CHI
CES Engineering

SIMULATION SYSTEMS

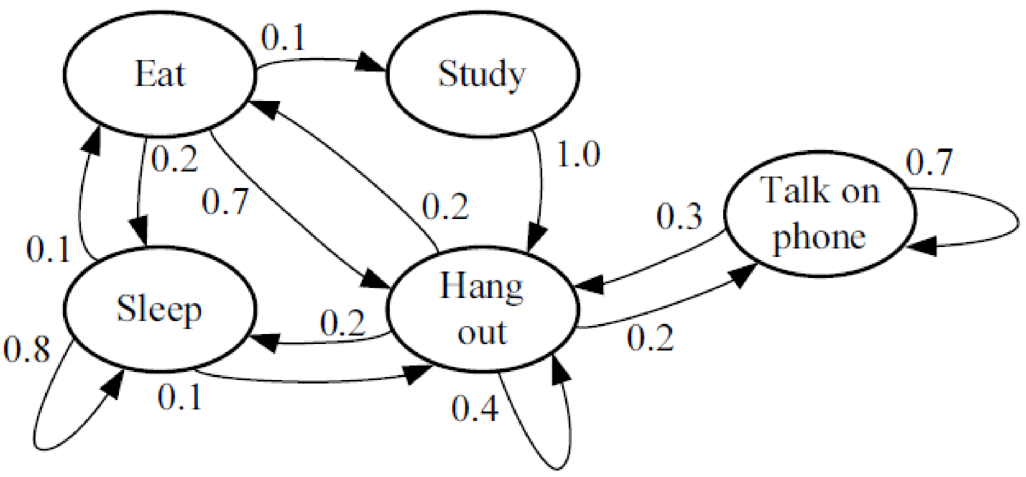
HIDDEN MARKOV MODELS

Prof. Yasser Mostafa Kadah

Markov Models

- Standard Markov model is defined by three elements:
  - A set of states $Q = \{q_1, q_2, \ldots, q_n\}$
  - An initial state distribution $\Pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$
  - A set of transition probabilities

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix}$$



Hidden Markov Models (HMM)

- HMM is called "hidden" because we usually assume that we do not see the states of the model, but rather a set of outputs influenced by them
- To make an HMM, we extend our standard Markov model with the following two features:
  - A set of possible outputs $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$
  - A set of output probabilities

$$B = \begin{bmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{bmatrix}$$

b_{ij} is the probability of emitting output j in state i

HMM

- Consider HMM example defined as:

$$Q = \{q_1, q_2, q_3\}$$

$$\Pi = \{\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\}$$

$$P = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

$$\Sigma = \{a, b, c\}$$

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0.1 & 0.8 & 0.1 \\ 0.2 & 0.2 & 0.6 \end{bmatrix}$$

State Transitions

$$q_1 \rightarrow q_2 \rightarrow q_1 \rightarrow q_3 \rightarrow q_2 \rightarrow q_3 \rightarrow q_2 \rightarrow q_3$$

Simultaneously

Output Observations

a b a c c a b c

HMM

- In HMM, we don't get to observe the state of the model directly. Rather, whenever it is in any state i (one of M states), it emits a *symbol* k, chosen probabilistically from a set of K symbols.
- The probability of emitting symbol number *k* from state number *i* is denoted by

$$b_i(k) \equiv P(\text{symbol } k \mid \text{state } i) \qquad (0 \leq i < M, \quad 0 \leq k < K)$$

$$\sum_{k=0}^{K-1} b_i(k) = 1 \qquad (0 \leq i < M)$$

HMM

- When the model evolves through N time steps, the *hidden states* are a vector of integers,

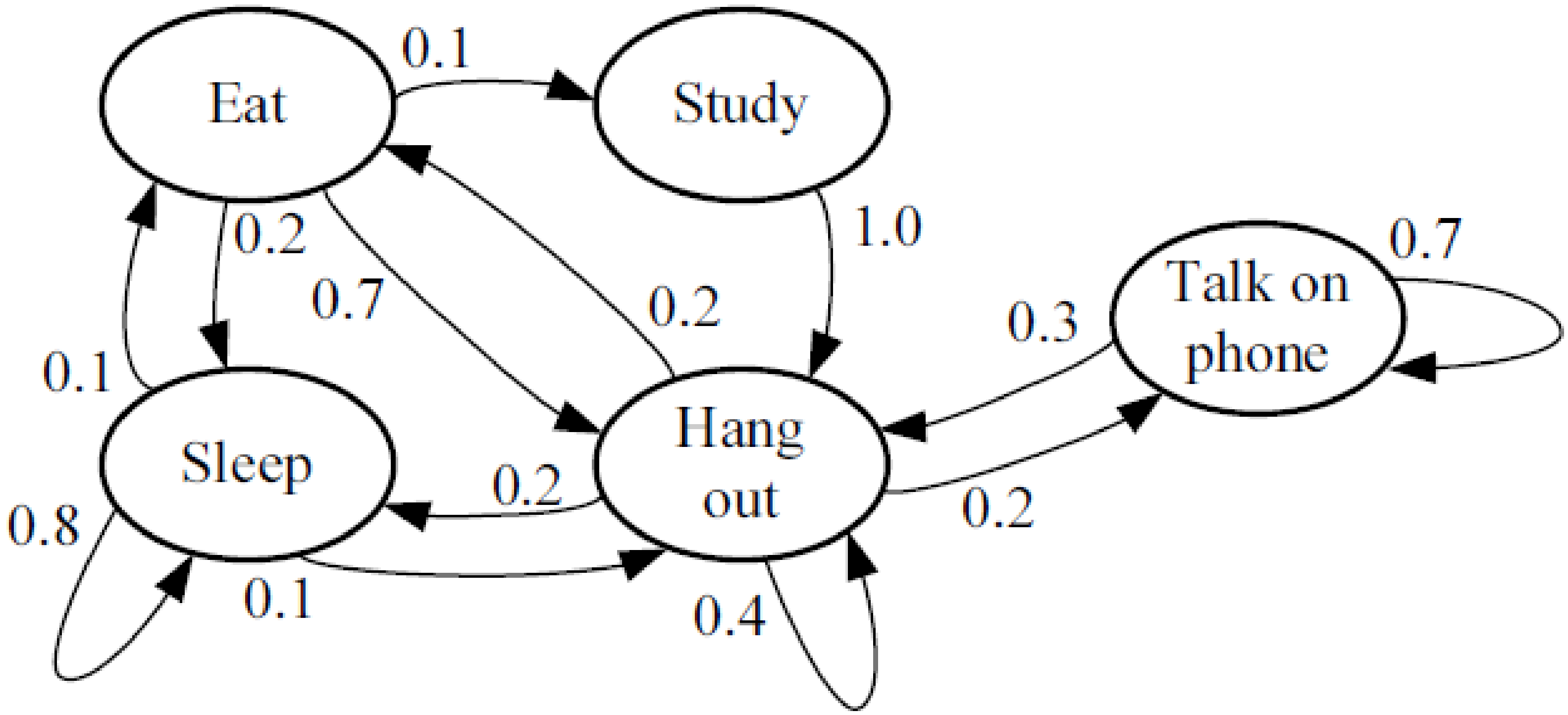
$$\mathbf{s} = \{s_t\} = (s_0, s_1, \ldots s_{N-1}) \qquad 0 \leq s_i < M$$

while the *observations* or *data* are a vector of integers,

$$\mathbf{y} = \{y_t\} = (y_0, y_1, \ldots y_{N-1}) \qquad 0 \leq y_i < K$$

HMM

- Example: Teen life Markov model
  - States: Eat, Study, Sleep, Talk, Hang out
  - Output is answer to parent query, "What are you doing?"



HMM

- Teen Life example: Table of symbols and their probabilities of being emitted from each state
  - response to repeated query, “What are you doing?”

			$i = 0$	1	2	3	4
k	symbol	meaning	Eat	Hang	Study	Talk	Sleep
0	o	[silence]	0.2	0.2	0	0.3	0.5
1	s	“I’m studying!”	0	0	1.0	0.2	0
2	b	“I’m busy!”	0	0.6	0	0.4	0
3	g	[grunt]	0.8	0.2	0	0.1	0
4	z	[snore]	0	0	0	0	0.5

key point is that the emitted symbols give only incomplete, or garbled, state information

HMM

- A state can emit more than one possible symbol, and a symbol can be emitted by more than one possible state.
- Problems solved by HMM methods
  - Problem 1: Given an observed set of outputs x and HMM find the best estimate of hidden state string S to produce x.
  - Problem 2: Given x and HMM, find the probability of generating x from HMM.
    - Useful for evaluating different possible models as the source of x
  - Problem 3: Given the observations x and the geometry (graph structure) of HMM, find the parameters of HMM that maximize the probability of producing x from *HMM*
    - i.e., the maximum likelihood parameter set for generating x

HMM Problem 1: Optimizing State Assignments

- If we have a sequence of observations x=x1; x2; . . . ; x_T and an HMM and we want to find the best sequence of states S=S1; S2; . . . ; ST to explain x
- What is meant by "optimizing" is to ask for the complete state set S maximizing the total likelihood of the outputs and states, given the HMM

$$\max_S \ Pr\{x, S \mid \lambda\}$$

$$Pr\{x, S \mid \lambda\} = Pr\{x \mid S, \lambda\} \ Pr\{S \mid \lambda\}$$

Easily computed for any given S

$$\prod_{i=1}^{T} b_{S_i, x_i}$$

$$\pi_{S_1} \prod_{i=2}^{T} p_{S_{i-1}, S_i}$$

HMM Problem 1: Optimizing State Assignments

- Choosing S to optimize for these probabilities is not so simple, but it turns out that we can do it efficiently by a dynamic programming algorithm called the Viterbi algorithm
  - Variational calculus not regular calculus: trying to find optimal “function” rather than optimal point

HMM Problem 2: Evaluating Output Probability

- Measure the goodness of fit of a given model for a given output sequence
  - Maximize probability that the output is produced over all possible models

$$\max_{\lambda_i} \; Pr\{x \mid \lambda_i\} \; Pr\{\lambda_i\}$$

- This probability of the output, given the model, will be computed by summing over all possible assignments of states to the outputs

HMM Problem 2: Evaluating Output Probability

- If runtime is not an issue, then problem is easy to solve
  - Just enumerate over all possible sequences of states S and add up the likelihoods over all of these state sequences:

$$\sum_S Pr\{x, S \mid \lambda\} = \sum_S Pr\{x \mid S, \lambda\}\ Pr\{S \mid \lambda\}$$

$$= \sum_S (b_{S_1 x_1} \times b_{S_2 x_2} \times \cdots \times b_{S_T x_T})(\pi_{S_1} \times p_{S_1 S_2} \times p_{S_2 S_3} \times \cdots \times p_{S_{T-1} S_T}).$$

- It is difficult in practice to enumerate all possible sequences: N^T possible sequences
- A much more efficient method is forward method or forward-backward method
  - Dynamic programming like Viterbi algorithm

Problem 3: Training the Model

- Inferring the HMM parameters from a set of training data
- Training the model can be relatively easy if we have labeled training data
  - Data in which we know the true assignment of states
- Done using Expectation Maximization (EM) based methods

Assignments

- Find software packages for HMM and try them
- Provide an overview of different applications of HMM problems