the observation probability by summing over the probabilities of all possible hidden state paths that could generate the observation sequence, but it does so efficiently by implicitly folding each of these paths into a single forward trellis.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Forward Algorithm (2)

- Let $\alpha_t(j)$ represents the probability of being in state j after seeing the first t observations, given the HMM parameters λ

$$\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

- The value of $\alpha_t(j)$ is computed by summing over the probabilities of every path that could lead us to state j at time t.

- We can compute $\alpha_t(j)$ recursively by summing over the extensions of all paths from the previous time step that could lead us to the current state.

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$$

- The three factors that are multiplied in the previous equation in extending the previous paths to compute the forward probability at time t are

$\alpha_{t-1}(i)$	the **previous forward path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Forward Algorithm (3)

- The following steps are used compute the probability of a sequence of observation given the HMM parameters λ. (i.e. $P(O|\lambda)$)

1. Initialization:

$$\alpha_1(j) = \pi_j b_j(o_1) \quad 1 \leq j \leq N$$

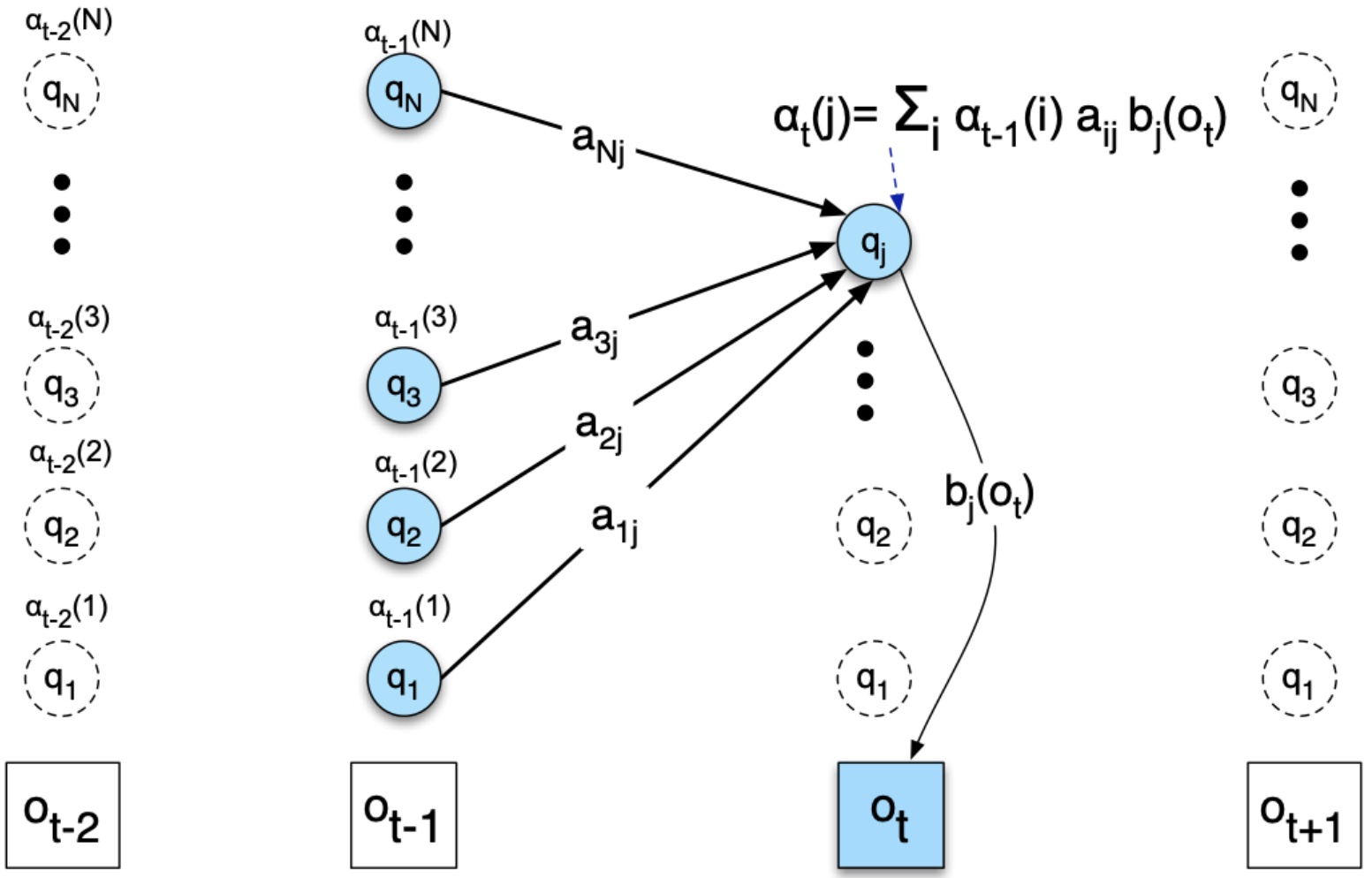
2. Recursion:

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t); \quad 1 \leq j \leq N, 1 < t \leq T$$

3. Termination:

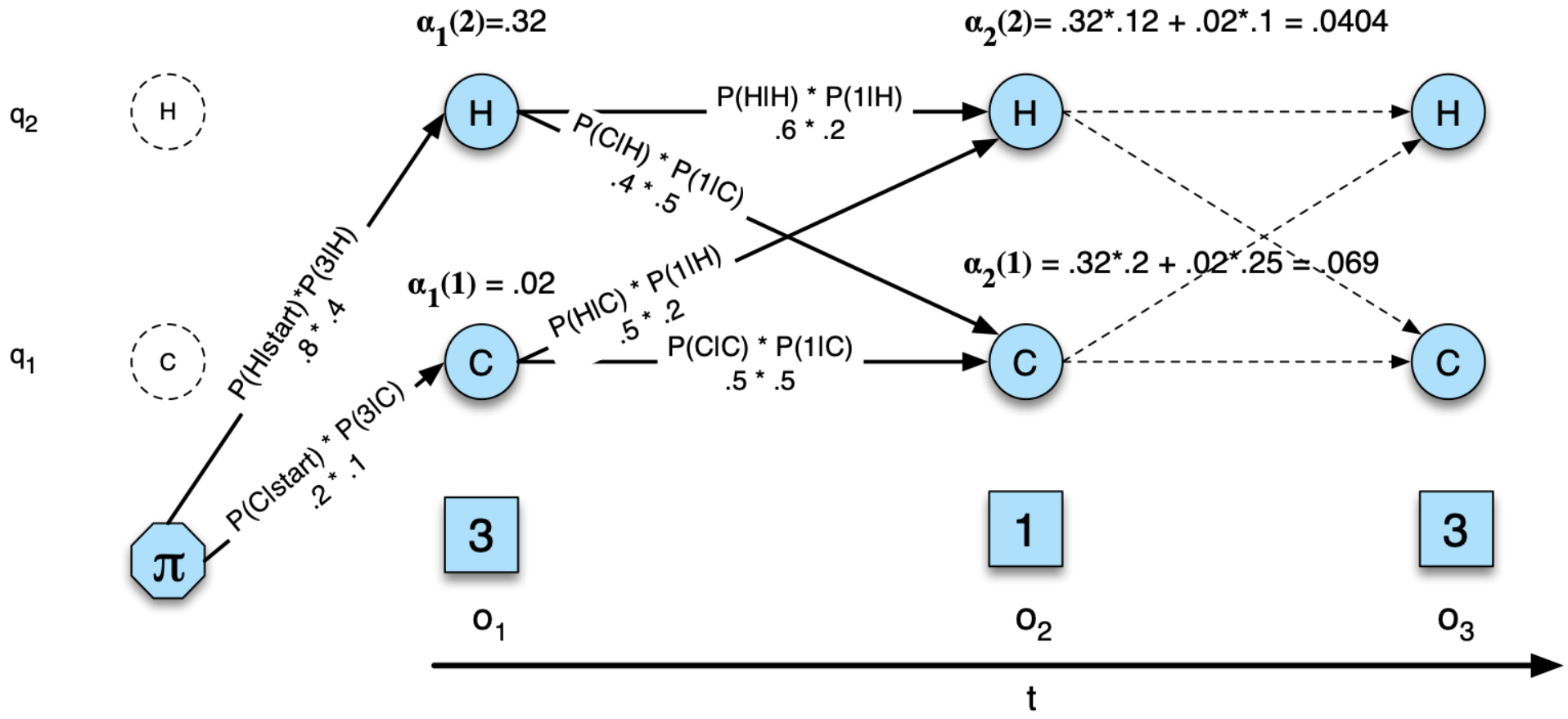
$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Visualizing the computation of a single element $\alpha_t(i)$ in the trellis



Computing Likelihood using Forward Algorithm

- Now we can use forward trellis for computing the total observation likelihood for the ice-cream events 3 1 3



STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Decoding Problem

- Given an HMM $\lambda = (A, B, \pi)$ and an observation sequence $O = (o_1, o_2, \ldots, o_T)$, find the most probable sequence of states $Q = q_1 q_2 q_3 \ldots q_T$
- In the ice-cream example, given a sequence of ice-cream observations 3 1 3 and an HMM, the task of the decoder is to find the best hidden weather sequence (H H H)
- We might propose to find the best sequence as follows:
  - For each possible hidden state sequence (HHH, HHC, HCH, etc.), we could compute the likelihood of the observation sequence given that hidden state sequence.
  - Then we could choose the hidden state sequence with the maximum observation likelihood.
- Brute force consideration of all paths takes exponential time. Use efficient Viterbi algorithm instead.
- Like the forward algorithm, Viterbi is a kind of dynamic programming.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Viterbi algorithm (1)

- Let $v_t(j)$ be the probability of the most probable path that could lead us to state j at time t given the HMM parameters λ

$$v_t(j) = \max_{q_1,\cdots,q_{t-1}} P(q_1 \ldots q_{t-1}, o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

- Note that we represent the most probable path by taking the maximum over all possible previous state sequences $\max_{q_1,\cdots,q_{t-1}}$

- We can compute $v_t(j)$ recursively. Given that we had already computed the probability of being in every state at time t − 1, we compute the Viterbi probability by taking the most probable of the extensions of the paths that lead to the current state.

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

- The three factors that are multiplied in the previous equation for extending the previous paths to compute the Viterbi probability at time t are

$v_{t-1}(i)$	the **previous Viterbi path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

STUDENTS-HUB.com Uploaded By: Jibreel Bornat

The Viterbi algorithm (2)

- we can give a formal definition of the Viterbi recursion as follows:

1. **Initialization:**

$$\begin{aligned} v_1(j) &= \pi_j b_j(o_1) \qquad 1 \le j \le N \\ bt_1(j) &= 0 \qquad\qquad\quad 1 \le j \le N \end{aligned}$$

2. **Recursion**

$$\begin{aligned} v_t(j) &= \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t); \quad 1 \le j \le N, 1 < t \le T \\ bt_t(j) &= \operatorname*{argmax}_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t); \quad 1 \le j \le N, 1 < t \le T \end{aligned}$$

3. **Termination:**

$$\text{The best score:} \quad P* = \max_{i=1}^{N} v_T(i)$$

$$\text{The start of backtrace:} \quad q_T* = \operatorname*{argmax}_{i=1}^{N} v_T(i)$$

The Viterbi algorithm (3)

- Note that the Viterbi algorithm is identical to the forward algorithm except that it takes the max over the previous path probabilities whereas the forward algorithm takes the sum.

- Note also that the Viterbi algorithm has one component that the forward algorithm doesn't have: backpointers $bt_t(j)$

- The reason that Viterbi has backpointers is that while the forward algorithm needs to produce an observation likelihood, the Viterbi algorithm must produce a probability and also the most likely state sequence.

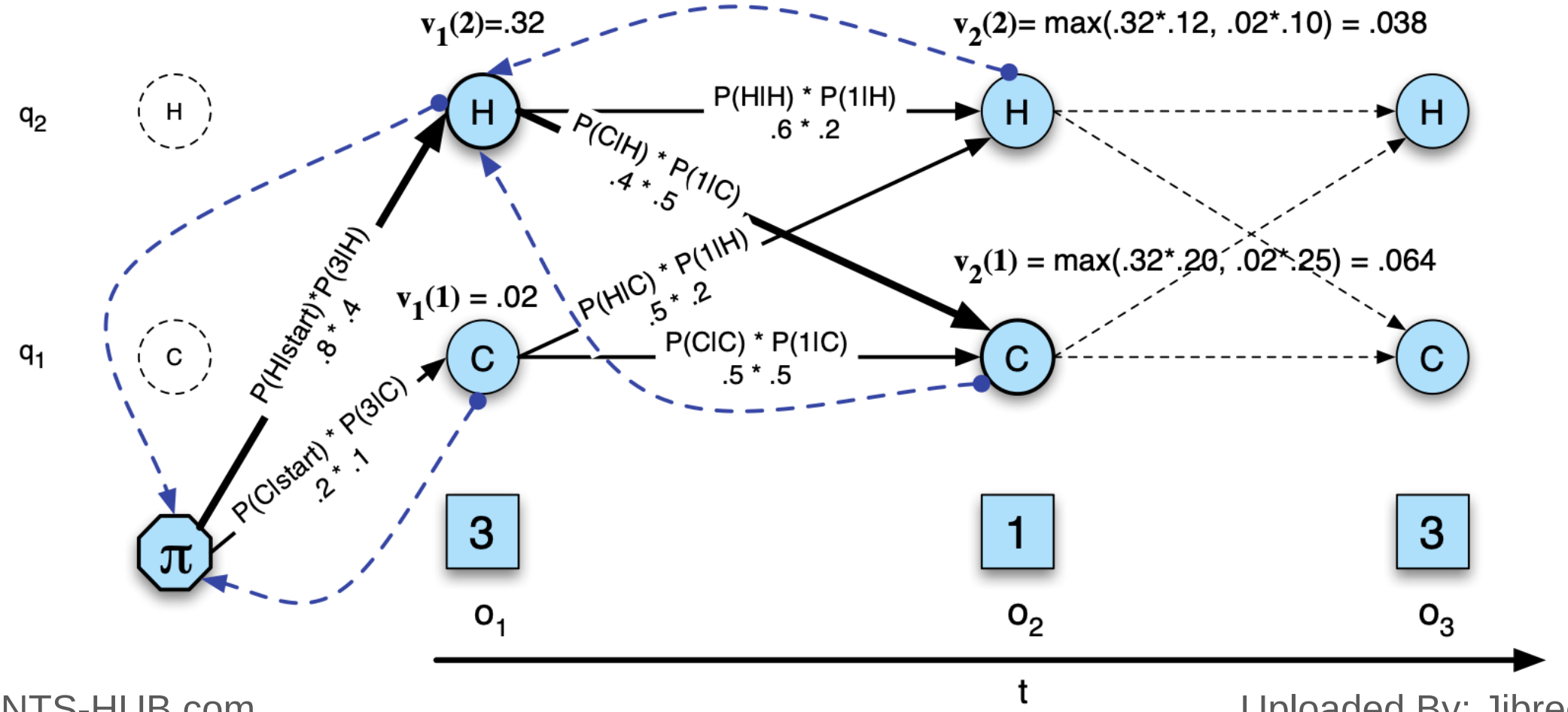
- We compute this best state sequence by keeping track of the path of hidden states that led to each state and then at the end backtracing the best path to the beginning

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Viterbi Example

- given a sequence of ice-cream observations 3 1 3 and an HMM, find the best hidden weather sequence
- Solution (incomplete)



STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Learning Problem

- Given an observation sequence $O=(o_1, o_2, \ldots, o_T)$ and general structure of HMM (numbers of hidden and visible states), learn HMM parameters $\lambda=(A, B, \pi)$ that best fit training data, that is maximizes $P(O|\lambda)$.
- The standard algorithm for HMM training is the forward-backward, or Baum-Welch algorithm.
- It is a special case of the Expectation-Maximization (EM) algorithm.
- EM is an iterative algorithm, computing an initial estimate for the probabilities, then using those estimates to computing a better estimate, and so on, iteratively improving the probabilities that it learns.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Learning Problem – A simple Version

- Let us begin by considering the much simpler case of training a fully **visible** Markov model, where we know both the state and the observation.
- Example:
  - Training sequences

```
3     3    2          1     1     2          1     2   3
hot   hot  cold       cold  cold  cold       cold  hot hot
```

  - First, we can compute π from the count of the 3 initial hidden states: $\pi_h = 1/3 \quad \pi_c = 2/3$
  - Next we can directly compute the A matrix from the transitions, ignoring the final hidden states:

$$p(hot|hot) = 2/3 \quad p(cold|hot) = 1/3$$
$$p(cold|cold) = 2/3 \quad p(hot|cold) = 1/3$$

  - and the B matrix:

$$P(1|hot) = 0/4 = 0 \quad p(1|cold) = 3/5 = .6$$
$$P(2|hot) = 1/4 = .25 \quad p(2|cold = 2/5 = .4$$
$$P(3|hot) = 3/4 = .75 \quad p(3|cold) = 0$$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Back to the HMM Learning Problem

- For a real HMM, we cannot compute these counts directly from an observation sequence since we don't know which path of states was taken through the machine for a given input.
- The Baum-Welch algorithm solves this by iteratively estimating the counts.
- We will start with an estimate for the transition and observation probabilities and then use these estimated probabilities to derive better and better probabilities.
- To understand the algorithm, we need to define a useful probability related to the forward probability and called the backward probability.

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Backward Probability

- The backward probability $\beta_t(i)$ is the probability of seeing the observations from time t + 1 to the end, given that we are in state i at time t

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \dots o_T | q_t = i, \lambda)$$

- It is computed recursively in a similar manner to the forward algorithm.

1. **Initialization:**

$$\beta_T(i) \;=\; 1, \quad 1 \le i \le N$$

2. **Recursion**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j), \quad 1 \le i \le N, 1 \le t < T$$

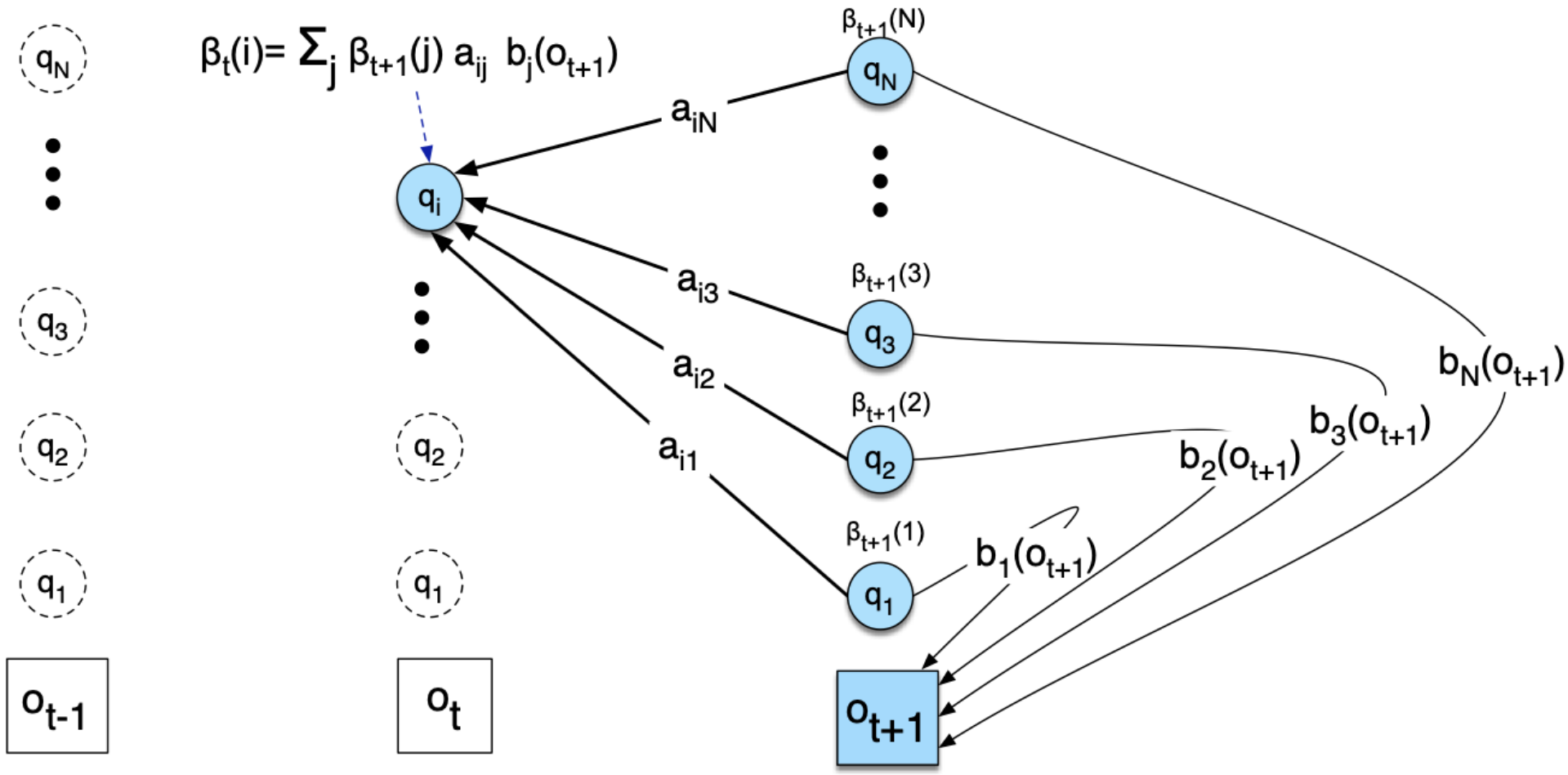
3. **Termination:**

$$P(O|\lambda) = \sum_{j=1}^{N} \pi_j\, b_j(o_1)\, \beta_1(j)$$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

The Backward Induction Step



Estimating the transition probabilities a_{ij} (1)

- Let's begin by seeing how to estimate $\hat{a}_{ij}$ by a variant of simple maximum likelihood estimation:

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

- To compute the numerator, assume we had some estimate of the probability that a given transition i → j was taken at a particular point in time t in the observation sequence. If we knew this probability for each particular time t, we could sum over all times t to estimate the total count for the transition i → j.

- More formally, let's define the probability ξ_t as the probability of being in state i at time t and state j at time t + 1, given the observation sequence and of course the model:

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | O, \lambda)$$

Estimating the transition probabilities a_{ij} (2)

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda)$$

- To compute ξ_t , we first compute a probability which is similar to ξ_t , but differs in including the probability of the observation

$$\text{not-quite-}\xi_t(i,j) = P(q_t = i, q_{t+1} = j, O|\lambda)$$

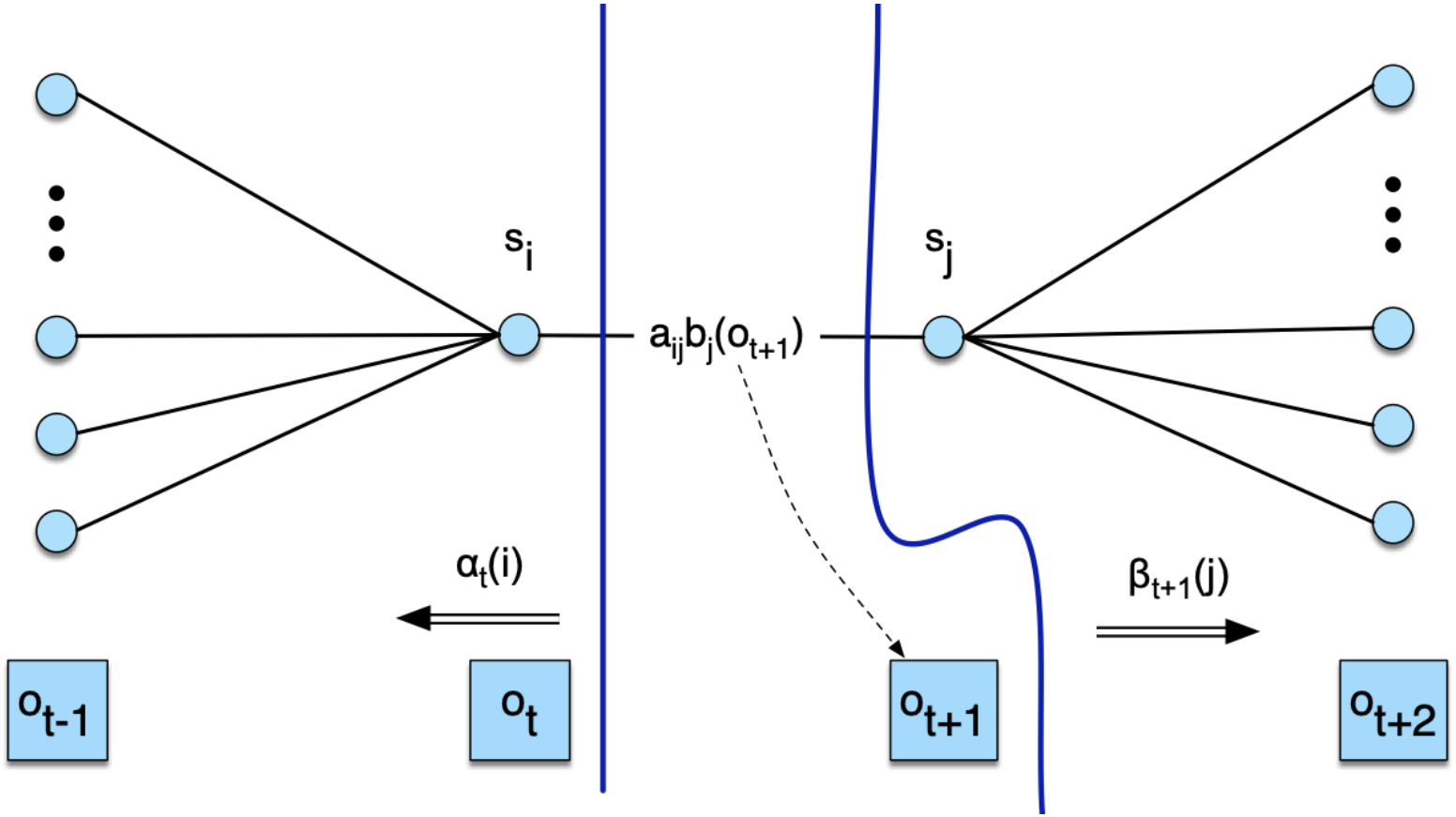
- Then we can simply compute ξ_t as follows

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda) = \frac{P(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)}$$

$$= \frac{\text{not-quite-}\xi_t(i,j)}{P(O|\lambda)}$$

Estimating the transition probabilities a_{ij} (3)

- How to compute $\text{not-quite-}\xi_t(i,j) = P(q_t = i, q_{t+1} = j, O|\lambda)$



$$\text{not-quite-}\xi_t(i,j) = \alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

Estimating the transition probabilities a_{ij} (4)

- Now we can compute ξ_t as follows

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda) = \frac{\text{not-quite-}\xi_t(i,j)}{P(O|\lambda)} = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

- Recall that we want to estimate $\hat{a}_{ij}$ using the following intuition

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

- Then

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat

Estimating the observation probability (1)

- We will compute the the probability of a given symbol v_k from the observation vocabulary V , given a state j: $\hat{b}_j(v_k)$ as follows

$$\hat{b}_j(v_k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j}$$

- For this, we will need to know the probability of being in state j at time t, which we will call $\gamma_t(j)$

$$\gamma_t(j) = P(q_t = j|O, \lambda)$$

- Once again, we will compute this by including the observation sequence in the probability:

$$\gamma_t(j) = \frac{P(q_t = j, O|\lambda)}{P(O|\lambda)}$$

Estimating the observation probability (2)

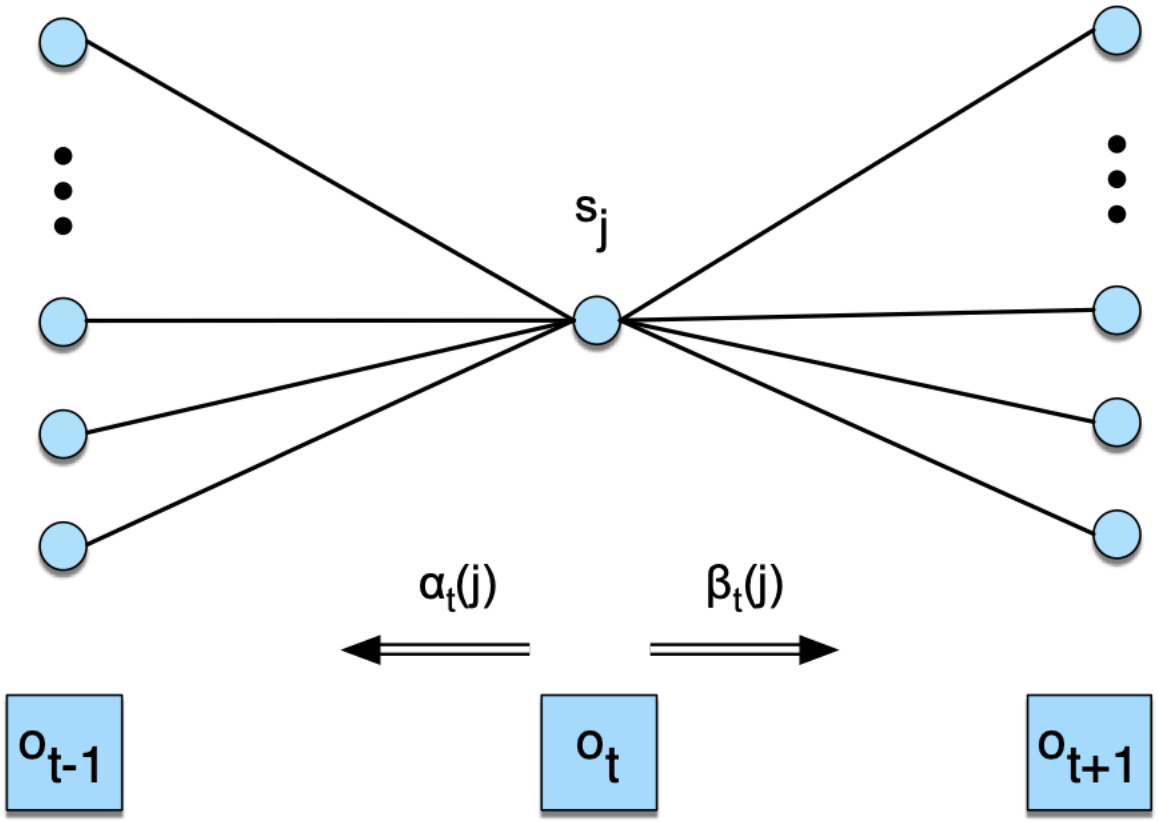
$$\gamma_t(j) = \frac{P(q_t = j, O|\lambda)}{P(O|\lambda)}$$

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{P(O|\lambda)}$$

And the observation probability can be computed as follows

$$\hat{b}_j(v_k) = \frac{\sum_{t=1\ s.t.O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Computing $P(q_t = j, O|\lambda)$



The forward-backward algorithm

function FORWARD-BACKWARD(*observations* of len T, *output vocabulary* V, *hidden state set* Q) **returns** *HMM=(A,B)*

initialize A and B

iterate until convergence

E-step

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(q_F)} \quad \forall\, t \text{ and } j$$

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\alpha_T(q_F)} \quad \forall\, t,\ i, \text{ and } j$$

M-step

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t.\ O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

return A, B

STUDENTS-HUB.com

Uploaded By: Jibreel Bornat