SPRING 2023

CS 378: INTRO TO SPEECH AND AUDIO PROCESSING

Hidden Markov Models 1

DAVID HARWATH
Assistant Professor, UTCS

The University of Texas at Austin
Department of Computer Science
College of Natural Sciences

Today’s big questions

- How can we model time-series data in general?
  - One way: using Hidden Markov Models (HMMs)
- What is a Hidden Markov Model and how does it work?
- How are HMMs used in speech recognition systems?

Today’s agenda

- HMM motivation and intuitive introduction
- HMM mathematical formulation
- HMM algorithms
  - Scoring: Forward-Backward Algorithm
  - Decoding: Viterbi and Forward-Backward Algorithms
  - Training: Baum-Welch Algorithm
- HMMs for phone and word modeling in ASR

Modeling time series data

There are many problems in science and engineering that involve data in the form of a *time series*

- Speech waveforms
- Biometrics like heart rate
- Stock prices

Modeling time series data

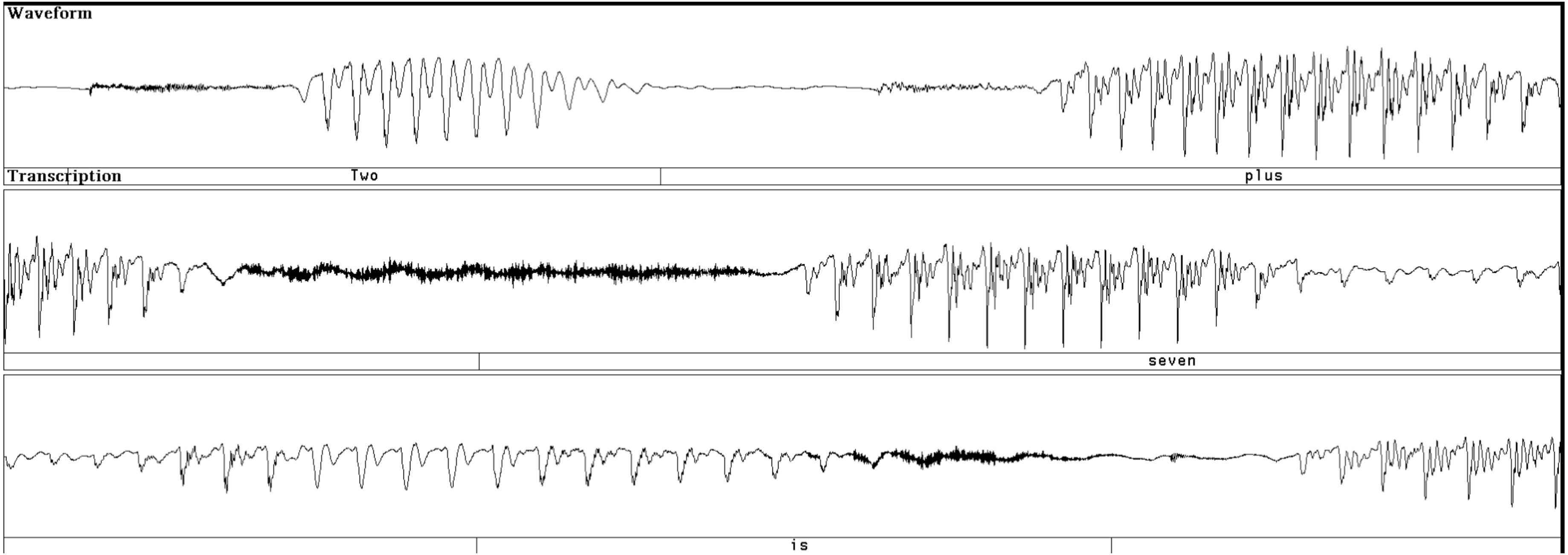
Generally, we have access to the time series data but are actually interested in inferring something about what *caused* the time series to appear the way it did

- Speech waveform: The words that were spoken
- Heart rate: A patient's stress or physical exertion level
- Stock price: A company's internal management decisions

This underlying cause is often unobserved, or *hidden.*

Words as a hidden cause of a waveform

Notice that the "hidden cause" *changes over time…*



Adapted from *James Glass, and Victor Zue. 6.345 Automatic Speech Recognition. Spring 2003. Massachusetts Institute of Technology: MIT OpenCourseWare, https://ocw.mit.edu. License: Creative Commons BY-NC-SA.*

Hidden Markov Models

- A *Hidden Markov Model:*
  - Assumes a *hidden* state sequence that causes a signal we get to observe
  - Allows the hidden state sequence to change over time under a first order *Markov* assumption (the next hidden state only depends on the current hidden state)
  - *Models* the relationship between the hidden state sequence and the observed signal

A very simple HMM example

- I have a bag with two quarters and one marble in it.

- Quarter #1 is "regular"
  - If I flip it, heads and tails have an equal probability (0.5) of coming up

- Quarter #2 is "loaded"
  - The quarter is biased in favor of heads. If I flip it, P(Heads) = 0.9

$P(Heads) = 0.9$ $\quad$ $P(Heads) = 0.5$

A very simple HMM example

We play the following game:

1. I draw one item from the bag, each with equal probability (1/3). I don't show you the item.
2. If the item is a quarter, I flip the quarter and record the outcome (H or T) on a piece of paper. Then I go back to step 1.
3. If the item is the marble, the game stops and I go to step 4.
4. I give you the piece of paper. Your job is to infer the sequence of items that I drew from the bag.

$P(Heads) = 0.9$ $\quad$ $P(Heads) = 0.5$

A very simple HMM example

The paper recording
the flip outcomes
(You get to see this)

H T H H H T

Q2, Q1, Q1, Q2, Q1, Q1, M

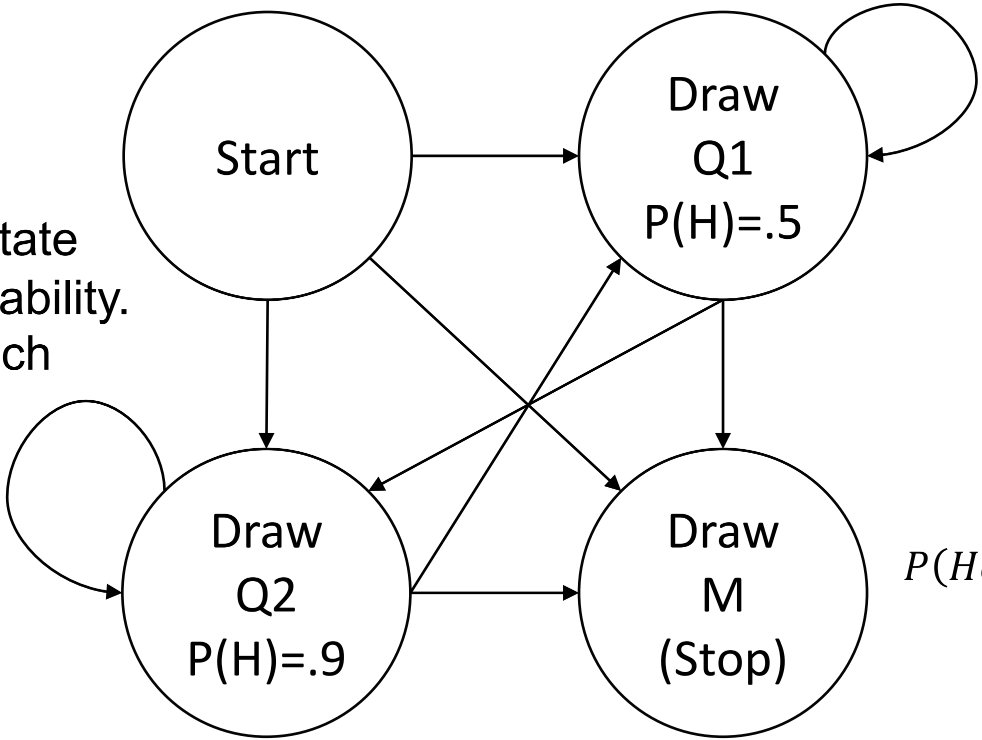
The underlying sequence of
objects I pulled from the bag
(You need to guess this)

$P(Heads) = 0.9$ $P(Heads) = 0.5$

A very simple HMM example

You decide to model the game with a probabilistic state machine:

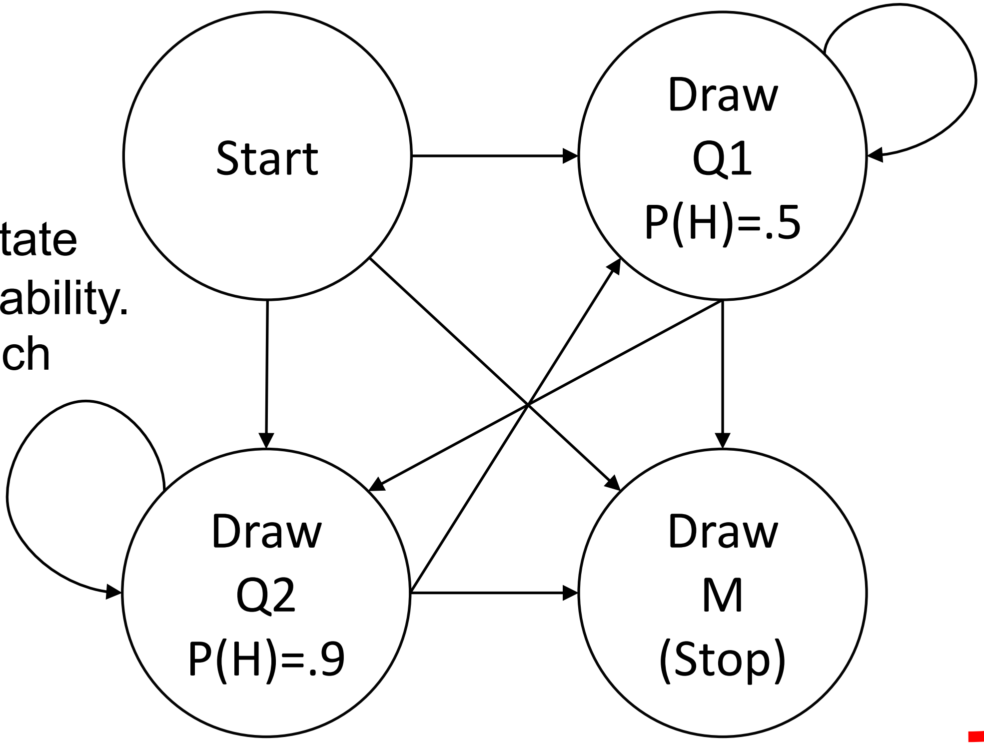
Every arrow represents a “state transition” probability. In this case, each one is just 1/3



$P(Heads) = 0.9$ $P(Heads) = 0.5$

A very simple HMM example

Every arrow represents a "state transition" probability. In this case, each one is just 1/3



This is a Hidden Markov Model!

Now we'll learn how to use it…

Today's agenda

- HMM motivation and intuitive introduction
- HMM mathematical formulation
- HMM algorithms
  - Scoring: Forward-Backward Algorithm
  - Decoding: Viterbi and Forward-Backward Algorithms
  - Training: Baum-Welch Algorithm
- HMMs for phone and word modeling in ASR

Parameterizing a discrete HMM

- $S = \{s_1, s_2, \dots, s_N\}$
  State set

  $q_t \in S$: The state at time t

- $X = \{x_1, x_2, \dots, x_M\}$
  Observation symbol set

  $o_t \in X$: The observation at time t

- $A \in \mathbb{R}^{N \times N}$
  State transition distribution

  $a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$

- $B \in \mathbb{R}^{N \times M}$
  State observation distribution

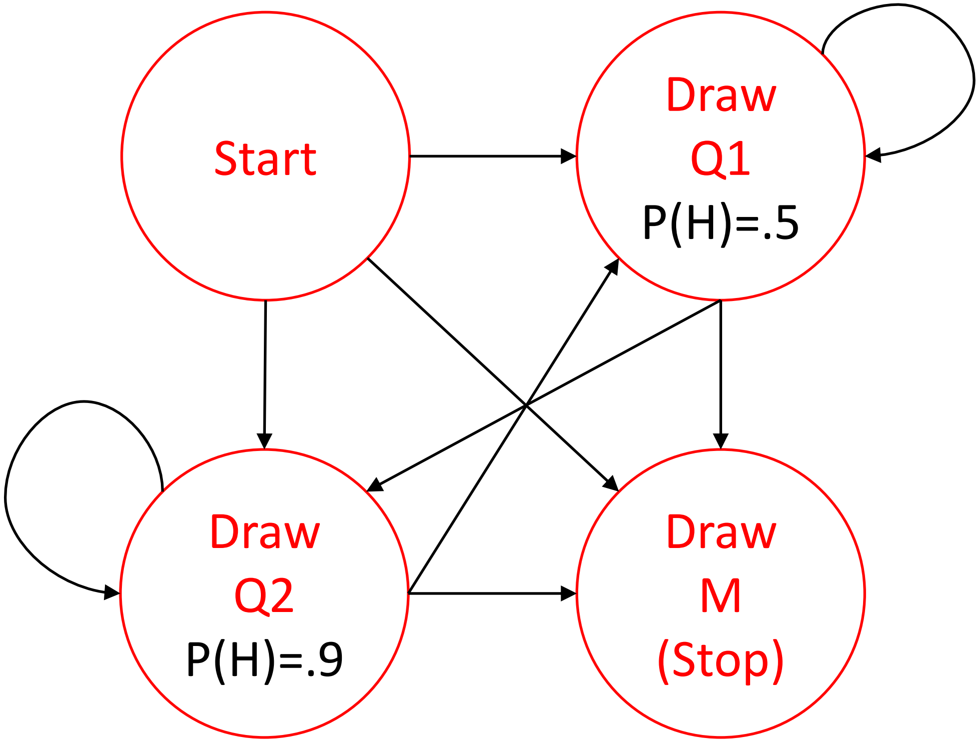
  $b_{ik} = P(o_t = x_k | q_t = s_i)$

- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$
  Initial state distribution

  $\pi_i = P(q_1 = s_i)$

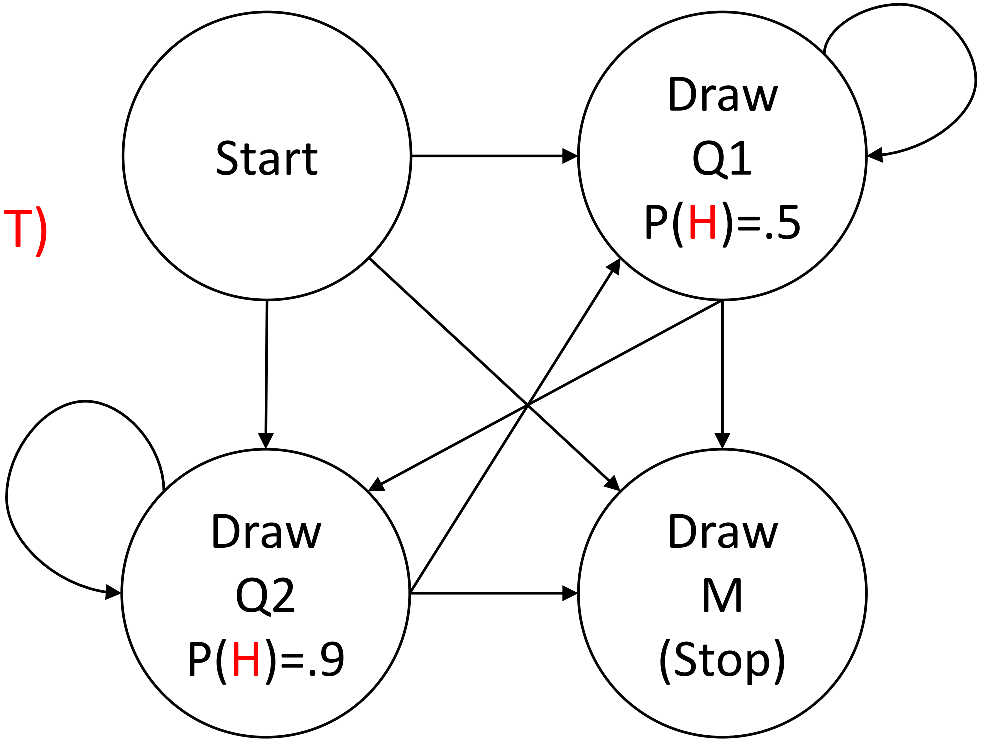
Parameterizing a discrete HMM

- $S = \{s_1, s_2, \dots, s_N\}$
  State set
- $X = \{x_1, x_2, \dots, x_M\}$
  Observation symbol set
- $A \in \mathbb{R}^{N \times N}$
  State transition distribution
- $B \in \mathbb{R}^{N \times M}$
  State observation distribution
- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$
  Initial state distribution



Parameterizing a discrete HMM

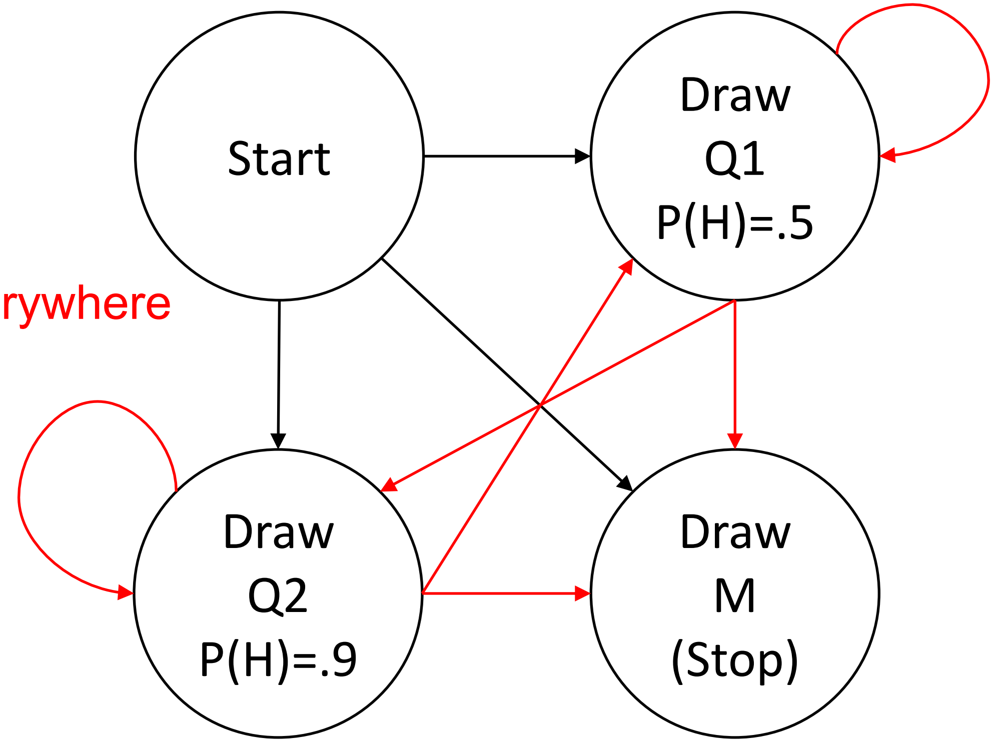
- $S = \{s_1, s_2, \dots, s_N\}$
  State set
- $X = \{x_1, x_2, \dots, x_M\}$
  Observation symbol set (H and T)
- $A \in \mathbb{R}^{N \times N}$
  State transition distribution
- $B \in \mathbb{R}^{N \times M}$
  State observation distribution
- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$
  Initial state distribution



Parameterizing a discrete HMM

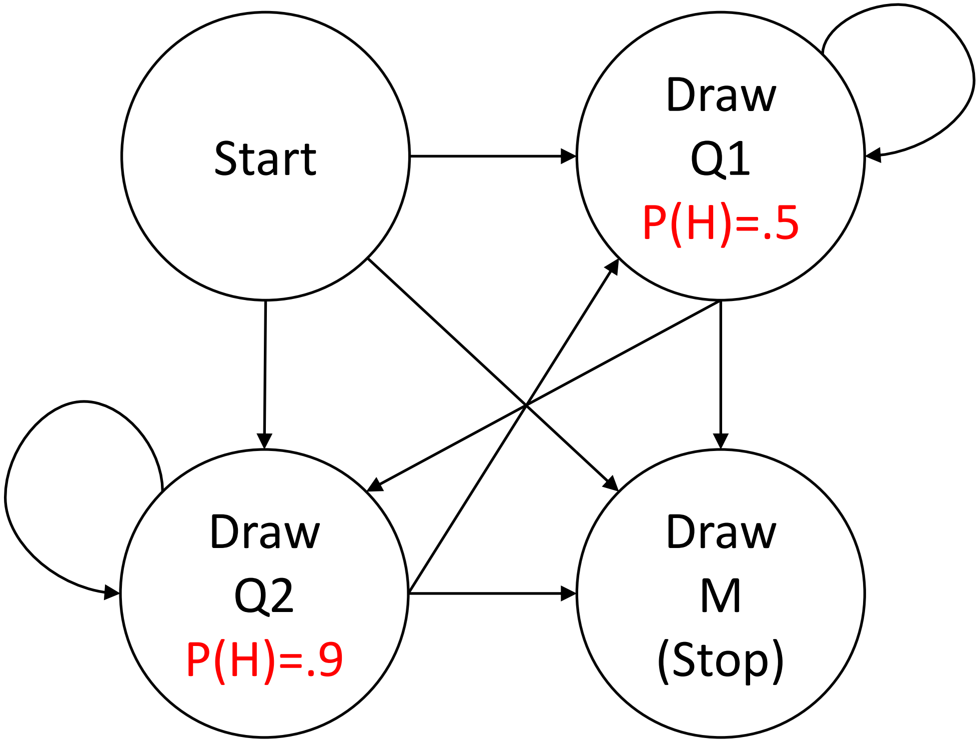
- $S = \{s_1, s_2, \dots, s_N\}$
  State set
- $X = \{x_1, x_2, \dots, x_M\}$
  Observation symbol set
- $A \in \mathbb{R}^{N \times N}$
  State transition distribution
- $B \in \mathbb{R}^{N \times M}$
  State observation distribution
- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$
  Initial state distribution

1/3 everywhere



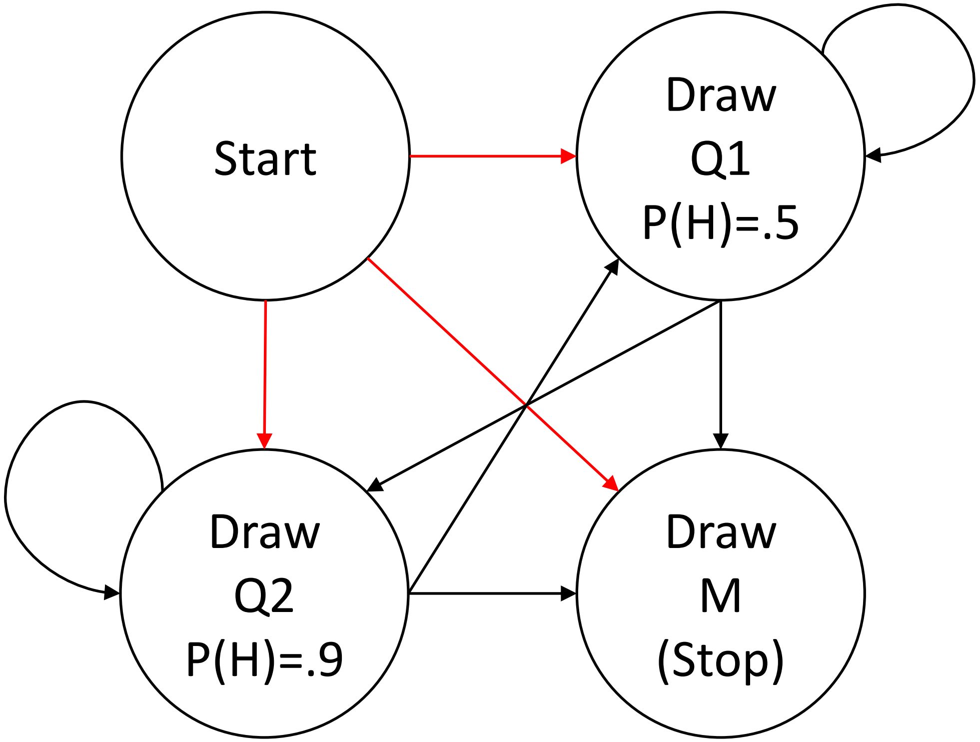
Parameterizing a discrete HMM

- $S = \{s_1, s_2, \ldots, s_N\}$
  State set
- $X = \{x_1, x_2, \ldots, x_M\}$
  Observation symbol set
- $A \in \mathbb{R}^{N \times N}$
  State transition distribution
- $B \in \mathbb{R}^{N \times M}$
  State observation distribution
- $\Pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$
  Initial state distribution



Parameterizing a discrete HMM

- $S = \{s_1, s_2, \dots, s_N\}$
  State set
- $X = \{x_1, x_2, \dots, x_M\}$
  Observation symbol set
- $A \in \mathbb{R}^{N \times N}$
  State transition distribution
- $B \in \mathbb{R}^{N \times M}$
  State observation distribution
- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$
  Initial state distribution



Parameterizing a discrete HMM

In practice, we define an HMM as $\lambda = \{A, B, \Pi\}$

- $A \in \mathbb{R}^{N \times N}$ $\quad a_{ij} = P(q_{t+1} = s_j | q_t = s_i)$

  State transition distribution

- $B \in \mathbb{R}^{N \times M}$ $\quad b_{ik} = P(o_t = x_k | q_t = s_i)$

  State observation distribution

- $\Pi = \{\pi_1, \pi_2, \dots, \pi_N\}$ $\quad \pi_i = P(q_1 = s_i)$

  Initial state distribution

How do you use an HMM?

There are 3 major HMM problems + algorithms

1. Given a parameterized HMM, how do you **compute the probability of a given observation sequence**?
   - Scoring Problem: Use the Forward (or Backward) Algorithm
2. Given a parameterized HMM, how do you **compute the optimal hidden state sequence** for a given observation sequence and some optimality criterion?
   - Decoding Problem: Use the Viterbi or Forward-Backward Algorithm
3. Given an HMM **with unknown parameters,** how do you **find the most likely parameters** given a collection of observation sequences?
   - Training Problem: Use the Baum-Welch Algorithm

Today’s agenda

- HMM motivation and intuitive introduction
- HMM mathematical formulation
- HMM algorithms
  - Scoring: Forward-Backward Algorithm
  - Decoding: Viterbi and Forward-Backward Algorithms
  - Training: Baum-Welch Algorithm
- HMMs for phone and word modeling in ASR

The HMM Scoring Problem

- We have an HMM model $\lambda = \{A, B, \Pi\}$
- We are given an observation sequence $O = \{o_1, o_2, \dots, o_T\}$
- Our goal: compute $P(O|\lambda)$
  - We can use either the Forward Algorithm or the Backward Algorithm to do this.

The Forward Algorithm

Define the *forward variable:*

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, q_t = s_i \mid \lambda)$$

The forward variable $\alpha_t(i)$ tells us the joint probability of seeing the prefix of the observation sequence up to time t <u>and</u> being in state s_i at time t

The Forward Algorithm

We can solve for $\alpha_t(i)$ using dynamic programming.

Consider the very first timestep as the base case:

$$\alpha_1(i) = P(o_1, q_1 = s_i \mid \lambda) = \underbrace{P(o_1 \mid q_1 = s_i, \lambda)}_{b_i(o_1)} \underbrace{P(q_1 = s_i \mid \lambda)}_{\pi_i}$$

The Forward Algorithm

Now the induction step to derive $\alpha_{t+1}(i)$ from $\alpha_t(i)$:

$$\alpha_{t+1}(i) = P(o_1, \dots, o_{t+1}, q_{t+1} = s_i \mid \lambda)$$

$$\alpha_{t+1}(i) = P(o_1, \dots, o_t , q_{t+1} = s_i \mid \lambda)\, P(o_{t+1} \mid q_{t+1} = s_i, \lambda)$$

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} P(o_1, \dots, o_t, q_t = s_j, q_{t+1} = s_i \mid \lambda)\right] \underbrace{P(o_{t+1} \mid q_{t+1} = s_i, \lambda)}_{b_i(o_{t+1})}$$

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \underbrace{P(o_1, \dots, o_t, q_t = s_j \mid \lambda)}_{\alpha_t(j)} \underbrace{P(q_{t+1} = s_i \mid q_t = s_j, \lambda)}_{a_{ji}}\right] b_i(o_{t+1})$$

The Forward Algorithm

Finally, we can compute the score $P(o_1, \dots, o_T \,|\lambda)$ from the values of the forward variable at the final timestep:

$$P(o_1, \dots, o_T \,|\lambda) = \sum_{i=1}^{N} P(o_1, \dots, o_T \,, q_T = s_i|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

The Forward Algorithm

Putting it all together:

Initialization: $\alpha_1(i) = b_i(o_1)\pi_i$

Induction: $\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j)\, a_{ji}\right] b_i(o_{t+1})$

Termination: $P(o_1, \ldots, o_T \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$

Forward Illustration