In the Name of God, the Merciful, the Compassionate

Introduction to Bioinformatics
07.1 - Hidden Markov Model Theory

Instructor: Hossein Zeinali

Amirkabir University of Technology

Spring 2020

The Markov Chain

- A Markov chain models a sequence that incorporates a minimum amount of memory without being completely memoryless.
- Let $\mathbf{X} = X_1, X_2, \ldots, X_n$ be a **sequence of random variables**. Based on the Bayes rule, we have:

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \prod_{i=2}^{n} P(X_i | X_1, \ldots, X_{i-1})$$

- The random variables X are said to form a first-order **Markov chain** (the **Markov assumption**), provided that:

$$P(X_i | X_1, \ldots, X_{i-1}) = P(X_i | X_{i-1})$$

- As a consequence, for the first-order Markov chain we have:

$$P(X_1, X_2, \ldots, X_n) = P(X_1) \prod_{i=2}^{n} P(X_i | X_{i-1})$$

The Markov Chain (Cont.)

- If we associate X_i to a state, the Markov chain can be represented by a finite state process.
- Consider a Markov chain with N distinct states labeled by $\{1, \ldots, N\}$ and the state at time t is denoted as s_t. The parameters of a discrete Markov chain can be described as follows:

$$\begin{aligned} a_{ij} &= P(s_t = j \mid s_{t-1} = i) \quad 1 \leq i, j \leq N \\ \pi_i &= P(s_1 = i) \quad 1 \leq i \leq N \end{aligned}$$

  where a_{ij} is the transition probability from state i to state j; and π_i is the initial probability that the Markov chain will start in state i.
- Both transition and initial probabilities are bound to the constraints:

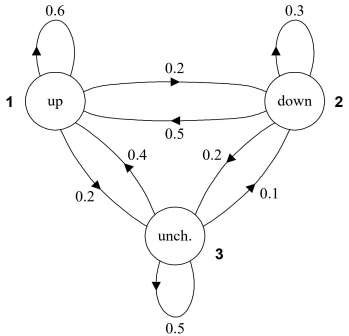
$$\sum_{j=1}^{N} a_{ij} = 1 \quad 1 \leq i \leq N$$

$$\sum_{i=1}^{N} \pi_i = 1$$

The Markov Chain (Cont.)

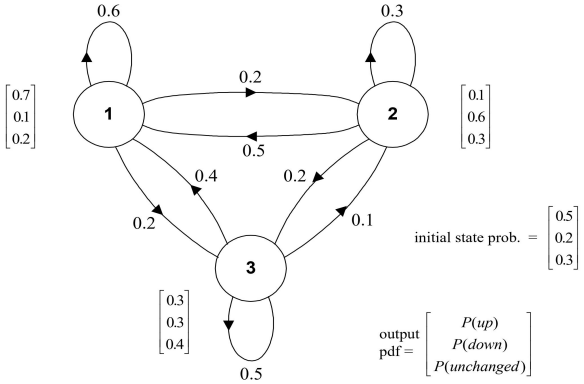
- This Markov chain is also called the **observable Markov model** because the output of the process is the set of states at each time instance t, where each state corresponds to an observable event X_i. There is one-to-one correspondence between the observable event sequence $\mathbf{X}$ and the Markov chain state sequence $\mathbf{S}$.
- Example: a simple three-state Markov chain for the Dow Jones Industrial average.

- **state 1** – up (in comparison to the index of previous day)
- **state 2** – down (in comparison to the index of previous day)
- **state 3** – unchanged (in comparison to the index of previous day)
- $\boldsymbol{\pi} = [0.5, 0.2, 0.3]^t$
- Example: $P(5 \text{ consecutive } \mathbf{up} \text{ days}) = \pi_1 {a_{11}}^4$



The Hidden Markov Model

- In the Markov chain, each state corresponds to a deterministically observable event.
- **Hidden Markov Model (HMM)**: the observation is a probabilistic function of the state.
- HMM is a double-embedded stochastic process with an underlying stochastic process (the state sequence) not directly observable.



The Hidden Markov Model (Cont.)

- There is no longer a one-to-one correspondence between the observation sequence and the state sequence.
- The state sequence is not observable and therefore hidden and this is why the world **hidden** is placed in front of Markov models.
- A hidden Markov model is defined by:
  - $\mathbf{O} = \{o_1, o_2, \ldots, o_M\}$ - An output observation alphabet. The observation symbols correspond to the physical output of the system being modeled. In previous example: $\mathbf{O} = \{up, down, unchanged\}$.
  - $\mathbf{\Omega} = \{1, 2, \ldots, N\}$ - A set of states representing the state space. Here s_t is denoted as the state at time t.
  - $\mathbf{A} = \{a_{ij}\}$ - A transition probability matrix, where a_{ij} is the probability of taking a transition from state i to state j, i.e.,

$$a_{ij} = P(s_t = j \mid s_{t-1} = i)$$

The Hidden Markov Model (Cont.)

- A hidden Markov model is defined by (cont.):
  - $\mathbf{B} = \{b_i(k)\}$ - An output probability matrix, where $b_i(k)$ is the probability of emitting symbol o_k when state i is entered. Let $\mathbf{X} = X_1, X_2, \ldots, X_t, \ldots$ be the observed output of the HMM. The state sequence $S = s_1, s_2, \ldots, s_t, \ldots$ is not observed (hidden), and $b_i(k)$ can be rewritten as follows:

$$b_i(k) = P(X_t = o_k \mid s_t = i)$$

  - $\boldsymbol{\pi} = \{\pi_i\}$ - A initial state distribution where:

$$\pi_i = P(s_0 = i)$$

- Since a_{ij}, $b_i(k)$ and π_i are all probabilities, they must satisfy the following properties:

$$a_{ij} \geq 0, b_i(k) \geq 0, \pi_i \geq 0 \quad \forall i, j, k$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad , \quad \sum_{k=1}^{M} b_i(k) = 1 \quad , \quad \sum_{i=1}^{N} \pi_i = 1$$

Discrete Density HMM Components

- A complete specification of an HMM includes:
  - N: total number of states
  - M: size of observation alphabets
  - O: observation alphabet
  - $\mathbf{A} \Rightarrow (N \times N)$ - State Transition Probability Matrix
  - $\mathbf{B} \Rightarrow (N \times M)$ - Output Occurrence Probability in each state
  - $\boldsymbol{\pi} \Rightarrow (1 \times N)$ - Initial State Probability
- For convenience, we use the following notation to indicate the whole parameter set of an HMM:

$$\boldsymbol{\Phi} = \{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\}$$

HMM Assumptions

- **Markov assumption**:

$$P(s_t|s_1, \ldots, s_{t-1}) = P(s_t|s_{t-1})$$

- **Output independence assumption**:

$$P(X_t|X_1, \ldots, X_{t-1}, s_1, \ldots, s_t) = P(X_t|s_t)$$

  which means the probability that a particular symbol is emitted at time t depends only on the state s_t and is conditionally independent of the past observations.

- These assumptions make evaluation, decoding, and learning feasible and efficient without significantly affecting the modeling capability, since those assumptions greatly reduce the number of parameters that need to be estimated.

Three Basic HMM Problems

1. **The Evaluation (Recognition) Problem** – Given a model $\mathbf{\Phi}$ and a sequence of observations $\mathbf{X} = (X_1, X_2, \ldots, X_T)$, what is the probability $P(\mathbf{X} \mid \mathbf{\Phi})$; i.e., the probability of the model that generates the observations? By doing evaluation, we could use HMM to do pattern recognition, since the likelihood $P(\mathbf{X} \mid \mathbf{\Phi})$ can be used to compute posterior probability $P(\mathbf{\Phi} \mid \mathbf{X})$.
2. **The Decoding Problem** – Given a model $\mathbf{\Phi}$ and a sequence of observations $\mathbf{X} = (X_1, X_2, \ldots, X_T)$, what is the most likely state sequence $\mathbf{S} = (s_0, s_1, s_2, \ldots, s_T)$ in the model that produces the observations?
3. **The Learning Problem** – Given a model $\mathbf{\Phi}$ and a set of observations, how can we adjust the model parameter $\hat{\mathbf{\Phi}}$ to maximize the joint probability (likelihood) $\prod_{\mathbf{X}} P(\mathbf{X} \mid \mathbf{\Phi})$?

How to Evaluate an HMM? First Problem Solution

- To calculate the probability (likelihood) $P(\mathbf{X} \mid \mathbf{\Phi})$ of the observation sequence $\mathbf{X} = (X_1, X_2, \ldots, X_T)$, we first enumerate all possible state sequences $\mathbf{S}$ of length T, that generate observation sequence $\mathbf{X}$, and then sum all the probabilities.

$$P(\mathbf{X} \mid \mathbf{\Phi}) = \sum_{all\ \mathbf{S}} P(\mathbf{S} \mid \mathbf{\Phi}) P(\mathbf{X} \mid \mathbf{S}, \mathbf{\Phi})$$

- For one particular state sequence $\mathbf{S} = (s_1, s_2, \ldots, s_T)$, where s_1 is the initial state, by applying Markov assumption we have:

$$P(\mathbf{S} \mid \mathbf{\Phi}) = P(s_1 \mid \mathbf{\Phi}) \prod_{t=2}^{T} P(s_t \mid s_{t-1}, \mathbf{\Phi}) = \pi_{s_1} a_{s_1 s_2} \ldots a_{s_{T-1} s_T}$$

- For the same state sequence $\mathbf{S}$, by applying the output-independent assumption we have:

$$P(\mathbf{X} \mid \mathbf{S}, \mathbf{\Phi}) = \prod_{t=1}^{T} P(X_t \mid s_t, \mathbf{\Phi}) = b_{s_1}(X_1) b_{s_2}(X_2) \ldots b_{s_T}(X_T)$$

How to Evaluate an HMM? First Problem Solution (Cont.)

- By substituting the last equations in the first one we get:

$$P(\mathbf{X} \mid \mathbf{\Phi}) = \sum_{all\ \mathbf{S}} \pi_{s_1} b_{s_1}(X_1) a_{s_1 s_2} b_{s_2}(X_2) \dots a_{s_{T-1} s_T} b_{s_T}(X_T)$$

- The direct evaluation of the above equation requires enumeration of $O(2TN^T)$ possible state sequences, which results in exponential computational complexity.
- Solution for complexity: store intermediate results and use them for subsequent state-sequence calculations to save computation. This algorithm is known as the ***forward algorithm***.
- Let's define forward probability as the probability that the HMM is in state i having generated partial observation $X_1, \dots, X_t$, i.e.

$$\alpha_t(i) = P(X_1, \dots, X_t, s_t = i \mid \mathbf{\Phi})$$

The Forward Algorithm

- **Step 1**: Initialization,

$$\alpha_1(i) = \pi_i b_i(X_1) \qquad 1 \leq i \leq N$$

- **Step 2**: Induction,

$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(X_t) \qquad 2 \leq t \leq T;\ 1 \leq j \leq N$$

- **Step 3**: Termination,

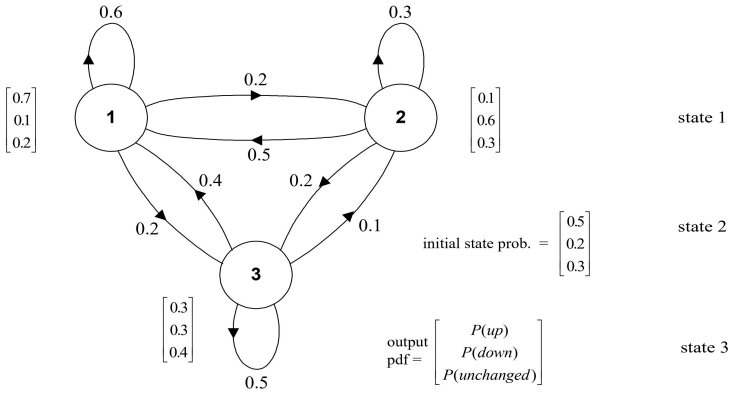
$$P(\mathbf{X} \mid \mathbf{\Phi}) = \sum_{i=1}^{N} \alpha_{_T}(i)$$

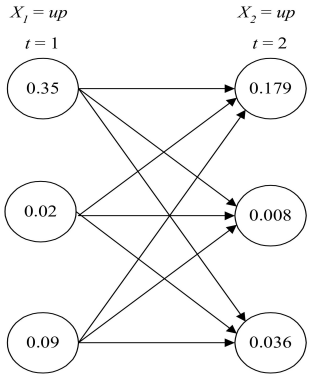
If it is required to end in the final state,

$$P(\mathbf{X} \mid \mathbf{\Phi}) = \alpha_{_T}(s_{_F})$$

- It is easy to show that the **complexity** for the forward algorithm is $O(TN^2)$ rather than the exponential one.

The Forward Algorithm: Example





The Backward Probabilities

- Let's define backward probability as the probability of generating partial observation $X_{t+1}, \ldots, X_T$ given that the HMM is in state i at time t, i.e.

$$\beta_t(i) = P(X_{t+1}, \ldots, X_T \mid s_t = i, \mathbf{\Phi})$$

- **Step 1**: Initialization,

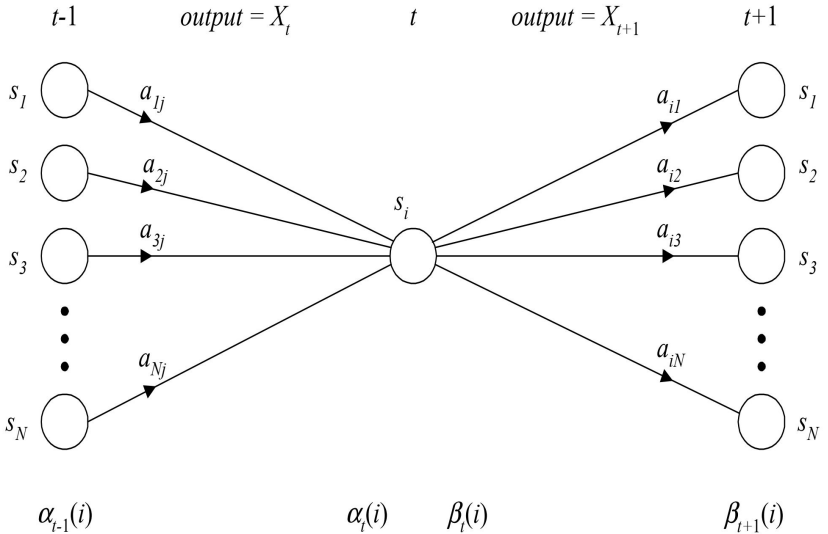
$$\beta_T(i) = 1 \qquad 1 \leq i \leq N$$

- **Step 2**: Induction,

$$\beta_t(i) = \left[\sum_{j=1}^{N} a_{ij} b_j(X_{t+1}) \beta_{t+1}(j) \right] \qquad t = T-1, \ldots, 1;\ 1 \leq i \leq N$$

- The complexity for computing backward probabilities is also $O(TN^2)$.

Forward vs Backward



How to Decode an HMM? The Viterbi Algorithm

- The forward algorithm computes the probability that an HMM generates an observation sequence by summing up the probabilities of all possible paths.
- In many applications, it is desirable to find the best path (or state sequence).
- We are looking for the state sequence $\mathbf{S} = (s_1, s_2, \ldots, s_T)$ that maximizes $P(\mathbf{S}, \mathbf{X} \mid \mathbf{\Phi})$.
- This problem is very similar to the optimal-path problem in dynamic programming.
- A formal technique based on dynamic programming, known as **Viterbi** algorithm can be used to find the best state sequence for an HMM.
- In practice, the same method can be used to evaluate HMMs that offers an approximate solution close to the case obtained using the forward algorithm.
- Instead of summing up probabilities from different paths coming to the same destination state, the Viterbi algorithm picks and remembers the best path.

How to Decode an HMM? The Trivial but not Valid Solution

- Let's define the probability of being in state S_i at time t given the observation sequence $\mathbf{X}$ and the model $\mathbf{\Phi}$, i.e.

$$\gamma_t(i) = P(s_t = i \mid \mathbf{X}, \mathbf{\Phi})$$

- The $\gamma_t(i)$ can be expressed simply in terms of the forward-backward variables, i.e.,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\mathbf{X} \mid \mathbf{\Phi})} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

  where the denominator is a scaling factor to make it a valid probability measure.

- Using $\gamma_t(i)$, we can solve for the individually most likely state s_t at time t as

$$s_t = \underset{1 \le i \le N}{\text{Argmax}} \left[\gamma_t(i)\right] \qquad 1 \le t \le T$$

- Note that there could be some problems with the resulting state sequence by this method, for example having an unreachable state in the state sequence.

The Viterbi Algorithm

- Let's define the best-path probability as the probability of the most likely state sequence at time t, which has generated the observation $X_1, \ldots, X_t$ (until time t) and ends in state i, i.e.

$$\delta_t(i) = \underset{s_1,\ldots,s_{t-1}}{\text{Max}} P(X_1, \ldots, X_t, s_1, \ldots, s_{t-1}, s_t = i \mid \mathbf{\Phi})$$

- **Step 1**: Initialization,

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(X_1) \qquad 1 \le i \le N \\ \psi_1(i) &= 0 \qquad \text{Backtracking variable} \end{aligned}$$

- **Step 2**: Induction,

$$\begin{aligned} \delta_t(j) &= \underset{1\le i\le N}{\text{Max}} \left[\delta_{t-1}(i) a_{ij}\right] b_j(X_t) \qquad 2 \le t \le T;\ 1 \le j \le N \\ \psi_t(i) &= \underset{1\le i\le N}{\text{Argmax}} \left[\delta_{t-1}(i) a_{ij}\right] \end{aligned}$$

The Viterbi Algorithm (Cont.)

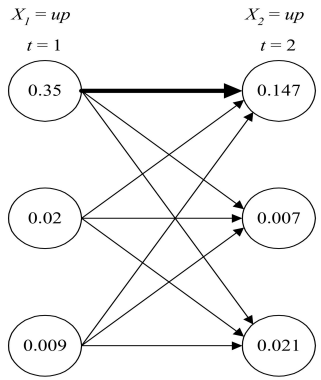
- **Step 3**: Termination,

$$p^* = \max_{1 \le i \le N} [\delta_T(i)]$$
$$q_T^* = \operatorname*{Argmax}_{1 \le i \le N} [\delta_T(i)]$$

- **Step 4**: Backtracking,

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad t = T-1, \ldots, 1$$
$$Q^* = (q_1^*, q_2^*, \ldots, q_T^*) \text{ is the best sequence}$$

- The complexity for the Viterbi algorithm is also $O(TN^2)$.



How to Estimate HMM Parameters? Baum-Welch Algorithm

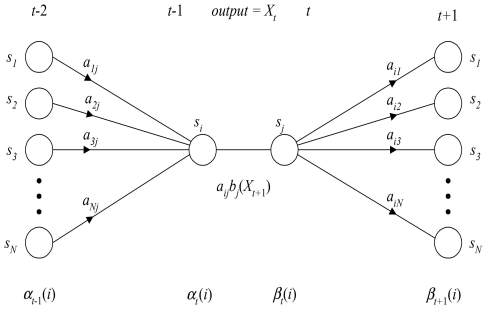
- It is very important to estimate the model parameters.
- This is by far the most difficult of the three problems.
- The problem can be solved by the iterative ***Baum-Welch*** algorithm, also known as the *forward-backward* algorithm.
- The HMM learning problem is a typical case of unsupervised learning where the data is incomplete because of the hidden state sequence.
- The Expectation Maximization (EM) algorithm is perfectly suitable for this problem and Baum-Welch algorithm uses the same principle as that of the EM algorithm.

The Baum-Welch Algorithm

- We define $\xi_t(i,j)$, which is the probability of taking the transition from state i to state j at time t, given the model and observation sequence, i.e.

$$\xi_t(i,j) = P(s_t = i, s_{t+1} = j \mid X_1, \ldots, X_T, \mathbf{\Phi}) = \frac{P(s_t = i, s_{t+1} = j, X_1, \ldots, X_T \mid \mathbf{\Phi})}{P(X_1, \ldots, X_T \mid \mathbf{\Phi})}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(X_{t+1}) \beta_{t+1}(j)}{\sum_{k=1}^{N} \alpha_T(k)} \quad \text{and we have} \quad \gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$



The Baum-Welch Algorithm (Cont.)

By doing EM optimization, we can find the following updating formulas for model parameters:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$

$$\hat{b}_j(k) = \frac{\sum_{t \in X_t = o_k} \sum_{i=1}^{N} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{i=1}^{N} \xi_t(i,j)}$$

$$\hat{\pi}_i = \sum_{j=1}^{N} \xi_1(i,j)$$

where:

$\sum_{t=1}^{T-1} \xi_t(i,j)$ is the expected number of transition from state i to state j.

$\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)$ is the expected number of transitions from state i.

$\sum_{t \in X_t = o_k} \sum_{i=1}^{N} \xi_t(i,j)$ is the expected number of times the observation data emitted from state j with the observation symbol o_k

$\sum_{t=1}^{T-1} \sum_{i=1}^{N} \xi_t(i,j)$ is the expected number of times the observation data emitted from state j.

The Baum-Welch Algorithm (Cont.)

- Since $\hat{a}_{ij}$, $\hat{b}_i(k)$ and $\hat{\pi}_i$ are all probabilities, they must satisfy the following restrictions:

$$\hat{a}_{ij} \geq 0, \quad \hat{b}_i(k) \geq 0, \quad \hat{\pi}_i \geq 0 \quad \forall\, i, j, k$$

$$\sum_{j=1}^{N} \hat{a}_{ij} = 1 \quad , \quad \sum_{k=1}^{M} \hat{b}_i(k) = 1 \quad , \quad \sum_{i=1}^{N} \hat{\pi}_i = 1$$

Reference

- Rabiner, Lawrence R. “A tutorial on hidden Markov models and selected applications in speech recognition.” Proceedings of the IEEE 77.2 (1989): 257-286.

Thanks for your attention$_{\text{g}}$