Topics

- Hidden markov models
  - posterior probabilities over states
  - the EM algorithm
  - viterbi (dynamic programming)
A hidden Markov model (HMM) is a model where we generate a sequence of outputs in addition to the Markov state sequence:

\[
\begin{align*}
    s_0 & \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots \\
    \downarrow & \quad \downarrow \quad \downarrow \\
    x_0 & \quad x_1 \quad x_2
\end{align*}
\]

- To fully specify an HMM, we need to know:
  1. the number of states \( m \)
  2. the initial state distribution \( P_0(s_0) \)
  3. the hidden state transition probabilities \( P_1(s_{t+1}|s_t) \)
  4. the output distribution \( P_o(x_t|s_t) \) (discrete or continuous)
HMM problems: review

- There are several problems we have to solve
  1. How do we evaluate the probability that our model generated the observation sequence \( \{x_0, x_1, \ldots, x_n\} \)?
     - forward-backward algorithm
  2. How do we uncover the most likely hidden state sequence corresponding to these observations?
     - dynamic programming
  3. How do we adapt the parameters of the HMM to better account for the observations?
     - the EM-algorithm
Recursive computation: review

$$s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_{n-1} \rightarrow s_n$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$x_0 \quad x_1 \quad x_2 \quad x_{n-1} \quad x_n$$

- **Forward (predictive) probabilities** $\alpha_t(i)$:

  $$\alpha_t(i) = P(x_0, \ldots, x_t, s_t = i)$$

  $$\frac{\alpha_t(i)}{\sum_j \alpha_t(j)} = P(s_t = i | x_0, \ldots, x_t)$$

- **Backward (diagnostic) propabilities** $\beta_t(i)$:

  $$\beta_t(i) = P(x_{t+1}, \ldots, x_n | s_t = i)$$

  (evidence about the current state from future observations)
Recursive computation: review

\[ s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_{n-1} \rightarrow s_n \]
\[ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \]
\[ x_0 \quad x_1 \quad x_2 \quad x_{n-1} \quad x_n \]

Forward recursion:

\[ \alpha_0(i) = P_0(s_0 = i) P_o(x_0 | s_0 = i) \]
\[ \alpha_t(i) = \sum_j \alpha_{t-1}(j) P_1(s_t = i | s_{t-1} = j) P_o(x_t | s_t = i) \]

Backward recursion:

\[ \beta_n(i) = 1 \]
\[ \beta_{t-1}(i) = \sum_j P_1(s_t = j | s_{t-1} = i) P_o(x_t | s_t = j) \beta_t(j) \]
Uses of forward/backward probabilities

- Complementary forward/backward probabilities

\[ \alpha_t(i) = P(x_0, \ldots, x_t, s_t = i) \]
\[ \beta_t(i) = P(x_{t+1}, \ldots, x_n | s_t = i) \]

permit us to evaluate various (posterior) probabilities

First, we can evaluate the probability of the observation sequence:

\[ P(x_0, \ldots, x_n) = \sum_i P(x_0, \ldots, x_n, s_t = i) \]

\[ = \sum_i P(x_0, \ldots, x_t, s_t = i)P(x_{t+1}, \ldots, x_n | s_t = i) \]

\[ = \sum_i \alpha_t(i) \beta_t(i) \]
Forward/backward probabilities cont’d

\[ s_0 \rightarrow \ldots \rightarrow s_t \rightarrow s_{t+1} \rightarrow \ldots \rightarrow s_n \]

\[ \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \]

\[ x_0 \quad x_t \quad x_{t+1} \quad x_n \]

- We can evaluate the posterior probability that the HMM was in a particular state \( i \) at time \( t \)

\[
P(s_t = i | x_0, \ldots, x_n) = \frac{P(x_0, \ldots, x_n, s_t = i)}{P(x_0, \ldots, x_n)}
\]

\[
= \frac{\alpha_t(i) \beta_t(i)}{\sum_j \alpha_t(j) \beta_t(j)} \overset{def}{=} \gamma_t(i)
\]
We can also compute the posterior probability that the system was in state \( i \) at time \( t \) AND transitioned to state \( j \) at time \( t+1 \):

\[
P(s_t = i, s_{t+1} = j | x_0, \ldots, x_n) = \frac{\alpha_t(i) \ P_1(s_{t+1} = j | s_t = i) P_o(x_{t+1} | s_{t+1} = j) \ \beta_{t+1}(j)}{\sum_j \alpha_t(j) \ \beta_t(j)}
\]

\[
def \xi_t(i, j),
\]

where \( t = 0, \ldots, n - 1 \).
The EM algorithm for HMMs

Assume we have $L$ observation sequences $x_{0}^{(l)}, \ldots, x_{n_{l}}^{(l)}$

**E-step:** compute the posterior probabilities

$$
\gamma_{t}^{(l)}(i) \quad \text{for all } l, i, \text{ and } t \ (t = 0, \ldots, n_{l})
$$

$$
\xi_{t}^{(l)}(i, j) \quad \text{for all } l, i, \text{ and } t \ (t = 0, \ldots, n_{l} - 1)
$$

**M-step:** First, the initial state distribution can be updated according to the expected fraction of times the sequences started from a specific state $i$

$$
\hat{P}_{0}(i) \leftarrow \frac{1}{L} \sum_{l=1}^{L} \gamma_{0}^{(l)}(i)
$$
M-step cont’d

Second, to update the transition probabilities, we first define the expected number of transitions from $i$ to $j$

$$\hat{n}(i, j) = \sum_{l=1}^{L} \sum_{t=0}^{n-1} \xi_{t}^{(l)}(i, j)$$

- The maximum likelihood estimate of the one step transition probabilities can be obtained by normalization

$$\hat{P}_1(j|i) \leftarrow \frac{\hat{n}(i, j)}{\sum_{j'} \hat{n}(i, j')}$$
M-step cont’d

- Third, if the outputs are discrete, we define the expected number of times a particular observations say \( x = k \) was generated from a specific state \( i \)

\[
\hat{n}_o(i, k) = \sum_{l=1}^{L} \sum_{t=0}^{n_l} \gamma_t^{(l)}(i) \delta(x_t^{(l)}, k)
\]

The ML estimate is again obtained by normalization

\[
\hat{P}_o(k|i) \leftarrow \frac{\hat{n}_o(i, k)}{\sum_{k'} \hat{n}_o(i, k')}
\]
M-step cont’d

- If the outputs are continuous (e.g., multi-variate Gaussian), we have to solve a weighted maximum likelihood estimation problem as in the mixture of Gaussians models.

Separately for each state \( i \) we maximize:

\[
J(\theta_i) = \sum_{l=1}^{L} \sum_{t=0}^{n_l} \gamma_t^{(l)}(i) \log P(x_t^{(l)}|\theta_i)
\]

with respect to the parameters \( \theta_i \) (e.g., the mean and the covariance).
HMM example

Observed output as a function of time

- We will try to model this with a 3-state HMM with Gaussian outputs \( p(x|s = i) = p(x|\mu_i, \sigma_i^2), \ i = 1, 2, 3. \)
prior/posterior means and $\gamma_t(\cdot)$

prior mean ($t$) = $\sum_i P_t(i) \mu_i$ (‘*’)

posterior mean ($t$) = $\sum_i \gamma_t(i) \mu_i$ (‘*’)

where $P_t(i)$ is the probability of being in state $i$ after $t$ steps without observations; $\mu_i$ is the mean output from the $i^{th}$ state
HMM example cont’d

Log-prob. of data after 0 iterations

after 7 iterations

final

Tommi Jaakkola, MIT AI Lab
Dynamic programming (Viterbi)

- The probability of generating a particular hidden state sequence $s_0 = 1$, $s_1 = 2$, $s_2 = 1$ and the observations is

$$P_0(1)P_o(heads|1) \times P_1(2|1)P_o(tails|2) \times P_1(1|2)P_o(heads|2)$$

$x_0 = heads$, $x_1 = tails$, $x_2 = heads$
Dynamic programming (Viterbi)

- The probability of the most likely (partial) state sequence and the corresponding observations:

\[
\delta_t(i) = \max_{s_0, \ldots, s_{t-1}} \left\{ P(x_0, \ldots, x_{t-1}, s_0, \ldots, s_{t-1}) \right\} P_o(x_t|s_t = i) \\
= \max_{s_0, \ldots, s_{t-1}} \left\{ P(s_0)P_o(x_0|s_0) \cdots P_1(s_t = i|s_{t-1}) \right\} P_o(x_t|s_t = i)
\]

\[x_0 = \text{heads}, \ x_1 = \text{tails}, \ x_2 = \text{heads}\]
Dynamic programming (Viterbi)

\[ x_0 = \text{heads}, \quad x_1 = \text{tails}, \quad x_2 = \text{heads} \]

- Recursive updates (same as forward probabilities but “sum” replaced with “max”)

\[
\begin{align*}
\delta_0(j) &= P_0(j)P_0(\text{heads}|j), \quad j = 1, 2 \\
\delta_1(1) &= \max \{ \delta_0(1)P_1(1|1), \delta_0(2)P_1(1|2) \} \times P_0(\text{tails}|1) \\
\delta_1(2) &= \max \{ \delta_0(1)P_1(2|1), \delta_0(2)P_1(2|2) \} \times P_0(\text{tails}|2) \\
&\ldots
\end{align*}
\]
Dynamic programming: backtracking

- The most likely value for state $s_2$ is the one that corresponds to the most likely path

$$s_2^* = \arg\max \{ \delta_2(1), \delta_2(2) \}$$

(say $s_2^* = 1$ as in the figure)

- The most likely previous state is

$$s_1^* = \arg\max \{ \delta_1(1)P_1(1|1), \delta_1(2)P_1(1|2) \}$$

and so on... (what about observations?)
Dynamic programming: properties

Red path (dotted): most likely path landing on $s_2 = 2$
Blue path (dashed): most likely path landing on $s_2 = 1$

• Possible?