HIDDEN MARKOV MODELS(HMMs)

120060027
SAMET TONYALI

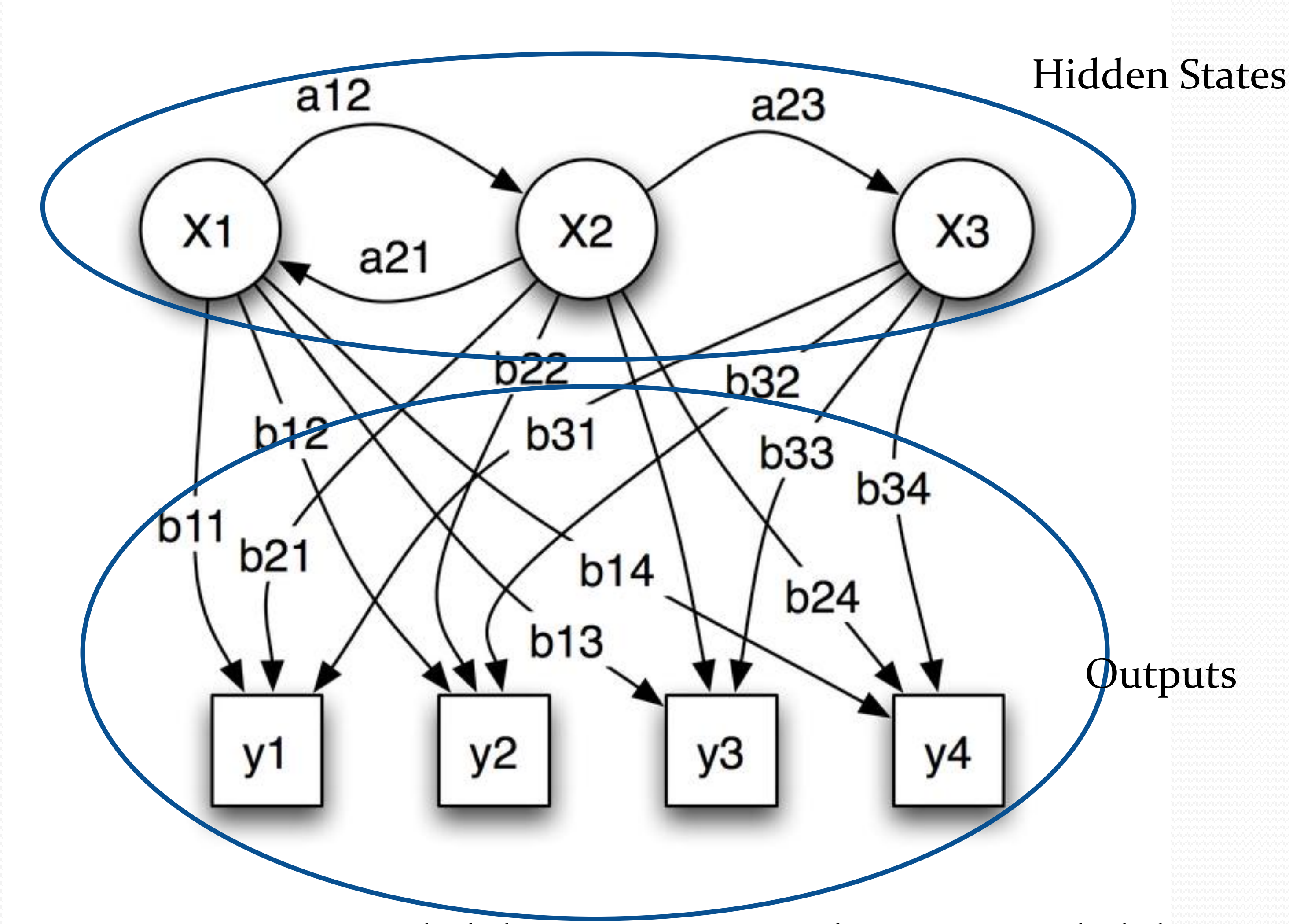
What is an HMM?

- An HMM is a statistical Markov Model in which the system being modeled is assumed to be a Markov process with unobserved(hidden) states.
- We know only outputs of process sequence, but not the states.

What is difference from regular MM?

- In a regular Markov Model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters.
- In an HMM, the state is not directly visible, but output which is dependent on the state, is visible.

An Illustrative Figure



X: state **a:** transition probability **y:** output **b:** output probabilities

Parameters of an HMM

- The parameters of an HMM are of two types:
  - Transition Probabilities
  - Emission Probabilities(Output Probabilities)

A Simple Example

- Alice and Bob live apart from each other and talk together daily over telephone about what did they do that day.
- Bob is only interested in three activities:
  - Walking in the park
  - Shopping
  - Cleaning his apartment
- His activities are dependent on the weather on a given day.


OH ALICE... YOU'RE THE ONE FOR ME
BUT BOB... IN A QUANTUM WORLD HOW CAN WE BE SURE?
Ψ^+ or Ψ^- ?

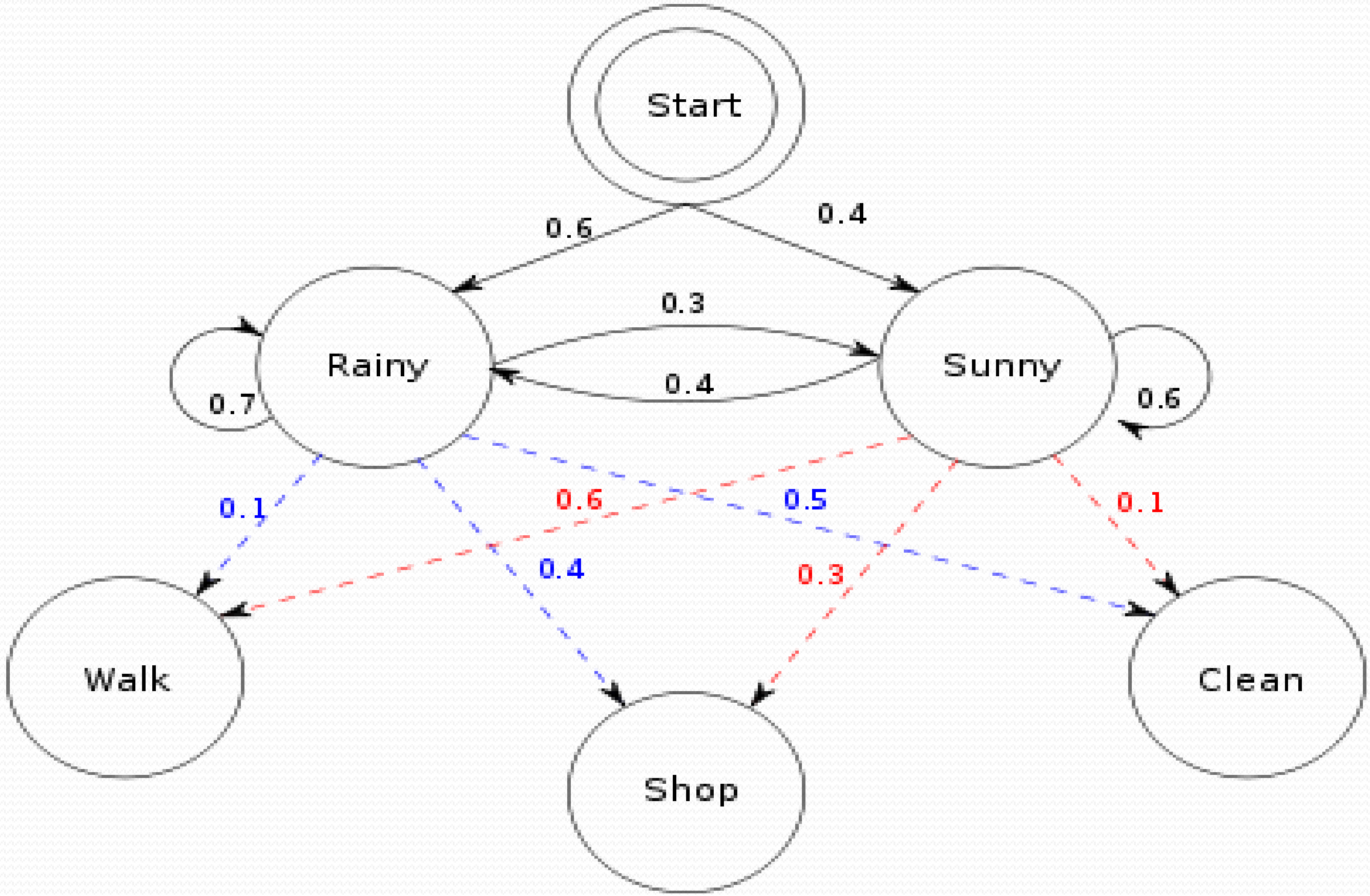
- Alice has no knowledge about how the weather is in where Bob lives. She tries to guess what the weather must have been like.
- There are two states: “Rainy” or “Sunny”, but Alice cannot observe them directly, that is, they are hidden from her.
- Since Bob tells Alice about his activities, those are the observations. The entire system is that of an HMM.

- Alice knows the general weather trends in the area, and what Bob likes to do on average. In other words, the parameters of the HMM are known.
- They can be written down in the Python programming language:

- states = ('Rainy', 'Sunny')
- observations = ('walk', 'shop', 'clean')
- start_probability = {'Rainy': 0.6, 'Sunny': 0.4}
- transition_probability= {
  - 'Rainy' : {'Rainy': 0.7,'Sunny':0.3},
  - 'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
- }
- emission_probability = {
  - 'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
  - 'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},
- }

- **start_probability:** Alice's belief about which state the HMM is in when Bob first calls her.
- **transition_probability:** The change of the weather in the underlying Markov chain.
- **emission_probability:** How likely Bob to perform a certain activity on each day.

Probability Distribution



Three Basic Problems of HMMs

- Given a model, we would like to evaluate the probability of any given observation sequence, $O = \{O_1 O_2 \dots O_T\}$
- Given a model and an observation sequence O, we would like to find out state sequence $Q = \{q_1\ q_2\ \dots q_T\}$, which has the highest probability of generating O, namely, we want to find Q^* (Optimal result).
- Given a training set of observation sequences, $X = \{O^k\}_k$, we would like to learn the model that maximizes the probability of generating X.

Viterbi Algorithm

- The Viterbi algorithm is a dynamic programming algorithm for finding the ***most*** likely sequence of hidden states – called the Viterbi path – that results in a sequence of observed events, generally in HMMs.

- The algorithm makes a number of assumptions:
  - Both the observed events and hidden states must be in a sequence. This sequence often corresponds to time.
  - These two sequences need to be aligned, and an instance of an observed event needs to correspond to exactly one instance of a hidden state.
  - Computing the most likely hidden sequence up to a certain point ***t,*** and the most likely sequence at point ***t-1***

These assumptions are all satisfied in a first-order ***hidden*** Markov model.

Algorithm

- Suppose we are given a Hidden Markov Model (HMM) with states Y, initial probabilities π_i of being in state i and transition probabilities $a_{i,j}$ of transitioning from state i to state j.
- Say we observe outputs $\boldsymbol{x_0}, \ldots, \boldsymbol{x_T}$.
- The state sequence $\boldsymbol{y_0}, \ldots, \boldsymbol{y_T}$ most likely to have produced the observations is given by the recurrence relations:

$$
\begin{aligned}
V_{0,k} &= \mathrm{P}\big(x_0 \mid k\big) \cdot \pi_k \\
V_{t,k} &= \mathrm{P}\big(x_t \mid k\big) \cdot \max_{y \in Y} \left(a_{y,k} V_{t-1,y}\right)
\end{aligned}
$$

- Here $V_{t,k}$ is the probability of the most probable state sequence responsible for the first $t + 1$ observations (we add one because indexing started at 0) that has k as its final state.
- The Viterbi path can be retrieved by saving back pointers which remember which state y was used in the second equation.
- Let Ptr(k,t) be the function that returns the value of y used to compute $V_{t,k}$ if $t > 0$, or k if $t = 0$. Then:

$$y_T = \arg\max_{y \in Y} \left(V_{T,y} \right)$$

$$y_{t-1} = \mathrm{Ptr}\left(y_t, t \right)$$

Complexity

- The complexity of this algorithm is $O(T \times |Y|^2)$.

References

- http://en.wikipedia.org/wiki/Viterbi_algorithm
- http://en.wikipedia.org/wiki/Hidden_Markov_model
- Alpaydin, Ethem, Chapter 13 – Hidden Markov Models p.305-326

Questions?