HMM for Bioinformatics

Paweł Błażej
Department of Genomics, Faculty of Biotechnology,
blazej@smorfland.uni.wroc.pl

3 kwietnia 2019

1. Hidden Markov models (HMMs) form the basis for the majority of gene finders in use today;

1. Hidden Markov models (HMMs) form the basis for the majority of gene finders in use today;
2. A number of extensions to the HMM formalism exist which have been found invaluable in achieving the accuracy and flexibility required of a practical, state-of-the-art gene finder.

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;
2. alphabet α;

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;
2. alphabet α;
3. transition distribution $P_t : Q \times Q \longrightarrow \mathbf{R}$;

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;
2. alphabet α;
3. transition distribution $P_t : Q \times Q \rightarrow \mathbf{R}$;
4. initial state q^0;

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;
2. alphabet α;
3. transition distribution $P_t : Q \times Q \rightarrow \mathbf{R}$;
4. initial state q^0;
5. final state q^f;

Definition

We introduce the notion of a hidden Markov model as a stochastic machine denoted by a 6-tuple:

$$M = (Q, \alpha, P_t, q^0, q^f, P_e)$$

where

1. state set Q;
2. alphabet α;
3. transition distribution $P_t : Q \times Q \rightarrow \mathbf{R}$;
4. initial state q^0;
5. final state q^f;
6. emission distribution $P_e : Q \times \alpha \rightarrow \mathbf{R}$.

Conventions

1. We reserve the symbol q for particular states in the model: $Q = \{q^0, q_1, \ldots, q_{m-1}, q^f\}$, for $m = |Q|$;

Conventions

1. We reserve the symbol q for particular states in the model: $Q = \{q^0, q_1, \ldots, q_{m-1}, q^f\}$, for $m = |Q|$;
2. We denote the elements of the list using some generic variable (sequence of hidden states), such as y i.e., $\phi = (y_0, y_1, \ldots, y_{n-1})$ for $n = |\phi|$.

Conventions

1. We reserve the symbol q for particular states in the model: $Q = \{q^0, q_1, \ldots, q_{m-1}, q^f\}$, for $m = |Q|$;
2. We denote the elements of the list using some generic variable (sequence of hidden states), such as y i.e., $\phi = (y_0, y_1, \ldots, y_{n-1})$ for $n = |\phi|$.
3. For convenience, we will always assume $q^f = q^0$ that is, the 0th state in Q will always serve the function of initial and final state for the HMM;

Conventions

1. Thus, we can now denote an HMM more compactly as:

$$M = (Q, \alpha, P_t, P_e).$$

Conventions

1. Thus, we can now denote an HMM more compactly as:

$$M = (Q, \alpha, P_t, P_e).$$

2. We reserve the letter s for the elements of the alphabet $\alpha = \{s_0, \ldots, s_{k-1}\}$ for $k = |\alpha|$;

Conventions

1. Thus, we can now denote an HMM more compactly as:

$$M = (Q, \alpha, P_t, P_e).$$

2. We reserve the letter s for the elements of the alphabet $\alpha = \{s_0, \ldots, s_{k-1}\}$ for $k = |\alpha|$;
3. When dealing with an output sequence S we will use a generic variable such as x to denote the individual symbols in the sequence: $S = x_0, \ldots, x_{L-1}$, for $L = |S|$.

Simple explanation

1. a machine M operates by starting in state q^0;

Simple explanation

1. a machine M operates by starting in state q^0;
2. transitioning stochastically from state to state according to $P_t(y_i|y_{i-1})$, for $\{y_i, y_{i-1}\} \subseteq Q$;

Simple explanation

1. a machine M operates by starting in state q^0;
2. transitioning stochastically from state to state according to $P_t(y_i|y_{i-1})$, for $\{y_i, y_{i-1}\} \subseteq Q$;
3. Upon an entering a state q, the machine emits a symbol s according to $P_e(s|q)$;

Simple explanation

1. a machine M operates by starting in state q^0;
2. transitioning stochastically from state to state according to $P_t(y_i|y_{i-1})$, for $\{y_i, y_{i-1}\} \subseteq Q$;
3. Upon an entering a state q, the machine emits a symbol s according to $P_e(s|q)$;
4. terminating in state q^f.

Simple example

Let us consider a simple example:

$$M_1 = (\{q_0, q_1, q_2\}, \{Y, R\}, P_t, P_e)$$

where

$$P_t = \{(q_0, q_1, 1), (q_1, q_1, 0.8), (q_1, q_2, 0.15), (q_1, q_0, 0.05),$$

$$(q_2, q_2, 0.7), (q_2, q_1, 0.3)\}$$

and

$$P_e = \{(q_1, Y, 1), (q_,1, R, 0), (q_2, Y, 0), (q_2, R, 1)\}.$$

Simple example

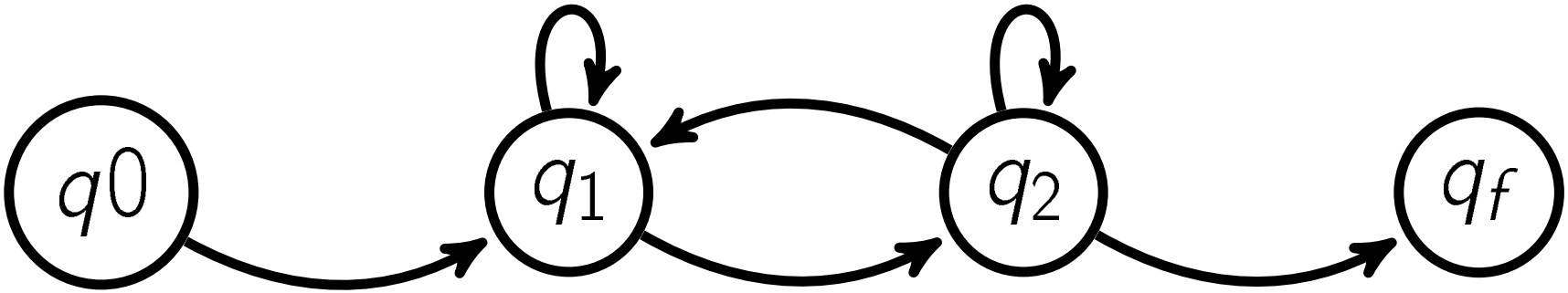
A single run of M_1, might produce the sequence S:

$$YRYRY.$$

An another run of the HMM we might observe S:

$$YRYYRYRRY.$$

HMM – example



The probability of $P(S|M_1)$

$$P(YRYRY|M_1) =$$

$$a_{0\to 1} \times b_{1,Y} \times a_{1\to 2} \times b_{2,R} \times a_{2\to 1} \times b_{1,Y} \times a_{1\to 2} \times b_{2,R} \times a_{2\to 1} \times b_{1,Y}.$$

where $a_{i\to j}$ denotes $P_t(q_j|q_i)$ whereas $b_{i,s}$ denotes $P_e(s|q_i)$.

Representing of HMMs

An HMM with states: $Q = \{q_1, q_2, \ldots, q_{n-1}\}$ and alphabet $\alpha = \{s_0, s_1, \ldots, s_{m-1}\}$ can be represented very simply in software by utilizing two matrices:

1. for the emissions probabilities, i.e. $n \times m$ matrix $E = (E_{ij})$ where $E_{ij} = P_e(s_j|q_i)$;

Representing of HMMs

An HMM with states: $Q = \{q_1, q_2, \ldots, q_{n-1}\}$ and alphabet $\alpha = \{s_0, s_1, \ldots, s_{m-1}\}$ can be represented very simply in software by utilizing two matrices:

1. for the emissions probabilities, i.e. $n \times m$ matrix $E = (E_{ij})$ where $E_{ij} = P_e(s_j|q_i)$;
2. for the transitions probabilities, i.e. $n \times n$ matrix $P = (P_{ij})$ where $P_{ij} = P_t(q_j|q_i)$.

The three basic problems for HMMs

1. Given the observation sequence $S = x_1, x_2, \ldots, x_k$ and the model $M = (Q, \alpha, q_0, P_t, P_e)$ how do we choose a corresponding hidden state sequence $y_1, y_2, \ldots, y_k$ which is optimal in some meaningful sense?

The three basic problems for HMMs

1. Given the observation sequence $S = x_1, x_2, \ldots, x_k$ and the model $M = (Q, \alpha, q_0, P_t, P_e)$ how do we choose a corresponding hidden state sequence $y_1, y_2, \ldots, y_k$ which is optimal in some meaningful sense?
2. Given the observation sequence $S = x_1, x_2, \ldots, x_k$ and the model $M = (Q, \alpha, q_0, P_t, P_e)$ how do we efficiently compute $P(S|M)$, the probability of the observation sequence, given the model?

The three basic problems for HMMs

1. Given the observation sequence $S = x_1, x_2, \ldots, x_k$ and the model $M = (Q, \alpha, q_0, P_t, P_e)$ how do we choose a corresponding hidden state sequence $y_1, y_2, \ldots, y_k$ which is optimal in some meaningful sense?
2. Given the observation sequence $S = x_1, x_2, \ldots, x_k$ and the model $M = (Q, \alpha, q_0, P_t, P_e)$ how do we efficiently compute $P(S|M)$, the probability of the observation sequence, given the model?
3. How do we adjust the model parameters $M = (Q, \alpha, q_0, P_t, P_e)$ to maximize $P(S|M)$?

Finding the most probable path

Decoding with an HMM can be performed using a dynamic programming procedure called the Viterbi algorithm. Given a model

$$M = (Q, \alpha, P_t, P_e)$$

with n hidden states and a nonempty sequence of emmited states

$$S = x_0 x_1, \ldots, x_{L-1},$$

the algorithm operates by progresively computing to find the most probable path.

Finding path – notations

The most probable path after the step k

$$\phi_{i,k} = y_0, \ldots, y_{k+1} \ (\forall_{0 \leqslant j \leqslant k+1} y_j \in Q;\ y_0 = q_0,\ y_{k+1} = q_i)$$

ending in state q_i at the position k whereby the model M could have generated the subsequence

$$x_0, x_1, \ldots, x_k.$$

Finding path – notations

Therefore:

$$\phi_{i,k} = \begin{cases} argmax_{\phi_{\mathbf{j,k-1}}+q_i}[P(\phi_{j,k-1}, x_0, \ldots, x_{L-1}) \cdot P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ q_0 q_i & if\ k = 0. \end{cases}$$

Finding path – notations

Therefore:

$$\phi_{i,k} = \begin{cases} argmax_{\phi_{j,k-1}+q_i}[P(\phi_{j,k-1}, x_0, \ldots, x_{L-1}) \cdot P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ q_0 q_i & if\ k = 0. \end{cases}$$

where

$$P(\phi_{j,k}, x_0, \ldots, x_{L-1}) = \begin{cases} max_j[P(\phi_{j,k-1}, x_0, \ldots, x_{L-1}) \cdot P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ P_t(q_i|q_0)P_e(x_0|q_i) & if\ k = 0. \end{cases}$$

Finding path – notations

Once we have computed $\phi_{i,k}$ for all states q_i and all positions k in the sequence, it is then a simple matter to select the most probable path for the full sequence of S by comparatively enumerating all paths ending at the last symbol x_{L-1}, with the additional consideration that the last act of the machine after emiting x_{L-1} must have been to transition into state q_0.

$$\phi' = argmax_{\phi_{i,L-1}} P(\phi_{i,L-1}, S) P_t(q_0|q_i)$$

The Viterbi algorithm – notations

The Viterbi algorithm utilizes the following dynamic programming recurrence to efficiently compute the probabilities $P(\phi_{i,k}, S_{0\ldots k})$ of the prospective paths:

$$V(i,k) = \begin{cases} max_j[V(j,k-1)P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ P_t(q_i|q_0)P_e(x_0|q_i) & if\ k = 0. \end{cases}$$

The Viterbi algorithm – notations

The Viterbi algorithm utilizes the following dynamic programming recurrence to efficiently compute the probabilities $P(\phi_{i,k}, S_{0\ldots k})$ of the prospective paths:

$$V(i,k) = \begin{cases} max_j[V(j,k-1)P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ P_t(q_i|q_0)P_e(x_0|q_i) & if\ k = 0. \end{cases}$$

Clearly, $V(i,k)$ represents the probability $P(\phi_{i,k}, S_{0\ldots k})$ of the most probable path $\phi_{i,k}$ which ends at the state q_i and emits the subsequence $x_0, \ldots, x_k$.

The Viterbi algorithm – notations

The optimal predecessor link $T(i,k)$

$$T(i,k) = \begin{cases} argmax_j[V(j,k-1)P_t(q_i|q_j)P_e(x_k|q_i)] & if\ k > 0 \\ P_t(q_i|q_0)P_e(x_0|q_i) & if\ k = 0. \end{cases}$$

Each element $T(i,k)$ is thus a state index j for the optimal predecessor q_j of q_i at position k in the sense that the optimal path $\phi_{i,k}$ must have as its last transition $q_j \rightarrow q_i$.

The Viterbi algorithm

Computing probability of a sequence

1. A procedure very similar to the Viterbi algorithm can be used to find the probability that a given model M emits (nonempty) sequence S during any given run of the machine i.e. $P(S|M)$;

Computing probability of a sequence

1. A procedure very similar to the Viterbi algorithm can be used to find the probability that a given model M emits (nonempty) sequence S during any given run of the machine i.e. $P(S|M)$;
2. Because M may potentially emit S via any number of paths through the states of the model, to compute the full probability of the sequence we need to sum over all possible paths emiting S.

The forward algorithm

$$F(i,k) = \begin{cases} 1 & \text{for } k = 0,\ i = 0 \\ 0 & \text{for } k > 0,\ i = 0 \\ 0 & \text{for } k = 0,\ i > 0 \\ \sum_{j=0}^{|Q|-1} F(j,k-1) P_t(q_i|q_j) P_e(x_k|q_i) & \text{for } 1 \leqslant k \leqslant |S|, \\ & 1 \leqslant i \leqslant |Q| \end{cases}$$

Therefore:

$$P(S|M) = \sum_{i=0}^{|Q|-1} F(i,|S|) P_t(q_0|q_i)$$