CH4: Hidden Markov Models: The Learning Problem

CS 182 Spring 2022

Scribes (from previous years): `alee113`, `chuang25`, `xqian6`

Scribes (2022): `hdandapa`, `ihuang8`, `spyon`

Compiled & edited by `eyouth`, `jwang194`

Reach out to cs1820tas@lists.brown.edu for any clarifications or corrections.

Overview of HMMs

Hidden Markov models (HMMs) are statistical Markov models whose states are unknown ("hidden"). Many biological sequences (i.e., DNA, protein, etc.) can be modelled as "outputs" generated by an HMM whose hidden state sequences and transition/emission probabilities can yield insight into the biology underlying the "observed" sequence. HMMs have found broad applications in bioinformatics and sequence inference.

Elements of an HMM

Construction

An HMM is constructed from several components:

States

Let $S = \{S_1, \ldots, S_N\}$ be the space of N states (which are unknown in standard HMM inference), and let q_t represent the state of the system at time t. The HMM state sequence proceeds in discrete units of time, starting from $t = 1$. The sequence of states followed by an HMM is denoted $Q = q_1 \ldots q_T$.

Emissions

Let $V = \{v_1, \ldots, v_M\}$ be the alphabet of M possible observed symbols ("emissions"), and let o_t represent the symbol emitted from state q_t at time t. For an HMM which is designed to infer properties of DNA sequences, $M = 4$ and $V = \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}\}$. The sequence of symbols emitted by an HMM is denoted $\mathcal{O} = o_1 \ldots o_T$.

Transition Probabilities

Let $A = \{a_{ij}\}$ represent the transition probabilities between any two (not necessarily distinct) states; i.e., $a_{ij} = P(q_{t+1} = S_j \,|\, q_t = S_i)$ for all $i, j \in \{1, \ldots, N\}$. Note that this probability is independent of t.

Emission Probabilities

Let $B = \{b_j(k)\}$ represent the emission probabilities in each state; i.e., $b_j(k) = b_{S_j}(v_k) = P(o_t = v_k \,|\, q_t = S_j)$ for all $j \in \{1, \ldots, N\}$ and $k \in \{1, \ldots, M\}$. Note that this probability is independent of t.

Initial State Distribution

Let $\pi = \{\pi_1, \ldots, \pi_N\}$ be the distribution of start states, such that $\pi_i = P(q_1 = S_i)$ for all $i \in \{1, \ldots, N\}$.

Definition

Define $\lambda = (A, B, \pi)$. The HMM λ is entirely defined by its transition (A), emission (B), and starting state (π) probabilities. Most applications of HMMs are focused on solving for or optimizing these parameters.

The 3 HMM Problems

Lawrence Rabiner's 1989 "tutorial" on HMMs outlines three fundamental HMM problems, summarized below. Problems 1 and 2 are covered in CS 181 and admit dynamic programming solutions of complexity $O(N^2T)$. Problem 3 is more difficult to solve and involves greater algorithmic complexity in its solution.

Problem 1: The Evaluation Problem

What is the probability of observing a sequence $\mathcal{O}$ given the HMM λ?

The probability of observing $\mathcal{O} = o_1 \dots o_T$ given an arbitrary sequence of states $Q = q_1 \dots q_T$ is

$$P(\mathcal{O} \mid Q) = \prod_{t=1}^{T} P(o_t \mid q_t) = \prod_{t=1}^{T} b_{q_t}(o_t)$$

while the prior probability of the sequence of states Q is

$$P(Q) = \pi_{q_1} \cdot \prod_{t=1}^{T-1} a_{q_t q_{t+1}}$$

Then the joint probability of observing $\mathcal{O}$ from the sequence of states Q is

$$P(\mathcal{O}, Q) = \pi_{q_1} b_{q_1}(o_1) \cdot a_{q_1 q_2} b_{q_2}(o_2) \cdots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

Therefore, the total probability of observing the sequence $\mathcal{O}$ given the HMM $\lambda = (A, B, \pi)$ can be computed by marginalizing the joint probability $P(\mathcal{O}, Q)$ over all possible sequences of states Q:

$$P(\mathcal{O} \mid \lambda) = \sum_{\text{all } Q} P(\mathcal{O}, Q) = \sum_{\text{all } Q} P(\mathcal{O} \mid Q) P(Q)$$

Problem 2: The Decoding Problem

What is the most likely sequence of states Q^* to have produced a sequence $\mathcal{O}$ given the HMM λ?

This problem can be solved by brute-force maximization over all possible sequences of states Q, but such an approach is computationally prohibitive in general. A dynamic programming solution such as that offered by the Viterbi algorithm or the Forward-Backward algorithm renders this problem computationally tractable. Several variables employed in these algorithms are also relevant to solving Problem 3:

Defn: The *forward variable* (α) gives the probability of observing a *prefix* of the emission sequence and being in a given state at the end of the prefix: $\alpha_t(i) = P(o_1 \dots o_t, q_t = S_i)$ for all $i \in \{1, \dots, N\}$ and $1 \leq t \leq T$.

Defn: The *backward variable* (β) gives the probability of observing a *suffix* of the emission sequence and being in a given state just before the start of the suffix: $\beta_t(i) = P(o_{t+1} \dots o_T, q_t = S_i)$ for all $i \in \{1, \dots, N\}$ and $1 \leq t \leq T$.

Defn: The *delta variable* (δ) gives the maximum probability of observing a prefix of the emission sequence and being in a given state at the end of the prefix: $\delta_t(i) = \underset{\text{all } q_1 \dots q_{t-1}}{\text{MAX}} P(o_1 \dots o_t, q_1 \dots q_{t-1} q_t, q_t = S_i)$ for all $i \in \{1, \dots, N\}$ and $1 \leq t \leq T$.

Problem 3: The Learning Problem

What parameters (A, B, π) maximize the probability of observing a sequence $\mathcal{O}$ given the HMM λ?

There is no exact analytical solution to this problem, although substantive numerical methods have been developed to obtain convergence to a desired degree. One such algorithm was developed by Leonard Baum and Lloyd Welch in the late 1960s and early 1970s.

The Baum-Welch Algorithm

The Baum-Welch algorithm employs an iterative process to "tune" the parameters (A, B, π) until some $\lambda' = (A', B', \pi')$ is achieved which locally maximizes $P(\mathcal{O} \mid \lambda')$. It is a special case of the broadly-applicable *expectation-maximization (EM) algorithm*, which is detailed further below. The Baum-Welch algorithm guarantees convergence to a desired threshold by constraining optimization ("re-estimation") of the parameters (A, B, λ) using properties of Lagrangian multipliers. Two additional relevant variables are as follows:

Defn: The *xi variable* (ξ) gives the probability of being in state S_i at time t and state S_j at time $t+1$, given the HMM λ and the sequence $\mathcal{O}$: $\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j \mid \mathcal{O}, \lambda)$ for all $i, j \in \{1, \ldots, N\}$ and $1 \leq t \leq T-1$.

Using the definitions of the forward and backward variables, the ξ variable can be rewritten as follows:

$$\xi_t(i,j) = \frac{P(q_t = S_i, q_{t+1} = S_j, \mathcal{O} \mid \lambda)}{P(\mathcal{O} \mid \lambda)}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i'=1}^{N} \sum_{j'=1}^{N} \alpha_t(i') a_{i'j'} b_{j'}(o_{t+1}) \beta_{t+1}(j')}$$

Defn: The *gamma variable* (γ) gives the probability of being in state S_i at time t, given the HMM λ and the sequence $\mathcal{O}$: $\gamma_t(i) = P(q_t = S_i \mid \mathcal{O}, \lambda)$ for all $i \in \{1, \ldots, N\}$ and $1 \leq t \leq T-1$.

The γ variable can be computed by marginalizing the ξ variable over all possible states $j \in \{1, \ldots, N\}$:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

Essentially, the ξ and γ variables yield "posterior" distributions which can be used to compute the expected numbers of transitions between pairs of states and "visits" to a given state, respectively:

$$\mathbb{E}(\#\text{ transitions } S_i \to S_j) = \sum_{t=1}^{T-1} \xi_t(i,j)$$

$$\mathbb{E}(\#\text{ "visits" to } S_i) = \sum_{t=1}^{T-1} \gamma_t(i)$$

Note that the "visits" to S_i are actually transitions *from* S_i; importantly, $q_t = S_i$ will be uncounted, since the ξ variable is only defined over $1 \leq t \leq T-1$. The value of $\gamma_T(i)$ can be computed by observing that $\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$, although this additional "visit" can generally be ignored if T is sufficiently large.

These values can then be used to update ("re-estimate") the parameters (A, B, π) as follows (consider the meaning behind each formula to gain an intuition for the process):

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad \text{for all } i, j \in \{1, \ldots, N\}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1, o_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad \text{for all } j \in \{1, \ldots, N\} \text{ and } k \in \{1, \ldots, M\}$$

$$\bar{\pi}_i = \gamma_1(i) \qquad \text{for all } i \in \{1, \ldots, N\}$$

This yields an updated set of parameters $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$. After each iterative step, either a local optimum is reached (i.e., $\bar{\lambda} \approx \lambda$ based on some convergence threshold) or the new model better explains the sequence observed (i.e., $P(\mathcal{O} \,|\, \lambda') > P(\mathcal{O} \,|\, \lambda)$). Iteration of this process eventually results in a *maximum likelihood estimation* (MLE) of the HMM (λ^*).

The re-estimation formulas can be derived by maximizing *Baum's auxiliary function*:

$$\begin{aligned} \mathcal{Q}(\lambda', \lambda) &= \sum_{\text{all } Q} \log[P(\mathcal{O}, Q \,|\, \lambda')] \cdot P(\mathcal{O}, Q \,|\, \lambda) \\ &= \sum_{\text{all } Q} \log[P(\mathcal{O}, Q \,|\, \lambda')] \cdot P(Q \,|\, \mathcal{O}, \lambda) \cdot P(\mathcal{O} \,|\, \lambda) \end{aligned}$$

Since $P(\mathcal{O} \,|\, \lambda)$ is not dependent on Q, it can be treated as a constant and so maximization can be equivalently performed using constrained optimization for the quantity

$$\sum_{\text{all } Q} \log[P(\mathcal{O}, Q \,|\, \lambda')] \cdot P(Q \,|\, \mathcal{O}, \lambda)$$

$$= \mathbb{E}_Q\big[\log[P(\mathcal{O}, Q \,|\, \lambda')] \,\big|\, \mathcal{O}, \lambda\big]$$

The *Baum-Welch Theorem* guarantees iterative improvement; i.e.,

$$\lambda' = \underset{\bar{\lambda}}{\text{ARGMAX}}[\mathcal{Q}(\bar{\lambda}, \lambda)] \implies P(\mathcal{O} \,|\, \lambda') \geq P(\mathcal{O} \,|\, \lambda)$$

Convergence to a local optimum λ^* is generally obtained by running the algorithm until the per-iteration change in log-likelihood falls below some fixed threshold.

The Baum-Welch algorithm can be implemented within the framework of the more general EM algorithm (developed by Arthur Dempster, Nan Laird, and Donald Rubin in 1977). The EM algorithm consists of two repeated steps: *expectation* ("E-step") and *maximization* ("M-step"). In the context of the Baum-Welch algorithm, the E-step is the calculation of Baum's auxiliary $\mathcal{Q}$ function, and the M-step is the maximization of this quantity to yield the arg max λ'. Iteration of the EM algorithm thus guarantees convergence to some λ^* which yields a locally-maximal probability of observing the sequence $\mathcal{O}$.

Throughout the re-estimation process, the parameters of the HMM are bound by the following stochastic constraints (which are automatically satisfied by the model at each iteration by construction):

$$\sum_{i=1}^{N} \bar{\pi}_i = 1 \qquad \sum_{j=1}^{N} \bar{a}_{ij} = 1 \text{ for all } i \in \{1, \ldots, N\} \qquad \sum_{k=1}^{M} \bar{b}_j(k) = 1 \text{ for all } j \in \{1, \ldots, N\}$$

These constraints can be employed to solve the optimization problem by the *method of Lagrange multipliers*. Under this method, the probability $P = P(\mathcal{O} \mid \lambda^*)$ is maximized when the following conditions hold:

$$\pi_i = \frac{\pi_i \frac{\partial P}{\partial \pi_i}}{\sum_{i'=1}^{N} \pi_{i'} \frac{\partial P}{\partial \pi_{i'}}}$$

$$a_{ij} = \frac{a_{ij} \frac{\partial P}{\partial a_{ij}}}{\sum_{j'=1}^{N} a_{ij'} \frac{\partial P}{\partial a_{ij'}}}$$

$$b_j(k) = \frac{b_j(k) \frac{\partial P}{\partial b_j(k)}}{\sum_{k'=1}^{M} b_j(k') \frac{\partial P}{\partial b_j(k')}}$$

It can be shown that these formulas are equivalent to those given by the Baum-Welch algorithm re-estimations above, proving that the originals formulas are indeed exactly correct at locally-optimal $P(\mathcal{O} \mid \lambda^*)$.

The Principle of Maximum Likelihood

The general principle of maximum likelihood supposes that a sample $\mathcal{D}_j$ is drawn from datasets $\{\mathcal{D}_1, \ldots, \mathcal{D}_c\}$, and that each sample is an independently and identically-distributed random variable from the distribution $p(x \mid w_j)$. This distribution is assumed to have a *known parameter form*; i.e., it is determined uniquely by the value of its parameter vector θ_j. For example, if $p(x \mid w_j) = N(\mu_j, \sigma_j)$, θ_j would be the vector of all components of μ_j and σ_j. The explicit dependence of $p(x \mid w_j)$ on θ_j may be shown by writing $p(x \mid \theta_j)$.

If these assumptions are satisfied, then optimization algorithms can be applied to select the MLE (θ_j^*) which yields the highest probability of observing the sample $\mathcal{D}_j$.

Solving the Maximum-Likelihood Problem

Inputs: training data ($\mathcal{D}_j$) sampled from datasets $\{\mathcal{D}_1, \ldots, \mathcal{D}_c\}$.

Outputs: estimates for the unknown parameter vectors $\{\theta_1, \ldots, \theta_c\}$.

For simplicity, it is assumed that classes of data are functionally different, so that $\mathcal{D}_i$ gives no information about θ_j for $i \neq j$. Then the problem reduces to c identical problems of the same form, and may be treated in terms of a generic data set $\mathcal{D}$, containing training samples which are drawn independently from the probability distribution $p(x \mid \theta)$ and used to estimate the unknown parameter vector θ.

Suppose $\mathcal{D}$ contains n samples $x_1, \ldots, x_n$. Because all x_i are drawn independently, the total probability of the data sample with respect to θ is

$$p(\mathcal{D} \mid \theta) = \prod_{i=1}^{n} p(x_i \mid \theta)$$

The maximum-likelihood estimate $\hat{\theta}$ is that which maximizes the likelihood $p(\mathcal{D} \mid \theta)$. Intuitively, this estimate corresponds to the value of θ which best supports the actual data sample observed.

For analytical reasons, it is generally easier to work with the logarithm of the likelihood than the likelihood itself. Since the log function is monotonically increasing, the value $\hat{\theta}$ which maximizes the log-likelihood also maximizes the likelihood. If $p(\mathcal{D} \mid \theta)$ is a differentiable function of θ, $\hat{\theta}$ can be found using standard methods from differential calculus.

Given $\theta = (\theta_1, \ldots, \theta_r)^\top$, the *gradient operator* (∇_θ) is defined as

$$\nabla_\theta = \left(\frac{\partial}{\partial \theta_1}, \ldots, \frac{\partial}{\partial \theta_r} \right)^\top$$

The *log-likelihood function* is defined as

$$L(\theta) = \log[p(\mathcal{D} \,|\, \theta)]$$

and the *maximum likelihood estimator* (MLE) is defined as

$$\hat{\theta} = \underset{\theta}{\text{ARGMAX}}[L(\theta)]$$

where the dependence on $\mathcal{D}$ is implicit.

By the independence condition assumed above,

$$\begin{aligned} L(\theta) &= \sum_{i=1}^{n} \log[p(x_i \,|\, \theta)] \\ \implies \nabla_\theta L(\theta) &= \sum_{i=1}^{n} \nabla_\theta \log[p(x_i \,|\, \theta)] \end{aligned}$$

and necessary conditions to solve for the maximum-likelihood estimate $\hat{\theta}$ can be obtained from the set of r equations given by

$$\nabla_\theta L(\theta) = 0$$

The Expectation-Maximization Algorithm

The methods for inferring the maximum-likelihood estimator $\hat{\theta}$ can be extended to enable learning of parameters governing a distribution from training points which may include missing data features. (If there is no missing data, $\hat{\theta}$ may be solved by maximizing the log-likelihood $L(\theta)$ as above.)

The basic idea of the EM algorithm is to iteratively estimate the likelihood of the data sample observed *given the data that is present*. Consider a full sample $\mathcal{D} = \{x_1, \ldots, x_n\}$ drawn from a single distribution. Suppose that some features are missing, so that each sample point can be defined as follows:

$$x_i = \{x_{i_g}, x_{i_b}\}$$

where present data features are denoted "good" (g) and missing data features are denoted "bad" (b). The features may then be separated into "good" and "bad" classes, $\mathcal{D}_g$ and $\mathcal{D}_b$, such that $\mathcal{D} = \mathcal{D}_g \cup \mathcal{D}_b$.

The *Baum function*, also known as the *central equation*, is defined as follows:

$$\mathcal{Q}(\theta; \theta^i) = \mathbb{E}_{\mathcal{D}_b}\left[\log[p(\mathcal{D}_g, \mathcal{D}_b; \theta)] \,|\, \mathcal{D}_g; \theta^i \right]$$

where $\mathcal{Q}$ is a function of θ with θ^i assumed to be fixed, and the expectation is marginalized over the missing features assuming that θ^i are the "true" parameters which describe the full distribution. θ represents a candidate vector for an improved estimate, while θ^i is the current best estimate for the full distribution. At each iteration, the algorithm selects θ^{i+1} as the candidate θ which maximizes $\mathcal{Q}(\theta; \theta^i)$.

Pseudocode for the EM algorithm is as follows:

Algorithm 1 EM Algorithm

BEGIN
initialize ϵ, θ^0, $i = 0$
repeat
  increment $i = i + 1$
  compute $\mathcal{Q}(\theta; \theta^i)$ {E step}
  set $\theta^{i+1} = \underset{\theta}{\text{ARGMAX}}[\mathcal{Q}(\theta; \theta^i)]$ {M step}
until $\mathcal{Q}(\theta^{i+1}; \theta^i) - \mathcal{Q}(\theta^i; \theta^{i-1}) \leq \epsilon$
return $\hat{\theta} = \theta^{i+1}$
END

The EM algorithm is most useful when the optimization of the $\mathcal{Q}$ function is simpler than that of the likelihood $L(\theta)$. Most importantly, it guarantees that the log-likelihood of the "good" data (with the "bad" data marginalized) will increase monotonically. Note that this is not equivalent to finding the particular values of the "bad" data which yield the maximum likelihood of the full, complete data!