HMM Algorithms

Peter Bell

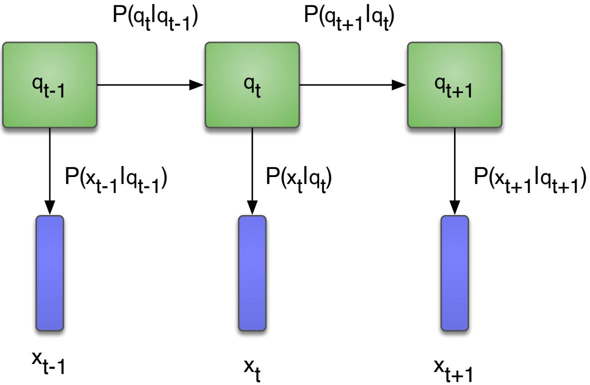
Automatic Speech Recognition— ASR Lecture 5
26 January 2026

Overview

HMM algorithms

- HMM recap
- HMM algorithms (2)
  - Likelihood computation (forward algorithm)
  - Finding the most probable state sequence (Viterbi algorithm)
  - Estimating the parameters (forward-backward and EM algorithms)

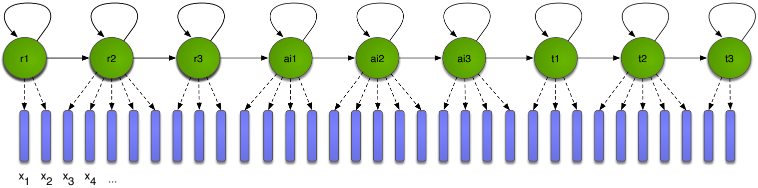
Recap: the HMM



- A *generative model* for the sequence $X = (x_1, \ldots, x_T)$
- Discrete states q_t are unobserved
- q_{t+1} is conditionally independent of $q_1, \ldots, q_{t-1}$, given q_t
- Observations x_t are conditionally independent of each other, given q_t.

HMMs for ASR

The three-state left-to-right topology for phones:



Computing likelihoods with the HMM

Joint likelihood of X and $Q = (q_1, \ldots, q_T)$:

$$P(X, Q|\lambda) = P(q_1)P(\mathrm{x}_1|q_1)P(q_2|q_1)P(\mathrm{x}_2|q_2)\ldots \quad (1)$$

$$= P(q_1)P(\mathrm{x}_1|q_1)\prod_{t=2}^{T} P(q_t|q_{t-1})P(\mathrm{x}_t|q_t) \quad (2)$$

$P(q_t)$ denotes the initial occupancy probability of each state

HMM parameters

The parameters of the model, λ, are given by:

- Transition probabilities $a_{kj} = P(q_{t+1} = j | q_t = k)$
- Observation probabilities $b_j(\mathrm{x}) = P(\mathrm{x} | q = j)$

The three problems of HMMs

Working with HMMs requires the solution of three problems:

1. **Likelihood** Determine the overall likelihood of an observation sequence $X = (x_1, \ldots, x_t, \ldots, x_T)$ being generated by a known HMM topology, $\mathcal{M}$.
   $\rightarrow$ the *forward algorithm*
2. **Decoding and alignment** Given an observation sequence and an HMM, determine the most probable hidden state sequence
   $\rightarrow$ the *Viterbi algorithm*
3. **Training** Given an observation sequence and an HMM, find the state occupation probabilities, in order to find the best HMM parameters λ
   $\rightarrow$ the *forward-backward* and *EM* algorithms

Viterbi algorithm

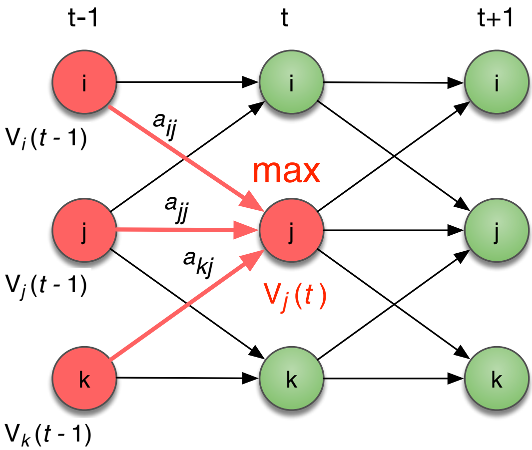
- Instead of finding the likelihood over all possible state sequences, as we do in the Forward algorithm, just consider just the most probable path:

$$P^*(X|\mathcal{M}) = \max P(X, Q|\mathcal{M})$$

- Define likelihood of the most probable partial path in state j at time t, $V_j(t)$
- If we are performing decoding or forced alignment, then only the most likely path is needed
- We need to keep track of the states that make up this path by keeping a sequence of *backpointers* to enable a Viterbi *backtrace*: the backpointer for each state at each time indicates the previous state on the most probable path

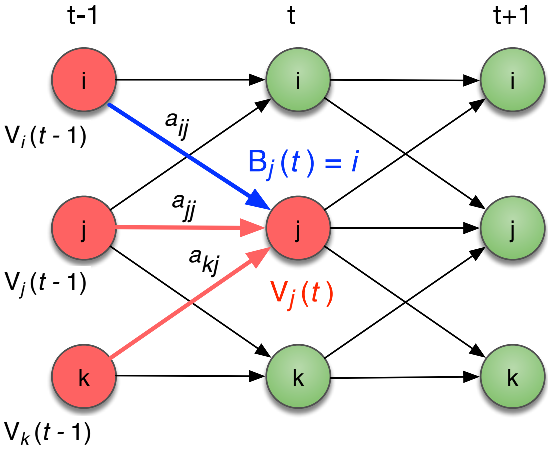
Viterbi Recursion

$$V_j(t) = \max_i V_i(t-1) a_{ij} b_j(x_t)$$



Viterbi Recursion

Backpointers to the previous state on the most probable path



2. Decoding: The Viterbi algorithm

- Initialisation

$$V_0(0) = 1$$
$$V_j(0) = 0 \qquad \text{if } j \neq 0$$
$$B_j(0) = 0$$

- Recursion

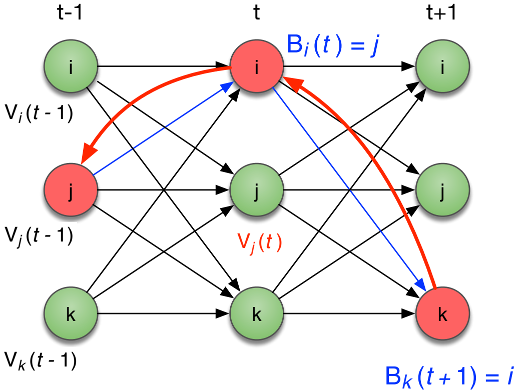
$$V_j(t) = \max_{i=0}^{J} V_i(t-1) a_{ij} b_j(\mathrm{x}_t)$$
$$B_j(t) = \arg\max_{i=0}^{J} V_i(t-1) a_{ij} b_j(\mathrm{x}_t)$$

- Termination

$$V_E = \max_{i=1}^{J} V_i(T) a_{iE}$$
$$B_E = \arg\max_{i=1}^{J} V_i(T) a_{iE}$$

Viterbi Backtrace

Backtrace to find the state sequence of the most probable path



3. Training: Forward-Backward algorithm

- Goal: Efficiently estimate the parameters of an HMM λ from an observation sequence X and known HMM topology $\mathcal{M}$:
- Parameters λ:
  - Transition probabilities $a_{kj} = P(q_{t+1} = j | q_t = k)$
  - Observation probabilities $b_j(\mathrm{x}) = P(\mathrm{x} | q = j)$
- Maximum likelihood training: find the parameters that maximise

$$
\begin{aligned}
F_{\mathrm{ML}}(\lambda) &= \log P(X | \mathcal{M}, \lambda) \\
&= \log \sum_{Q \in \mathcal{Q}} P(X, Q | \mathcal{M}, \lambda)
\end{aligned}
$$

Viterbi Training

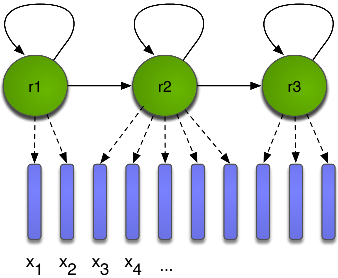
- If we knew the state-time alignment, then each observation feature vector could be assigned to a specific state
- A state-time alignment can be obtained using the most probable path obtained by Viterbi decoding
- Maximum likelihood estimate of a_{ij}, if $C(\, i \rightarrow j\,)$ is the count of transitions from i to j

$$\hat{a}_{ij} = \frac{C(\, i \rightarrow j\,)}{\sum_k C(\, i \rightarrow k\,)}$$

- Define indicator variable $z_{jt} = 1$ if the HMM is in state j at time t, and $z_{jt} = 0$ otherwise. If we knew the state-time alignment, this variable would be observed, and we could use it to obtain the standard maximum likelihood estimates for the mean of the observation probability distribution:

$$\hat{\mu}_j = \frac{\sum_t z_{jt} \boldsymbol{x}_t}{\sum_t z_{jt}}$$

Example



$$a_{11} = \frac{1}{2} \qquad a_{22} = \frac{4}{5} \qquad a_{33} = \frac{2}{3}$$

$$a_{12} = \frac{1}{2} \qquad a_{23} = \frac{1}{5} \qquad a_{3E} = \frac{1}{3}$$

EM Algorithm

- Viterbi training is an approximation—we would like to consider *all* possible paths
- In this case rather than having a hard state-time alignment we estimate a probability
- *State occupation probability*: The probability $\gamma_j(t)$ of occupying state j at time t given the sequence of observations.
- We can use this for an iterative algorithm for HMM training: the EM algorithm
- Application of EM algorithm to HMMs is called 'Baum-Welch algorithm

EM Algorithm

If we have some initial parameters λ_0 and we want to find new parameters to maximise the likelihood $F_{\text{ML}}(\lambda)$, then we can instead maximise

$$\sum_{Q \in \mathcal{Q}} P(Q|X, \mathcal{M}, \lambda_0) \log P(X, Q|\mathcal{M}, \lambda)$$

- **E-step** estimate the state occupation probabilities given the current parameters (Expectation)
- **M-step** re-estimate the HMM parameters based on the estimated state occupation probabilities (Maximisation)

Why does this work? See next lecture.

Backward probabilities

- To estimate the state occupation probabilities we need to define (recursively) another set of probabilities—the *Backward probabilities*

$$\beta_j(t) = p(\mathrm{x}_{t+1}, \ldots, \mathrm{x}_T \mid q_t = j, \mathcal{M})$$

  The probability of future observations given that the HMM is in state j at time t
- These can be recursively computed (going backwards in time)
  - Initialisation

$$\beta_i(T) = a_{iE}$$

  - Recursion

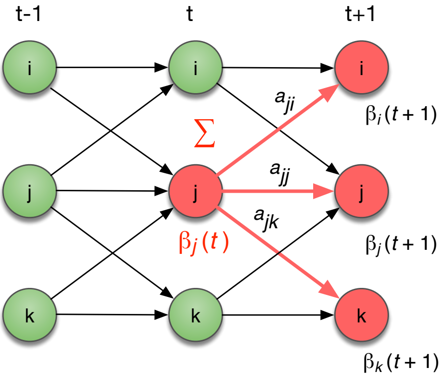
$$\beta_i(t) = \sum_{j=1}^{J} a_{ij} b_j(\mathrm{x}_{t+1}) \beta_j(t+1) \quad \text{for } t = T-1, \ldots, 1$$

  - Termination

$$p(\mathrm{X} \mid \mathcal{M}) = \beta_0(0) = \sum_{j=1}^{J} a_{0j} b_j(\mathrm{x}_1) \beta_j(1) = \alpha_E$$

Backward Recursion

$$\beta_j(t) = p(x_{t+1}, \ldots, x_T \mid q_t = j, \mathcal{M}) = \sum_{j=1}^{J} a_{ij} b_j(x_{t+1}) \beta_j(t+1)$$



State Occupation Probability

- The **state occupation probability** $\gamma_j(t)$ is the probability of occupying state j at time t given the sequence of observations
- Express in terms of the forward and backward probabilities:

$$\gamma_j(t) = P(q_t = j \,|\, \mathsf{X}, \mathcal{M}) = \frac{1}{\alpha_E} \alpha_j(\,t\,)\beta_j(\,t\,)$$

recalling that $p(\mathsf{X} \,|\, \mathcal{M}) = \alpha_E$

- Since

$$\begin{aligned}
\alpha_j(t)\beta_j(t) &= p(\mathsf{x}_1, \ldots, \mathsf{x}_t, q_t = j \,|\, \mathcal{M}) \\
&\qquad p(\mathsf{x}_{t+1}, \ldots, \mathsf{x}_T \,|\, q_t = j, \mathcal{M}) \\
&= p(\mathsf{x}_1, \ldots, \mathsf{x}_t, \mathsf{x}_{t+1}, \ldots, \mathsf{x}_T, q_t = j \,|\, \mathcal{M}) \\
&= p(\mathsf{X}, q_t = j \,|\, \mathcal{M})
\end{aligned}$$

$$P(q_t = j \,|\, \mathsf{X}, \mathcal{M}) = \frac{p(\mathsf{X}, q_t = j \,|\, \mathcal{M})}{p(\mathsf{X} \,|\, \mathcal{M})}$$

Re-estimation of transition probabilities

- Similarly to the state occupation probability, we can estimate $\xi_{i,j}(t)$, the probability of being in i at time t and j at $t+1$, given the observations:

$$\begin{aligned}
\xi_{i,j}(t) &= P(q_t = i, q_{t+1} = j \,|\, \mathrm{X}, \mathcal{M}) \\
&= \frac{p(q_t = i, q_{t+1} = j, \mathrm{X} \,|\, \mathcal{M})}{p(\mathrm{X} \,|\, \mathcal{M})} \\
&= \frac{\alpha_i(t) a_{ij} b_j(\mathrm{x}_{t+1}) \beta_j(t+1)}{\alpha_E}
\end{aligned}$$

- We can use this to re-estimate the transition probabilities

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

- See next lecture for re-estimation of obervation probabilities $b_j(x)$

Pulling it all together

- Iterative estimation of HMM parameters using the EM algorithm. At each iteration
  - E step: For all time-state pairs
    1. Recursively compute the forward probabilities $\alpha_j(t)$ and backward probabilities $\beta_j(t)$
    2. Compute the state occupation probabilities $\gamma_j(t)$ and $\xi_{i,j}(t)$
  - M step: Based on the estimated state occupation probabilities re-estimate the HMM parameters: transition probabilities a_{ij} and parameters of the obervation probabilities, $b_j(x)$
- The application of the EM algorithm to HMM training is sometimes called the Forward-Backward algorithm or Baum-Welch algorithm

Extension to a corpus of utterances

- We usually train from a large corpus of R utterances
- If x_t^r is the t th frame of the r th utterance X^r then we can compute the probabilities $\alpha_j^r(t)$, $\beta_j^r(t)$, $\gamma_j^r(t)$ and $\xi_{i,j}^r(t)$ as before
- The re-estimates are as before, except we must sum over the R utterances, ie:

$$\hat{a}_{ij} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \xi_{i,j}^r(t)}{\sum_{r=1}^{R} \sum_{k=1}^{J} \sum_{t=1}^{T} \xi_{i,k}^r(t)}$$

- In addition, we usually employ "*embedded training*", in which fine tuning of phone labelling with "*forced Viterbi alignment*" or forced alignment is involved. (For details see Section 9.7 in Jurafsky and Martin's SLP)

Summary: HMMs

- HMMs provide a generative model for statistical speech recognition
- Three key problems
  1. Computing the overall likelihood: the Forward algorithm
  2. Decoding the most likely state sequence: the Viterbi algorithm
  3. Estimating the most likely parameters: the EM (Forward-Backward) algorithm
- Solutions to these problems are tractable due to the two key HMM assumptions
  1. Conditional independence of observations given the current state
  2. Markov assumption on the states

References: HMM algorithms

* Jurafsky and Martin (2008). *Speech and Language Processing* (2nd ed.): section 9.7. (Also 9.5, 9.6, 9.8 for introduction to decoding)
- Gales and Young (2007). "The Application of Hidden Markov Models in Speech Recognition", *Foundations and Trends in Signal Processing*, **1** (3), 195–304: sections 1, 2.1, 2.2.