Multimodal Speech Recognition with Hidden Markov Models

ECE417 – Multimedia Signal Processing

Machine Problem 5,Fall 2017

Objective

- Develop a **bi-modal speech recognizer** using a **Hidden Markov Model** (HMM).
- Bi-modal means, in this case, that our data includes both **speech** and **lip tracking (visual)** features.
- The **feature extraction is already done** for you.
- You will:
  - **Train HMMs** for the spoken digits ‘2’ and ‘5’
  - **Use HMMs** to do **maximum likelihood estimation, 3 ways**
    - **Audio**
    - **Visual**
    - **Audio-Visual** (concatenate the audio and visual features)

The Data

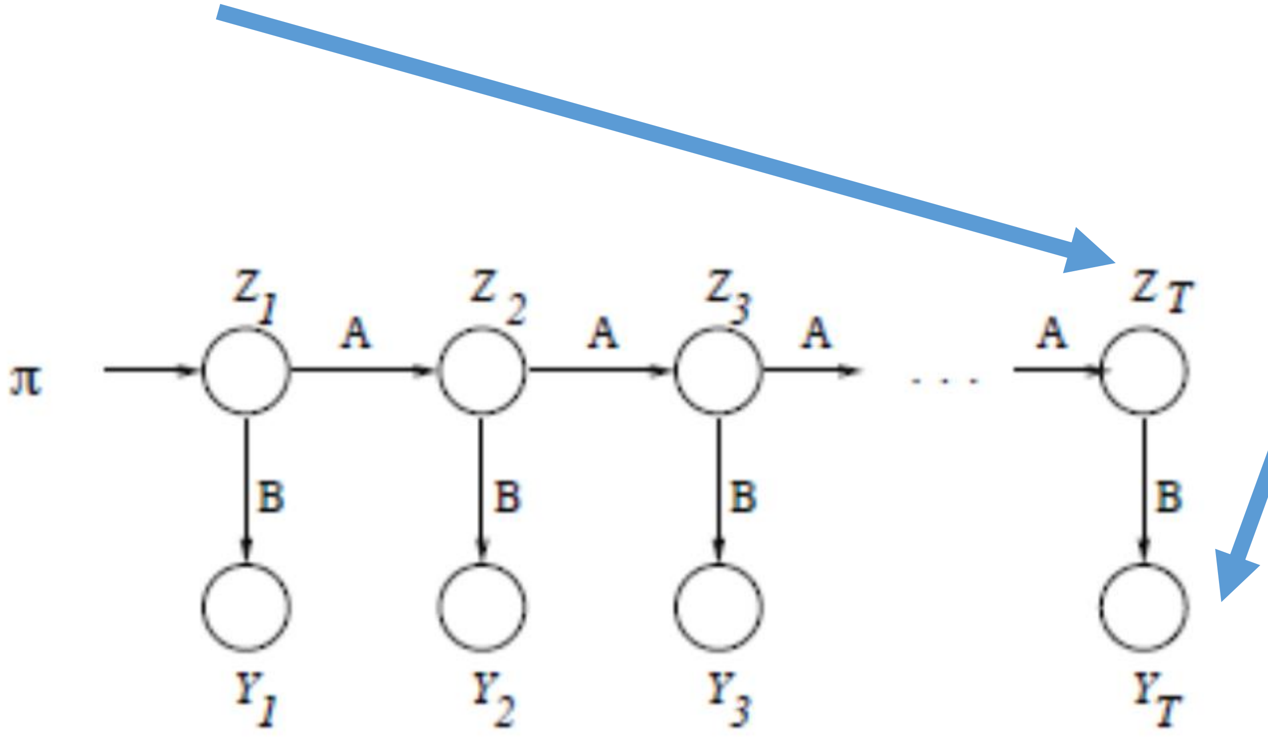
- **Audio data**: 24-dimensional **MFCCs**
- **Visual Data**: [w, h1, h2]
    - w: the lip width, **distance between mouth corners**
    - h1, h2: the **distance between the upper/lower lips and the line connecting the mouth corners**
- Files are named: [m].[n].[a|v].fea
    - m: the **class label**, either ‘2’ or ‘5’
    - n: the **index of the instance** (audio and visual instances at the same index go together, e.g., 2.1.a.fea, 2.1.v.fea)
    - a-> **audio** data,   v -> **video** data
- Each file has approximately **55-75 ordered frames**

Three HMM Problems to Solve

- **Given the training data** (one or more sequences), **find the best HMM**
  - **EM** Algorithm: Code provided for this
- **Given an HMM and a test sequence**, calculate the **likelihood**
  - **Forward-Backward** algorithm: You write
- **Given an HMM and a sequence**, find the most likely **hidden state sequence**
  - **Viterbi** algorithm: Bonus points

Hidden Markov Model (HMM) - 1

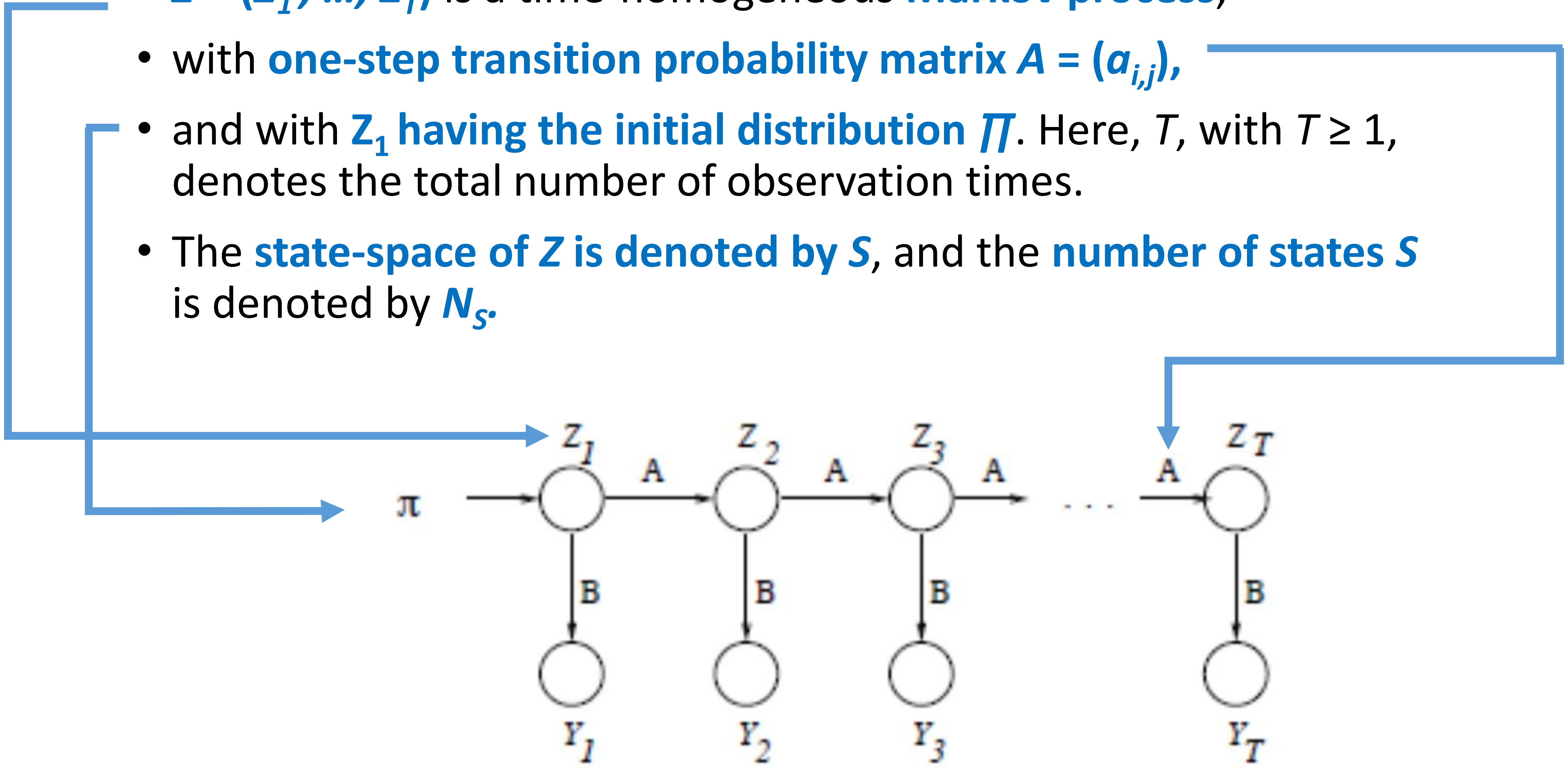
- ***X* = (*Y*,*Z*)**, where ***Z* is the unobserved** data and ***Y* is the observed** data



Structure of a Hidden Markov Model

Hidden Markov Model (HMM) - 2

- ***$Z = (Z_1, \dots, Z_T)$*** is a time-homogeneous **Markov process**,
- with **one-step transition probability matrix *$A = (a_{i,j})$*,**
- and with **Z_1 having the initial distribution *Π***. Here, *T*, with *$T \geq 1$*, denotes the total number of observation times.
- The **state-space of *Z* is denoted by *S***, and the **number of states *S*** is denoted by ***N_S*.**

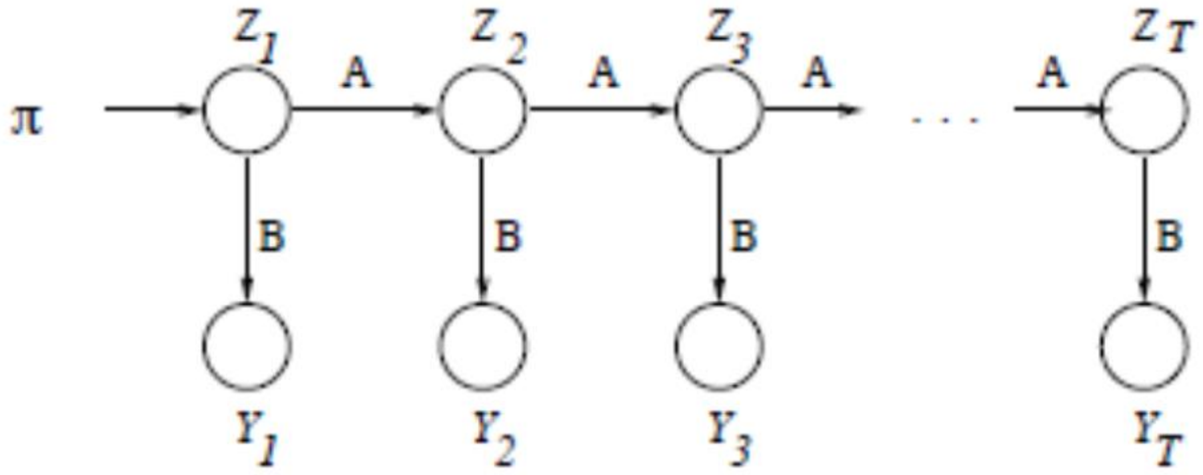


Hidden Markov Model (HMM)

- ***$Y=(Y_1, \ldots, Y_T)$* is the observed data**.

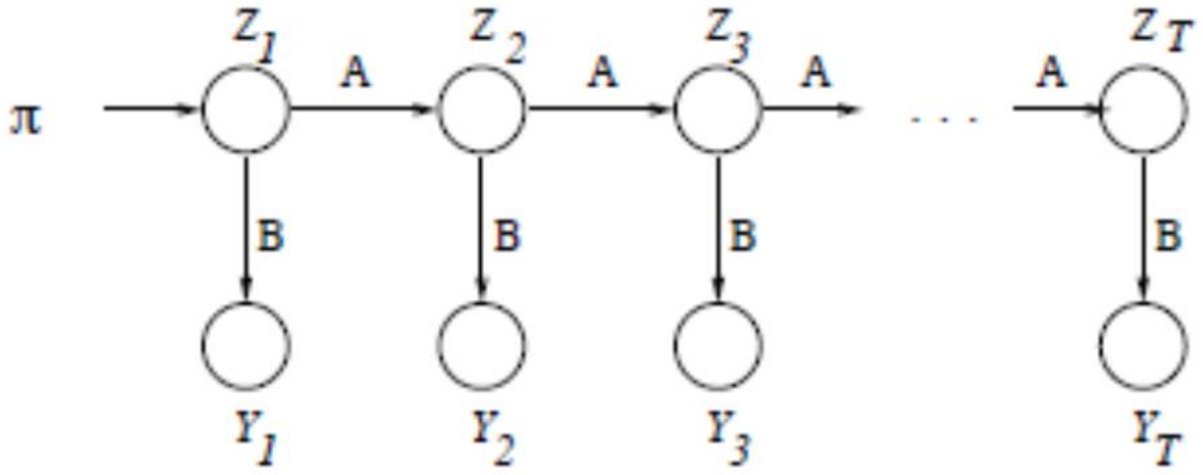
- It is such that **given *$Z=z$***, *for some $z = (z_1, \ldots, z_T)$*, the variables ***$Y_1, \ldots, Y_n$* are conditionally independent with *$P(Y_t = l \mid Z = z) = b_{z_t, l}$***, for a given **observation generation matrix $B = (b_{i,l})$.**

- The observations are assumed to take values in a size N_o, so that **B is an $N_s \times N_o$ matrix**, and **each row of B is a probability vector**.



Hidden Markov Model (HMM)

- The parameter for this model is ***λ = (Π, A, B).***



Training an HMM (find the best HMM)

[P0, A, mu, sigma] = ghmm_learn(data, N, Ainit)

data: Cell array of training matrices. Each entry in the cell array is a T*M array. T is length of sequence. M is the dimension of the observation (feature) vector.

N: The number of states

Ainit: The initial transition probability matrix

P0: Initial state probability matrix

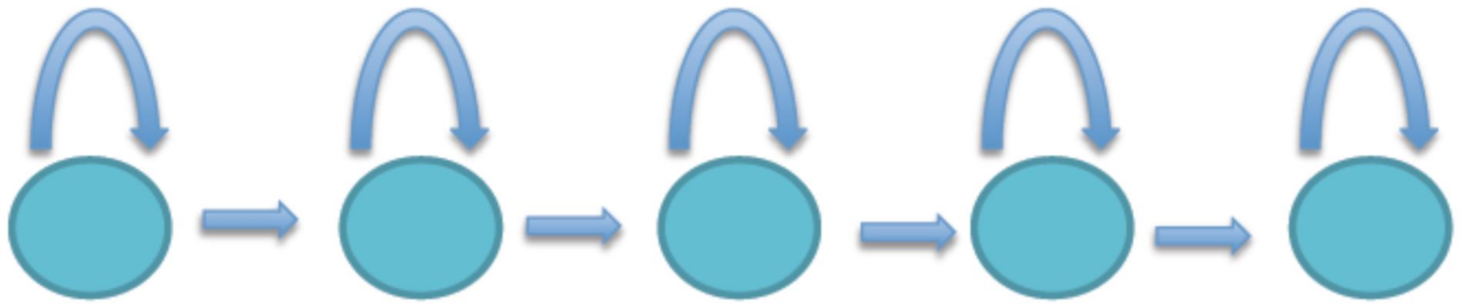
A: Transition Probability matrix

mu: mean matrix

sigma: covariance matrix

Setting Initial Transition Probabilities (A_init)

- Specify A_init to force desired flow through states
  (In EM updates, zero entries in A will remain zero)



0.8	0.2	0	0	0
0	0.8	0.2	0	0
0	0	0.8	0.2	0
0	0	0	0.8	0.2
0	0	0	0	1

Calculate the Likelihood of a Sequence

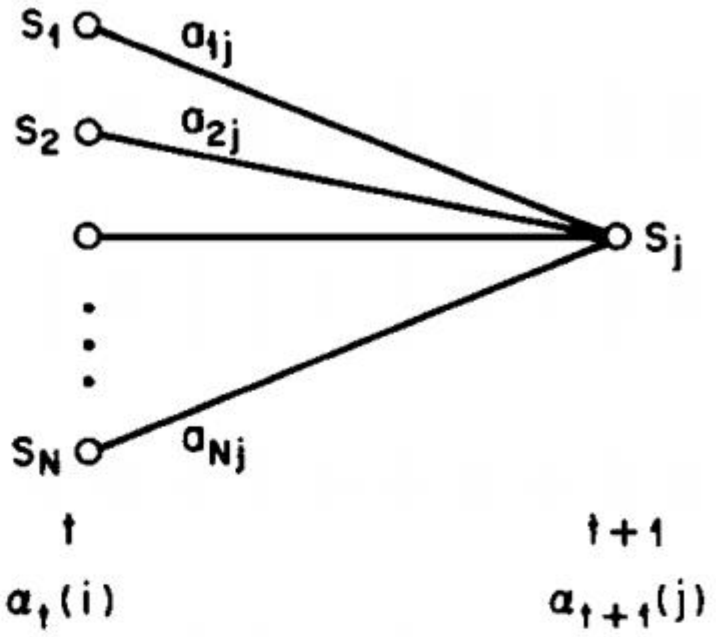
- Given the HMM and a sequence, calculate the likelihood that the sequence belongs to this HMM (ie, **calculate the likelihood of a sequence, given the model parameters**).
- The most straightforward way of calculating P(O|λ), where O is the observation sequence, and λ is the model, is computationally infeasible.
    - Given N states, and a sequence of T length, this direct calculation involves on the **order** of ***$2T * N^T$* calculations**!
- **Use the forward-backward algorithm instead** to compute P(O|λ)
    - It's computation is on the **order** of ***N^2T*** computations

The Forward Component

- Define the forward variable $\alpha_t(i)$ as the **probability of the partial observation sequence** $O_1, O_2, \ldots O_t$ (until time t), and state S_i at time t, given the model λ.
- $\alpha_t (i) = P(O_1 O_2 \ldots O_t, q_t = S_i|\lambda)$
- Initialize: $\alpha_1 (i) = P(O_1, q_1 = S_i|\lambda) = \pi_i B_i (O_1)$
- Use Recursive Formula to compute components at instant ‘t+1’ from instant ‘t’.

The Forward Component

- $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N_s} \alpha_t(i) A_{ij}\right] B_j(O_{t+1})$ $\Leftarrow$ Recursive Relation



$1 \le i \le Ns$

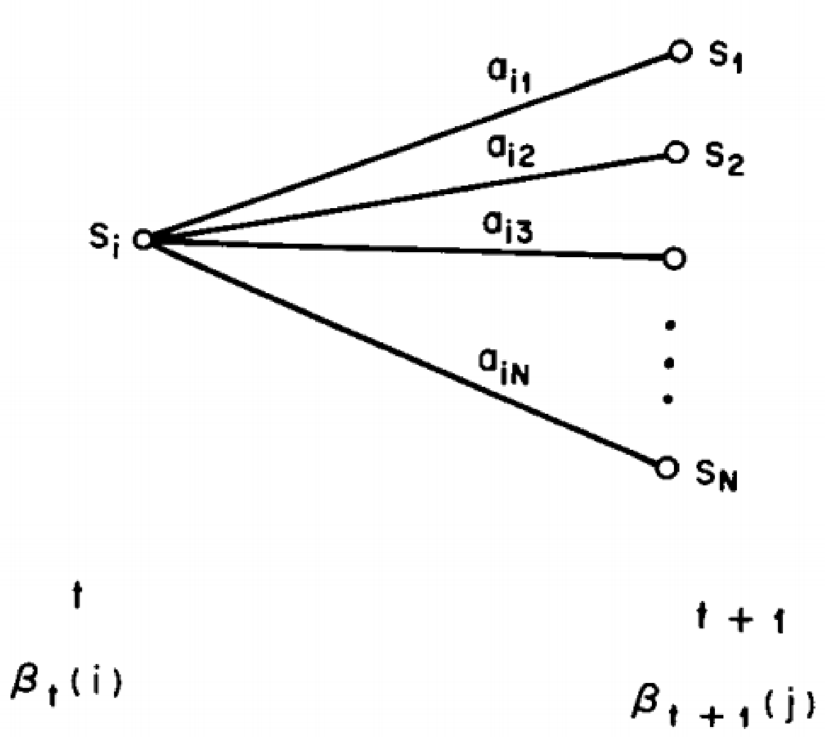
$1 \le t \le T-1$

The Backward Component

- Define the backward variable **$\beta_t(i)$ as the probability of the partial observation sequence from t+1 to the end**, given state S_i at time *t*, and model λ.
- $\beta_t(i) = P(O_{t+1}\ O_{t+2} \ldots O_T \mid q_t=S_i, \lambda)$
- Initialize: $\beta_T(i) = 1$
- Use Recursive Formula to compute components at instant ‘t’ from instant ‘t+1’.

The Backward Component

- $\beta_t(j) = \left[\sum_{i=1}^{N_s} \beta_{t+1}(i) A_{ji}\, B_i(O_{t+1})\right]$ $\Leftarrow$ Recursive Relation



1≤ j≤ Ns

1≤ t ≤T -1

Forward-Backward Algorithm Hints - 1

- (The forward-backward algorithm) The α's can be recursively computed forward in time, and the β's can be recursively computed backward in time, using:
- We want to compute $P(O|\lambda)$ to draw inference from a trained HMM.
- Using either the forward or the backward component we can do so.

Forward-Backward Algorithm Hints - 2

- P(O|λ) can be found as:

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

- Similarly we can also use the backward component to compute the desired probability for us. We can use either to do so.
- Also as both the forward and backward keep on multiplying small numbers, for sake of computing efficiency we use a scaling coefficient

$$C_t = \frac{1}{\sum_{i=1}^{N_s} \alpha_t(i)}$$

- It can be shown that $\log\big(P(O|\lambda)\big) = -\sum_{t=1}^{T} \log(C_t)$

Evaluating the HMM: Leave One Out Scheme

- The data has 10 sequences for the “2” and 10 sequences for the “5”

- Iterate through each sequence in the data set:
  - Each time, **exclude one sequence** from the training set; and **reserve it for testing.**
  - **Eg: You pick say 2.2 as test sequence, then you train an HMM for digit 2 using remaining 9 data sequence and HMM for digit 5 using the other 10 data set for 5 and pass the test sequence to both the HMMs and draw inferences and compute accuracy.**
  - **Repeat this 20 times** so that each sequence is tested
  - Calculate the **average accuracy and show the results in tabular form.**

Find the Most Probable Hidden State Sequence

- Given the observed data and an HMM model, find the most likely sequence of states.
- Use the **Viterbi algorithm** to solve this.

Viterbi Algorithm Overview

$$\begin{aligned}
\delta_j(t) &= \max_i \max_{\{z_1,\ldots,z_{t-2}\}} P(Z_1 = z_1, \ldots, Z_{t-2} = z_{t-2}, Z_{t-1} = i, Z_t = j, Y_1 = y_1, \cdots, Y_t = y_t|\theta) \\
&= \max_i \max_{\{z_1,\ldots,z_{t-2}\}} P(Z_1 = z_1, \ldots, Z_{t-2} = z_{t-2}, Z_{t-1} = i, Y_1 = y_1, \cdots, Y_{t-1} = y_{t-1}|\theta) a_{i,j} b_{jy_t} \\
&= \max_i \{\delta_i(t-1) a_{i,j} b_{jy_t}\}
\end{aligned}$$

(Viterbi algorithm) Compute the δ's and associated back pointers by a recursion forward in time:

$$\begin{aligned}
&\textit{(initial condition)} & \delta_i(1) &= \pi(i) b_{iy_1} \\
&\textit{(recursive step)} & \delta_j(t) &= \max_i \{\delta_i(t-1) a_{ij} b_{j,y_t}\} \\
&\textit{(storage of back pointers)} & \phi_j(t) &\triangleq \arg\max_i \{\delta_i(t-1) a_{i,j} b_{j,y_t}\}
\end{aligned}$$

Then $z^* = \widehat{Z}_{MAP}(y, \theta)$ satisfies $p_{cd}(y, z^*|\theta) = \max_i \delta_i(T)$, and z^* is given by tracing backward in time:

$$z_T^* = \arg\max_i \delta_i(T) \quad \textit{and} \quad z_{t-1}^* = \phi_{z_t^*}(t) \ \textit{for}\ 2 \le t \le T.$$

Matlab Functions of Interest

- Train HMM: ghmm_learn
  - **[P0,A,mu,sigma] = ghmm_learn(data, N, Ainit)**
- Forward algorithm gmhmm_fwd.m
  - **[alpha, scale] = gmihmm_fwd(Y,A,P0,mu,sigma)**
- Backward algorithm ghmm_bwd.m
  - **[beta] = gmihmm_bwd(Y, A, P0, mu, sigma, scale)**

Turn In

- Your write up, with the accuracy of the HMM for 3 cases (A, V, AV) corresponding to each digit (2 and 5).
- All your MATLAB codes. I will be running run.m and it should display the accuracies of HMM for the three cases and both digits as the output. Upload to Compass in the usual way you have been doing.
- Suggested Reading: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition' by Lawrence R. Rabiner

Sample Accuracy Table

Digits	Audio Recognition	Video Recognition	AV Recognition
2			
5			
Overall (Average of both cases)			

Eg: Accuracy for digit 2 is how many instances of 2 (out of 10) were more likely identified by the HMM trained by data set for digit 2 than data set trained by 5.