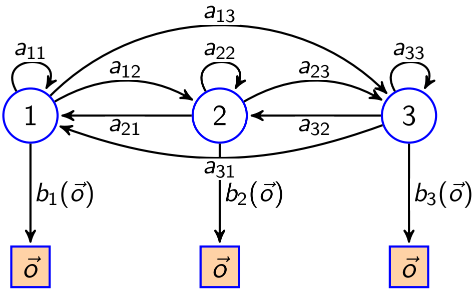
Lecture 15: A tutorial on hidden Markov models and selected applications in speech recognition, part 2

Mark Hasegawa-Johnson

ECE 537, Fall 2022

Outline

Hidden Markov Model



1. Start in state $q_t = i$ with pmf π_i.
2. Generate an observation, $\vec{o}$, with pdf $b_i(\vec{o})$.
3. Transition to a new state, $q_{t+1} = j$, according to pmf a_{ij}.
4. Repeat.

The Three Problems for an HMM

1. **Recognition:** Given two different HMMs, λ_1 and λ_2, and an observation sequence O. Which HMM was more likely to have produced O? In other words, $p(O|\lambda_1) > p(O|\lambda_2)$?
2. **Segmentation:** What is $p(q_t = i|O, \lambda)$?
3. **Training:** Given an initial HMM λ, and an observation sequence O, can we find $\bar{\lambda}$ such that $p(O|\bar{\lambda}) > p(O|\lambda)$?

The Forward Algorithm

Definition: $\alpha_t(i) \equiv p(\vec{o}_1, \ldots, \vec{o}_t, q_t = i|\lambda)$. Computation:

1. **Initialize:**

$$\alpha_1(i) = \pi_i b_i(\vec{o}_1), \quad 1 \le i \le N$$

2. **Iterate:**

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(\vec{o}_t), \quad 1 \le j \le N,\ 2 \le t \le T$$

3. **Terminate:**

$$p(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

The Backward Algorithm

Definition: $\beta_t(i) \equiv p(\vec{o}_{t+1}, \ldots, \vec{o}_T | q_t = i, \lambda)$. Computation:

1. **Initialize:**

$$\beta_T(i) = 1, \quad 1 \le i \le N$$

2. **Iterate:**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\vec{o}_{t+1}) \beta_{t+1}(j), \quad 1 \le i \le N, \; 1 \le t \le T-1$$

3. **Terminate:**

$$p(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(\vec{o}_1) \beta_1(i)$$

Segmentation

1. **The State Posterior:**

$$\gamma_t(i) = p(q_t = i|O,\lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{k=1}^{N}\alpha_t(k)\beta_t(k)}$$

2. **The Segment Posterior:**

$$\begin{aligned}\xi_t(i,j) &= p(q_t = i, q_{t+1} = j|O,\lambda)\\ &= \frac{\alpha_t(i)a_{ij}b_j(\vec{o}_{t+1})\beta_{t+1}(j)}{\sum_{k=1}^{N}\sum_{\ell=1}^{N}\alpha_t(k)a_{k\ell}b_\ell(\vec{o}_{t+1})\beta_{t+1}(\ell)}\end{aligned}$$

The Three Problems for an HMM

1. **Recognition:** Given two different HMMs, λ_1 and λ_2, and an observation sequence O. Which HMM was more likely to have produced O? In other words, $p(O|\lambda_1) > p(O|\lambda_2)$?
2. **Segmentation:** What is $p(q_t = i|O, \lambda)$?
3. **Training:** Given an initial HMM λ, and an observation sequence O, can we find $\bar{\lambda}$ such that $p(O|\bar{\lambda}) > p(O|\lambda)$?

Outline

Maximum Likelihood Training

Suppose we're given several observation sequences of the form $O = [\vec{o}_1, \ldots, \vec{o}_T]$. Suppose, also, that we have some initial guess about the values of the model parameters (our initial guess doesn't have to be very good). Maximum likelihood training means we want to compute a new set of parameters, $\bar{\lambda} = \left\{\bar{\pi}_i, \bar{a}_{ij}, \bar{b}_j(\vec{o})\right\}$ that maximize $p(O|\bar{\lambda})$.

1. **Initial State Probabilities:** Find values of $\bar{\pi}_i$, $1 \leq i \leq N$, that maximize $p(O|\bar{\lambda})$.
2. **Transition Probabilities:** Find values of $\bar{a}_{ij}$, $1 \leq i, j \leq N$, that maximize $p(O|\bar{\lambda})$.
3. **Observation Probabilities:** Learn $\bar{b}_j(\vec{o})$. What does that mean, actually?

Learning the Observation Probabilities

There are four typical ways of modeling the observations:

1. **Discrete:** Vector quantize $\vec{o}$, using some VQ method. Suppose $\vec{o}$ is the k^{th} codevector; then we just need to learn $b_j(k)$ such that

$$b_j(k) \geq 0, \quad \sum_{k=0}^{K-1} b_j(k) = 1$$

2. **Gaussian:** Model $b_j(k)$ as a Gaussian or mixture Gaussian, and learn its parameters.
3. **Neural Net:** Model $b_j(k)$ as a neural net, and learn its parameters.

For now, assume discrete observations.

Maximum Likelihood Training

Given discrete observations, we need to learn the following parameters:

1. **Initial State Probabilities:** $\bar{\pi}_i$ such that

$$\bar{\pi}_i \geq 0, \quad \sum_{i=1}^{N} \bar{\pi}_i = 1$$

2. **Transition Probabilities:** $\bar{a}_{ij}$ such that

$$\bar{a}_{ij} \geq 0, \quad \sum_{j=1}^{N} \bar{a}_{ij} = 1$$

3. **Observation Probabilities:** $\bar{b}_j(k)$ such that

$$\bar{b}_j(k) \geq 0, \quad \sum_{k=1}^{K} \bar{b}_j(k) = 1$$

Maximum Likelihood Training with Known State Sequence

Impossible assumption: Suppose that we actually know the state sequences, $Q = [q_1, \ldots, q_T]$, matching with each observation sequence $O = [\vec{o}_1, \ldots, \vec{o}_T]$. Then what would be the maximum-likelihood parameters?

Maximum Likelihood Training with Known State Sequence

Our goal is to find $\lambda = \{\pi_i, a_{ij}, b_j(k)\}$ in order to maximize

$$\begin{aligned}
\mathcal{L}(\lambda) &= \ln p(Q, O|\lambda) \\
&= \ln \pi_{q_1} + \ln b_{q_1}(o_1) + \ln a_{q_1,q_2} + b_{q_2}(o_2) + \ldots \\
&= \ln \pi_{q_1} + \sum_{i=1}^{N} \left(\sum_{j=1}^{N} n_{ij} \ln a_{ij} + \sum_{k=1}^{K} m_{ik} \ln b_i(k) \right)
\end{aligned}$$

where

- n_{ij} is the number of times we saw $(q_t = i, q_{t+1} = j)$,
- m_{ik} is the number of times we saw $(q_t = i, k_t = k)$

Maximum Likelihood Training with Known State Sequence

$$\mathcal{L}(\lambda) = \ln \pi_{q_1} + \sum_{i=1}^{N} \left(\sum_{j=1}^{N} n_{ij} \ln a_{ij} + \sum_{k=1}^{K} m_{ik} \ln b_i(k) \right)$$

When we differentiate that, we find the following derivatives:

$$\frac{\partial \mathcal{L}}{\partial \pi_i} = \begin{cases} \frac{1}{\pi_i} & i = q_1 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = \frac{n_{ij}}{a_{ij}}$$

$$\frac{\partial \mathcal{L}}{\partial b_j(k)} = \frac{m_{jk}}{b_j(k)}$$

These derivatives are **never** equal to zero! What went wrong?

Maximum Likelihood Training with Known State Sequence

Here's the problem: we forgot to include the constraints
$\sum_i \pi_i = 1$, $\sum_j a_{ij} = 1$, and $\sum_k b_j(k) = 1$!
We can include the constraints using a Lagrangian optimization.

Maximum Likelihood Training with Known State Sequence

The Lagrangian, $\mathcal{J}(\lambda)$, is the thing we want to optimize ($\mathcal{L}(\lambda)$), plus the things that should be zero, each of which is multiplied by an arbitrary constant called a **Lagrange multiplier**:

$$\mathcal{J}(\lambda) = \ln \pi_{q_1} + \sum_{i=1}^{N} \left(\sum_{j=1}^{N} n_{ij} \ln a_{ij} + \sum_{k=1}^{K} m_{ik} \ln b_i(k) \right)$$

$$+\kappa \left(1 - \sum_{i=1}^{N} \pi_i \right) + \sum_{i=1}^{N} \mu_i \left(1 - \sum_{j=1}^{N} a_{ij} \right) + \sum_{i=1}^{N} \nu_i \left(1 - \sum_{k=1}^{M} b_j(k) \right)$$

1. First solve for the parameters as functions of the Lagrange multipliers.
2. Second, set the Lagrange multipliers equal to whatever value will zero out the constraints.

Maximum Likelihood Training with Known State Sequence

Step 1: Solve for the parameters as functions of the Lagrange multipliers. If we set

$$\frac{\partial \mathcal{J}(\lambda)}{\partial \pi_i} = \frac{\partial \mathcal{J}(\lambda)}{\partial a_{ij}} = \frac{\partial \mathcal{J}(\lambda)}{\partial b_{jk}} = 0,$$

we get:

$$\bar{\pi}_i = \begin{cases} \frac{1}{\kappa} & i = q_1 \\ 0 & \text{otherwise} \end{cases}, \quad \bar{a}_{ij} = \frac{n_{ij}}{\mu_i}, \quad \bar{b}_j(k) = \frac{m_{jk}}{\nu_j}$$

Maximum Likelihood Training with Known State Sequence

Step 2: Set the Lagrange multipliers to whatever value zeros out the constraints:

$$\bar{\pi}_i = \begin{cases} 1 & i = q_1 \\ 0 & \text{otherwise} \end{cases}$$

$$\bar{a}_{ij} = \frac{n_{ij}}{\sum_{j=1}^{N} n_{ij}}$$

$$\bar{b}_j(k) = \frac{m_{jk}}{\sum_{k=1}^{M} m_{jk}}$$

Maximum Likelihood Training with Known State Sequence

Using the Lagrange multiplier method, we can show that the maximum likelihood parameters for the HMM are:

1. **Initial State Probabilities:**

$$\bar{\pi}_i = \frac{\#\text{ state sequences that start with } q_1 = i}{\#\text{ state sequences in training data}}$$

2. **Transition Probabilities:**

$$\bar{a}_{ij} = \frac{\#\text{ frames in which } q_{t-1} = i, q_t = j}{\#\text{ frames in which } q_{t-1} = i}$$

3. **Observation Probabilities:**

$$\bar{b}_j(k) = \frac{\#\text{ frames in which } q_t = j, k_t = k}{\#\text{ frames in which } q_t = j}$$

Outline

Expectation Maximization

When the true state sequence is unknown, then we can't maximize the likelihood $p(O, Q|\bar{\lambda})$ directly. Instead, we maximize Baum's auxilary function:

$$Q(\lambda, \bar{\lambda}) = \sum_Q p(Q|O, \lambda) \ln p(O, Q|\bar{\lambda})$$

This method has two key advantages:

- The maximizer of $Q(\lambda, \bar{\lambda})$ can be computed analytically.
- Baum proved that, regardless of the value of λ,

$$\max_{\bar{\lambda}} Q(\lambda, \bar{\lambda}) \quad \Rightarrow \quad P(O|\bar{\lambda}) \geq P(O|\lambda)$$

Baum-Welch Re-Estimation: Overview

1. Start out by setting λ to any arbitrary initial value.
2. Iterate:
   1. Find $\bar{\lambda} = \operatorname{argmax} Q(\lambda, \bar{\lambda})$
   2. Set $\lambda = \bar{\lambda}$
3. Stop when $P(O|\lambda)$ stops (quickly) increasing.

Calculating the Baum Auxiliary

The Baum auxiliary is:

$$
\begin{aligned}
Q(\lambda,\bar{\lambda}) &= \sum_Q p(Q|O,\lambda)\ln p(O,Q|\bar{\lambda}) \\
&= \sum_{i=1}^{N} p(q_1=i|O,\lambda)\ln \bar{\pi}_i \\
&+ \sum_{t=1}^{T-1}\sum_{i=1}^{N}\sum_{j=1}^{N} p(q_t=i,q_{t+1}=j|O,\lambda)\ln \bar{a}_{ij} \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{k=1}^{M} p(q_t=i,o_t=k|O,\lambda)\ln \bar{b}_j(k)
\end{aligned}
$$

Now we need to find those three probabilities.

Calculating the Baum Auxiliary

First: $p(q_1 = i|O, \lambda)$. We already know this one! It's

$$p(q_1 = i|O, \lambda) = \gamma_1(i)$$

Calculating the Baum Auxiliary

Second: $p(q_t = i, q_{t+1} = i|O, \lambda)$. This one is a two-step state posterior, calculated similar to γ. Rabiner uses the letter ξ for this probability:

$$
\begin{aligned}
p(q_t = i, q_{t+1} = j|O, \lambda) &= \frac{p(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)} \\
&= \frac{\alpha_t(i) a_{ij} b_j(\vec{o}_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} \\
&\equiv \xi_t(i,j)
\end{aligned}
$$

Calculating the Baum Auxiliary

Finally: $p(q_t = i, o_t = k | O, \lambda)$.

$$p(q_t = i, o_t = k | O, \lambda) = \begin{cases} p(q_t = i | O, \lambda) & o_t = k \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} \gamma_t(i) & o_t = k \\ 0 & \text{otherwise} \end{cases}$$

Calculating the Baum Auxiliary

Putting it all together,

$$
\begin{aligned}
Q(\lambda, \bar{\lambda}) &= \sum_{Q} p(Q|O, \lambda) \ln p(O, Q|\bar{\lambda}) \\
&= \sum_{i=1}^{N} \gamma_1(i) \ln \bar{\pi}_i \\
&+ \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j) \ln \bar{a}_{ij} \\
&+ \sum_{i=1}^{N} \sum_{k=1}^{M} \sum_{t:o_t=k} \gamma_t(i) \ln \bar{b}_j(k)
\end{aligned}
$$

Maximizing the Baum Auxiliary

Now let's create a Lagrangian:

$$\mathcal{J}(\lambda) = \sum_{i=1}^{N} \gamma_1(i) \ln \bar{\pi}_i + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j) \ln \bar{a}_{ij} + \sum_{i=1}^{N} \sum_{k=1}^{M} \sum_{t:o_t=k} \gamma_t(i) \ln \bar{b}$$

$$+\kappa \left(1 - \sum_{i=1}^{N} \bar{\pi}_i\right) + \sum_{i=1}^{N} \mu_i \left(1 - \sum_{j=1}^{N} \bar{a}_{ij}\right) + \sum_{i=1}^{N} \nu_i \left(1 - \sum_{k=1}^{M} \bar{b}_j(k)\right)$$

...differentiate it, and set the derivative equal to zero.

Maximizing the Baum Auxiliary

Here's the result of that differentiation:

1. **Initial State Probabilities:**

$$\bar{\pi}_i = \frac{\gamma_1(i)}{\sum_{i'=1}^{N} \gamma_1(i')}$$

2. **Transition Probabilities:**

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{j'=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j')}$$

3. **Observation Probabilities:**

$$\bar{b}_j(k) = \frac{\sum_{t:o_t=k} \gamma_t(i)}{\sum_{i'=1}^{N} \sum_{t:o_t=k} \gamma_t(i')}$$

Maximizing the Baum Auxiliary

If you look closely at the equations on the previous slide, you will see that they are just like the known-state case, except that instead of counting **known** state frequencies, we now compute **expected** state frequencies!

1. **Initial State Probabilities:**

$$\bar{\pi}_i = \frac{E\left[\#\text{ state sequences that start with } q_1 = i\right]}{\#\text{ state sequences in training data}}$$

2. **Transition Probabilities:**

$$\bar{a}_{ij} = \frac{E\left[\#\text{ frames in which } q_{t-1} = i, q_t = j\right]}{E\left[\#\text{ frames in which } q_{t-1} = i\right]}$$

3. **Observation Probabilities:**

$$\bar{b}_j(k) = \frac{E\left[\#\text{ frames in which } q_t = j, o_t = k\right]}{E\left[\#\text{ frames in which } q_t = j\right]}$$

Outline

Baum-Welch with Gaussian Probabilities

The requirement that we vector-quantize the observations is a problem. It means that we can't model the observations very precisely.
It would be better if we could model the observation likelihood, $b_j(\vec{o})$, as a probability density in the space $\vec{o} \in \Re^D$. One way is to use a parameterized function that is guaranteed to be a properly normalized pdf. For example, a Gaussian:

$$b_i(\vec{o}) = \mathcal{N}\left(\vec{o}; \vec{\mu}_i, \Sigma_i\right)$$

Calculating the Baum Auxiliary

The Baum auxiliary is now:

$$
\begin{aligned}
Q(\lambda, \bar{\lambda}) &= \sum_{Q} p(Q|O, \lambda) \ln p(O, Q|\bar{\lambda}) \\
&= \sum_{i=1}^{N} \gamma_1(i) \ln \bar{\pi}_i \\
&+ \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j) \ln \bar{a}_{ij} \\
&+ \sum_{i=1}^{N} \sum_{t=1}^{T} \gamma_t(i) \ln \mathcal{N}\left(\vec{o}_t; \vec{\mu}_i, \Sigma_i\right)
\end{aligned}
$$

Maximizing the Baum Auxiliary

When we maximize the Baum auxiliary, we get:

$$\bar{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)\vec{o}_t}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\bar{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(\vec{o}_t - \vec{\mu}_i)(\vec{o}_t - \vec{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}$$

Maximizing the Baum Auxiliary

Notice the similarity to what we would do if the states were known:

- **Known states:** μ_i is the sample mean of the observations, Σ_i is their sample variance.
- **Known states:**
  - μ_i is the weighted average, where the weights are $\gamma_t(i)$.
  - Σ_i is the weighted sample variance, where the weights are $\gamma_t(i)$.

Outline

The Baum-Welch Algorithm: Initial and Transition Probabilities

1. **Initial State Probabilities:**

$$\bar{\pi}_i = \frac{\sum_{sequences} \gamma_1(i)}{\# \text{ sequences}}$$

2. **Transition Probabilities:**

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{j=1}^{N} \sum_{t=1}^{T-1} \xi_t(i,j)}$$

The Baum-Welch Algorithm: Observation Probabilities

1. **Discrete Observation Probabilities:**

$$\bar{b}_j(k) = \frac{\sum_{t:\vec{o}_t=k} \gamma_t(j)}{\sum_t \gamma_t(j)}$$

2. **Gaussian Observation PDFs:**

$$\bar{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)\vec{o}_t}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\bar{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(\vec{o}_t - \vec{\mu}_i)(\vec{o}_t - \vec{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)}$$