EECS 496: Sequential Decision Making

Soumya Ray
sray@case.edu
Office: Olin 516
Office hours : T 4-5:30 or by appointment

Recap

- We can handle evidence in approximate inference in several ways. The simplest is to do _____ sampling, but ____ samples which _____ with the evidence. This is called _____.
- Alternatively, we could just sample the ______ variables. Then assign each sample a ____, which is the _______ of the _______ given _______ . This is called ______ .
- If we have evidence _____ in the topological sort, or evidence that is ______, both the above have problems. These problems are (i) ______ (ii) _______ .
- To alleviate these issues we can stop ___________ ________ samples. This creates a ______ _____ over samples. The resulting algorithm is called _________.
- What is burn in time? Why do we need it?
- A specific approach in the case of Bayes nets is called ____ ____. This generates the next sample by choosing a _______ variable V. Then it samples from Pr(V|____).
- If a Markov chain is _____ and satisfies ____ ____, then it will eventually converge to a _____ _____.
- Using this, we can show that Gibbs sampling produces the right result for a Bayes net, because _______ .

Today

- Sequential Probabilistic Models (Ch 15)

Sequential Models

- So far, no explicit representation of time

- What happens when we have a random process evolving over time?

Sequential Models

- We see a sequence of observations $o_1,\ldots,o_n$
- Each element is drawn from some background alphabet or vocabulary
  - Text classification---each observation is a word
  - PFM---each observation is an amino acid
  - Parsing---each observation is a word or phrase
  - Activity recognition from video---each observation is a frame
  - Radar/Lidar/Sensing---each observation is a sensor measurement
- Observation sequences have varying lengths

Generative Process Model

- Assume sequential data is generated by an underlying generating process
- This process has a state that could be discrete or continuous
- The state evolves over time
- At discrete time points, we observe something about the state
  - These are our sequences of observations

Discrete Time, Discrete State Sequential Probabilistic Models

Minus 1th approach

- Let’s ignore everything about process state and dependence; pretend the data isn’t sequential at all
  – Sometimes a reasonable first approximation
- Naïve Bayes for text classification

Zeroth approach

- Let's ignore dependence, but not process state
- We'll record a probability distribution over observations at each state of interest
  – "Position Specific Scoring Matrix"
- Only useful for fixed length sequences
  – Can also be used to find the subsequence of length *k* with highest probability

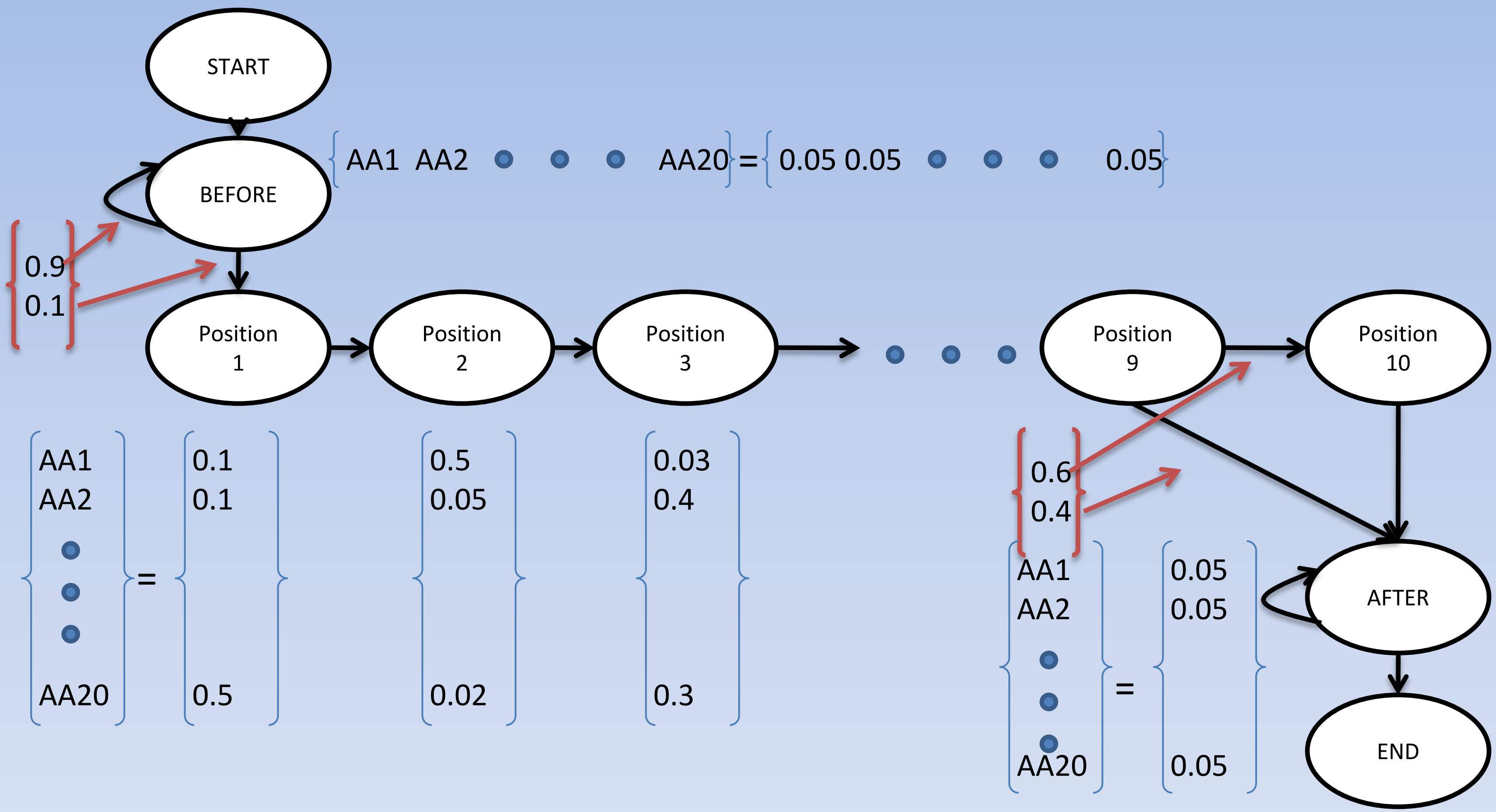
Example

- Suppose members of some family of proteins have a *motif*: a sequence of 10 amino acids somewhere that is specific to this family

k^{th} order approach

- We'll model dependence of "order k"
  - Assume each state is dependent on the previous k states
- We'll study the case for $k=1$
  - $k>1$ are straightforward extensions (just messy algebraically)
  - "First order Hidden Markov Models"

Hidden Markov Models



Set of **states**={BEFORE, Position1,….,Position10, AFTER}
Set of **emissions**={AA1, AA2,…, AA20}

Hidden Markov Models

- HMMs are generative process models for the joint distribution $\Pr(\mathbf{s},\mathbf{o})$

$$\Pr(\{s_1, s_2, \ldots, s_n\}, \{o_1, o_2, \ldots, o_n\}) =$$

$$\Pr(s_1)\Pr(o_1 \mid s_1)\prod_{r=2}^{n}\Pr(o_r \mid s_r)\Pr(s_r \mid s_{r-1})$$

"**Emission Probability**": How likely is process state s_r to emit observation o_r

"**Transition Probability**": How likely is process state s_{r-1} to transition to state s_r

Questions

- What’s “Markov” about this model?
- What changes for $k>1$?
- What’s “hidden”?

Key Issues

- Inference
  - What is the probability of an observed sequence $\mathbf{o}$?
  - What is the most likely sequence of process states $\mathbf{s}$ that could have emitted an observed sequence $\mathbf{o}$?

- Learning
  - Given a training set of observation sequences and a model structure, how do we estimate parameters for the model?

Issue #1: $\Pr(\mathbf{o})$

- Clearly, $\Pr(\mathbf{o}) = \sum_{\mathbf{s}} \Pr(\mathbf{s}, \mathbf{o})$
  - Sum over all possible state sequences that could generate $\mathbf{o}$
  - But the number of state sequences that could generate $\mathbf{o}$ could be exponential in the length of $\mathbf{o}$
- Key observation:
  - Many state sequences share prefixes
  - We only need to compute probabilities for a shared prefix *once*
    - In fact, because of the Markov property, we can do better
  - We can use dynamic programming to store and reuse these computations

Forward Algorithm

- Let $\alpha_k(i) = \Pr(o_1, \ldots, o_i, s_i = k)$
  - Denotes the probability that the model has emitted the first i observations and is now in state k
- We want to compute $\alpha_{END}(n)$ (recall the observation sequence is extended with dummy START and END symbols)
- Construct a table of size n-by-m, n=length of observed sequence, m=number of states
- The forward algorithm is a dynamic programming procedure that will fill in this table with α values

Forward Algorithm

- Initialize: $\alpha_{START}(0)=1, \alpha_k(0)=0, k \neq START$

- Recursion: Emitting observation i Transition to state k

$$\alpha_k(i) = \Pr(o_i \mid s_i = k) \sum_p \alpha_p(i-1) \Pr(s_i = k \mid s_{i-1} = p)$$

$$\alpha_k(i) = \Pr(o_1, \ldots, o_i, s_i = k)$$

$$\alpha_p(i-1) = \Pr(o_1, \ldots, o_{i-1}, s_{i-1} = p)$$

Issue #2: Most Likely Path

- Given an observation sequence, what is the most likely sequence of states that could emit it?

$$\mathbf{s}^* = \arg\max_{\mathbf{s}} \Pr(\mathbf{s} \mid \mathbf{o})$$

- Dumb way: enumerate all possible $\mathbf{s}$
- Smart way: dynamic programming, as before, taking advantage of Markov property
  - Viterbi algorithm

Viterbi Algorithm

- Let $\gamma_k(i) = \Pr(o_1, \ldots, o_i, s_i^* = k)$
  – Denotes the probability that the most likely path is at state k after emitting the first i observations

- We want $\gamma_{END}(n)$

- Notice that this just gives us the *probabilities*
  – To get the path, we will also need to maintain pointers to certain table elements

Viterbi Algorithm

- Initialize: $\gamma_{START}(0)=1, \gamma_k(0)=0, k \neq START$
- Recursion:

Emitting observation i

Transition to state k

$$\gamma_k(i) = \Pr(o_i \mid s_i^* = k) \max_p \gamma_p(i-1) \Pr(s_i^* = k \mid s_{i-1}^* = p)$$

$$\gamma_k(i) = \Pr(o_1, \ldots, o_i, s_i^* = k)$$

$$\gamma_p(i-1) = \Pr(o_1, \ldots, o_{i-1}, s_{i-1}^* = p)$$

To get the path, store the arg max's of the recursive computation.