Hidden Markov Models

Stochastic Regular Grammars

I. Holmes

Department of Bioengineering
University of California, Berkeley

Spring semester

Outline

- Early motivation: isochores
  - Long regions of uniform GC content (which is correlated with gene density, recombination frequency...)
    - e.g. Major Histocompatibility Complex (MHC) class II and class III sequences on human chromosome 6
    - Lengths 900.9 kb, 642.1 kb; GC-content 41%, 52%
  - Gary Churchill: first Hidden Markov Model for isochore detection (1989)
  - Earliest non-thermodynamic hit to "isochore" on PubMed is 1986, Alonso *et al*
  - HMM analogy: occasionally dishonest casino (Durbin *et al*)
- Hidden Markov model: notation
  - Let x denote hidden state, y observed symbol. State space includes START and END
  - Let $e(x, y)$ be probability of emitting character y in state x
  - Let $t(i, j)$ be probability of transition to state j if currently in state i
- Idea of a particular partitioning as a "path" through the

- Definition of the posterior probability that position n is in state k: sum over paths

$$P(x_n = k|Y) = \frac{\sum_X P(X,Y)\delta(x_n = k)}{P(Y)}$$

- Splitting the path into three parts: $< n$, $= n$ and $> n$

$$P(x_n|Y) = \sum_{x_1 \ldots x_{n-1}} \sum_{x_{n+1} \ldots x_L} \frac{P(x_1 \ldots x_n, y_1 \ldots y_n|x_0)\ P(x_{n+1} \ldots x_{L+1}, y$$

where

$$B_n(x_n) = P(x_{L+1}, y_{n+1} \ldots y_L|x_n) = \sum_{x_{n+1} \ldots x_L} P(x_{n+1} \ldots x_{L+1}, y_{n+1} \ldots$$

Likewise,

$$P(x_n, x_{n+1}|Y) = \frac{F_n(x_n)t(x_n, x_{n+1})e(x_{n+1}, y_{n+1})B_{n+1}(x_{n+1})}{P(Y)}$$

- Motivation: pairwise sequence alignment, pairwise genefinding, etc.
- Let x denote hidden state, y character in sequence Y, z character in sequence Z
- Let $\Delta y(x)$ be 1 if state x emits a character to Y, and 0 otherwise; likewise $\Delta z(x) = 1$ iff x emits to Z
- Emission probability $e(x, y, z)$ is defined as follows:
  - If $\Delta y(x) = 1$ and $\Delta z(x) = 0$, then x is called a **delete** state and $e(x, y, z) \equiv e_d(x, y)$ is a function of x and y only
  - If $\Delta y(x) = 0$ and $\Delta z(x) = 1$, then x is called an **insert** state and $e(x, y, z) \equiv e_i(x, z)$ is a function of x and z only
  - If $\Delta y(x) = 1$ and $\Delta z(x) = 1$, then x is called a **match** state and $e(x, y, z) \equiv e_m(x, y, z)$ is a function of x, y and z
  - If $\Delta y(x) = 0$ and $\Delta z(x) = 0$, then x is called a **null** state and $e(x, y, z)$ is a function of x only (typically just 1)
  - We will assume for now that there are no null states (apart

- As before, $t(i, j)$ is the probability of transition to state j if currently in state i
- Suppose sequence lengths are K, L so observed data are $Y = \{y_1 \dots y_K\}$ and $Z = \{z_1 \dots z_L\}$
- Again we have a state path $x_1, x_2 \dots x_N$ and for convenience we set $x_0 =$START and $x_{N+1} =$END.
  - Denote by Λ_{kl} the event that there exists a *break* at (k, l):

$$\Lambda_{kl} \;\Rightarrow\; \exists n : \sum_{i=1}^{n} \Delta y(x_i) = k, \sum_{i=1}^{n} \Delta z(x_i) = l$$

So Λ_{kl} means that, at some point n on the state path, the model has emitted k symbols to Y and l symbols to Z.

- Viterbi

$$V_{kl}(x_n) = \max_{x_1 \ldots x_{n-1}} P\left(\Lambda_{kl}, x_1 \ldots x_n, y_1 \ldots y_k, z_1 \ldots z_l | x_0\right)$$

Recursion (assuming no null states)

$$V_{kl}(x_n) = \begin{cases} e(x_n, y_k, z_l) \max_{x_{n-1}} t(x_{n-1}, x_n) V_{k-\Delta y(x_n), l-\Delta z(x_n)}(x_{n-1}) \\ 1 \\ 0 \\ 0 \end{cases}$$

- Forward

$$F_{kl}(x_n) = P\left(\Lambda_{kl}, x_n, y_1 \ldots y_k, z_1 \ldots z_l | x_0\right) = \sum_{x_1 \ldots x_{n-1}} P\left(\Lambda_{kl}, x_1 \ldots x_n,\right.$$

Recursion (assuming no null states)

$$F_{kl}(x_n) = \begin{cases} e(x_n, y_k, z_l) \sum_{x_{n-1}} t(x_{n-1}, x_n) F_{k-\Delta y(x_n), l-\Delta z(x_n)}(x_{n-1}) & \text{if} \\ 1 & \text{if} \\ 0 & \text{if} \\ 0 & \text{if} \end{cases}$$

- Backward

$$B_{kl}(x_n) = P(\Lambda_{kl}, x_{N+1}, y_{k+1} \dots y_K, z_{l+1} \dots z_L | x_n) = \sum_{x_{n+1} \dots x_N} P(\Lambda_{kl}, x$$

Recursion (assuming no null states)

$$B_{kl}(x_n) = \begin{cases} \sum_{x_{n+1}} t(x_n, x_{n+1}) e(x_{n+1}, y_{k+1}, z_{l+1}) B_{k+\Delta y(x_n+1), l+\Delta z(x_n+} \\ t(x_n, \text{END}) \\ 0 \end{cases}$$

- Evidence, posterior probabilities & EM counts

$$P(Y,Z) = \sum_x F_{KL}(x)t(x,\mathrm{END})$$

$$P(\Lambda_{kl}, x_n | Y, Z) = \frac{F_{kl}(x_n)B_{kl}(x_n)}{P(Y)}$$

$$P(\Lambda_{kl}, x_n, x_{n+1} | Y) = \frac{F_{kl}(x_n)t(x_n,x_{n+1})e(x_{n+1},y_{k+1},z_{l+1})B_{k+\Delta y}}{P(Y)}$$

$$\hat{t}(i,j) = \sum_{k=0}^{K}\sum_{l=0}^{L} P(\Lambda_{kl}, x_n = i, x_{n+1} = j | Y, Z)$$

$$\hat{e}_m(x,y,z) = \sum_{k:y_k=y}\sum_{l:z_l=z} P(\Lambda_{kl}, x_n = x | Y, Z)$$

$$\hat{e}_d(x,y) = \sum \sum^{L} P(\Lambda_{kl}, x_n = x | Y, Z)$$

Decision theory ("optimal accuracy").

- Decision theory: maximise expected "reward", making use of the posterior distribution
- Overlap score: an objective function (i.e. reward) that compares predicted alignment α with true alignment α'
  - Overlap score is $|\alpha \cap \alpha'|$, where an alignment is viewed as a set of match co-ords $\alpha = \{(k_1, l_1), (k_2, l_2) \ldots\}$
  - Several other good objective functions (e.g. "Cline shift score"); overlap is simpler, albeit less realistic
    - NB also $\delta(\alpha = \alpha')$ which only rewards perfect alignments, yielding a multiplicative, Viterbi-like recursion
  - Example criteria: how good is alignment for structure prediction? homology detection? benchmark of choice?
    - e.g. PROBCONS (Batzoglou *et al*) uses the sum-of-pairs score, same as the BAliBASE benchmark
- Posterior expectation of overlap score for an alignment (NB only match states have $\Delta y(x)\Delta z(x) \neq 0$)

- Dynamic programming algorithms whose finite state automata are almost or exactly Pair HMMs
  - Needleman-Wunsch; Smith-Waterman; Gotoh; Altschul, Proteins 1998
  - General implementations: DART library (C++), Exonerate (C), HMMoC (Java/C++), ...

- Can readily extend the Pair HMM to a multi-sequence HMM for multiple sequence alignment
  - Arbitrary number N of output sequences $Y^{(1)}, Y^{(2)}, Y^{(3)} \ldots Y^{(N)}$ of lengths $L_1 \ldots L_N$ (see e.g. Holmes 2003)
  - Dynamic programming time/memory complexity is $O(\prod_n L_n)$—not cheap
  - Ultimately, would like to structure $\Delta Y^{(n)}(x)$, $t(x, x')$ and $e(x, y^{(1)} \ldots y^{(N)})$ according to some underlying phylogenetic tree
    - The DP algorithms can also be tree structured, c.f. "progressive alignment"
  - For now, we ignore phylogenetic structure of indels (Δ,t) and concentrate on substitution model (e)
- Initial, simplistic, restrictive concept of Evolutionary HMM, lacking a good gap model:
  - **A single-sequence HMM that emits**

- HMMs model $P(Y) = \sum_X P(X, Y)$ (*generative* modeling). ML training maximizes this probability.
- Intuitively, since we are interested in predicting X correctly, it may make more sense to model conditional probability $P(X|Y)$ (*discriminative* modeling)
- Consider the conditional likelihood for an HMM, expressed in terms of the *feature vector* $\{u, f\}$ implied by X:

$$\log P(X|Y) = \frac{1}{P(Y)} \exp \left(\sum_{i,j} u(i,j) \log t(i,j) + \sum_{i,k} f(i,k) \log e(i,k \right.$$

where $P(Y)$ is computed by the Forward algorithm.

- We can write down a likelihood $P(X|Y)$ for a similarly trellis-structured graphical model as follows

Summary

- HMMs