HMMs for Acoustic Modeling

Lecture 1b

CS 753, 2021

Instructor: Preethi Jyothi, IITB

Recap: HMMs

Recap: HMMs

What are (first-order) HMMs?

Recap: HMMs

✓ What are (first-order) HMMs?

Recap: HMMs

✓ What are (first-order) HMMs?

What are the simplifying assumptions governing HMMs?

Recap: HMMs

- ✓ What are (first-order) HMMs?
- ✓ What are the simplifying assumptions governing HMMs?

Recap: HMMs

✓ What are (first-order) HMMs?

✓ What are the simplifying assumptions governing HMMs?

What is the forward algorithm? What is it used to compute?

Computing Likelihood: Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

Recap: HMMs

- ✓ What are (first-order) HMMs?
- ✓ What are the simplifying assumptions governing HMMs?
- ✓ What is the forward algorithm? What is it used to compute?

Computing Likelihood: Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

Recap: HMMs

- ✓ What are (first-order) HMMs?
- ✓ What are the simplifying assumptions governing HMMs?
- ✓ What is the forward algorithm? What is it used to compute?

Computing Likelihood: Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

What is the Viterbi algorithm? What is it used to compute?

Decoding: Given as input an HMM $\lambda = (A, B)$ and a sequence of observations $O = o_1, o_2, ..., o_T$, find the most probable sequence of states $Q = q_1 q_2 q_3 \ldots q_T$.

Recap: HMMs

- ✓ What are (first-order) HMMs?
- ✓ What are the simplifying assumptions governing HMMs?
- ✓ What is the forward algorithm? What is it used to compute?

  Computing Likelihood: Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O|\lambda)$.

- ✓ What is the Viterbi algorithm? What is it used to compute?

  Decoding: Given as input an HMM $\lambda = (A, B)$ and a sequence of observations $O = o_1, o_2, ..., o_T$, find the most probable sequence of states $Q = q_1 q_2 q_3 \ldots q_T$.

Learning in HMMs

Problem 1 (Likelihood):	Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O\|\lambda)$.
Problem 2 (Decoding):	Given an observation sequence O and an HMM $\lambda = (A, B)$, discover the best hidden state sequence Q.
Problem 3 (Learning):	Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

Learning: Given an observation sequence O and the set of possible states in the HMM, learn the HMM parameters A and B.

EM for general HMMs: Baum-Welch algorithm (1972)

(*predates the general formulation of EM (1977)*)

Learning in HMMs

Problem 1 (Likelihood):	Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O\|\lambda)$.
Problem 2 (Decoding):	Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.
Problem 3 (Learning):	Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

Learning: Given an observation sequence O and the set of possible states in the HMM, learn the HMM parameters A and B.

Forward-backward or **Baum-Welch (EM)** algorithm

EM for general HMMs: Baum-Welch algorithm (1972)

(*predates the general formulation of EM (1977)*)

What is the EM Algorithm?

Expectation Maximization (EM) Algorithm

Expectation Maximization (EM) Algorithm

EM is an iterative algorithm used to compute Maximum Likelihood (ML) (or Maximum A posteriori MAP) estimates in the presence of missing or hidden data.

Expectation Maximization (EM) Algorithm

EM is an iterative algorithm used to compute Maximum Likelihood (ML) (or Maximum A posteriori MAP) estimates in the presence of missing or hidden data.

E step: Estimate hidden variables given the observations and current estimates of the model parameters.

Expectation Maximization (EM) Algorithm

EM is an iterative algorithm used to compute Maximum Likelihood (ML) (or Maximum A posteriori MAP) estimates in the presence of missing or hidden data.

Two step iterative algorithm:

E step: Estimate hidden variables given the observations and current estimates of the model parameters.

Expectation Maximization (EM) Algorithm

EM is an iterative algorithm used to compute Maximum Likelihood (ML) (or Maximum A posteriori MAP) estimates in the presence of missing or hidden data.

Two step iterative algorithm:

E step: Estimate hidden variables given the observations and current estimates of the model parameters.

M step: Estimating model parameters by maximising the likelihood function using estimates of the hidden data from the E step.

EM Algorithm: Fitting Parameters to Data

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples x_i, i=1, …, N

Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i; \theta)$ $= \sum_i \log \underline{\sum_z \Pr(x_i, z; \theta)}$

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples x_i, i=1, …, N Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i; \theta)$

Initial parameters: θ^0 (x is observed and z is hidden)

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples $x_{i,}$ i=1, …, N Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i; \theta)$

Initial parameters: θ^0 (x is observed and z is hidden)

Iteratively compute θ^ℓ as follows that optimises an auxiliary function $Q(\theta, \theta^{\ell-1})$:

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples x_i, i=1, …, N

Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\underline{\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i; \theta)}$

Initial parameters: θ^0 (x is observed and z is hidden)

Iteratively compute θ^ℓ as follows that optimises an auxiliary function $Q(\theta, \theta^{\ell-1})$:

$$\underline{Q(\theta, \theta^{\ell-1})} = \sum_{i=1}^{N} \sum_{z} \Pr(z | x_i; \theta^{\ell-1}) \log \Pr(x_i, z; \theta)$$

$$\theta^\ell = \arg\max_\theta \underline{Q(\theta, \theta^{\ell-1})}$$

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples x_i, i=1, …, N

Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i;\theta)$

Initial parameters: θ^0 (x is observed and z is hidden)

Iteratively compute θ^ℓ as follows that optimises an auxiliary function $Q(\theta, \theta^{\ell-1})$:

$$Q(\theta, \theta^{\ell-1}) = \sum_{i=1}^{N} \sum_{z} \Pr(z|x_i;\theta^{\ell-1}) \log \Pr(x_i, z;\theta)$$

$$\theta^\ell = \arg\max_\theta Q(\theta, \theta^{\ell-1})$$

Estimate θ^ℓ cannot get worse over iterations because for all θ:

$$\mathcal{L}(\theta) - \mathcal{L}(\theta^{\ell-1}) \geq \underline{Q(\theta, \theta^{\ell-1}) - Q(\theta^{\ell-1}, \theta^{\ell-1})}$$ ~~or~~ stay the same

The EM algorithm is guaranteed to increase the likelihood at every iteration & hence guaranteed to converge

EM Algorithm: Fitting Parameters to Data

Observed data: i.i.d samples x_i, i=1, …, N Hidden data: Denoted by z

Goal: Find $\arg\max_\theta \mathcal{L}(\theta)$ where $\mathcal{L}(\theta) = \sum_{i=1}^{N} \log \Pr(x_i; \theta)$

Initial parameters: θ^0 (x is observed and z is hidden)

Iteratively compute θ^ℓ as follows that optimises an auxiliary function $Q(\theta, \theta^{\ell-1})$:

$$Q(\theta, \theta^{\ell-1}) = \sum_{i=1}^{N} \sum_{z} \Pr(z|x_i; \theta^{\ell-1}) \log \Pr(x_i, z; \theta)$$

$$\theta^\ell = \arg\max_\theta Q(\theta, \theta^{\ell-1})$$

Estimate θ^ℓ cannot get worse over iterations because for all θ:

$$\mathcal{L}(\theta) - \mathcal{L}(\theta^{\ell-1}) \geq Q(\theta, \theta^{\ell-1}) - Q(\theta^{\ell-1}, \theta^{\ell-1})$$

EM is guaranteed to converge to a local optimum or saddle points [Wu83]

[Optional] EM: How do we arrive at the auxiliary function?

[Optional] EM: How do we arrive at the auxiliary function?

$$Q(\theta, \theta^{\ell-1}) = \sum_{i=1}^{N} \sum_{z} \Pr(z|x_i;\theta^{\ell-1}) \log \Pr(x_i, z; \theta)$$

Jensen's inequality

$\log \sum_i \lambda_i x_i \geq \sum_i \lambda_i \log x_i$

$\lambda_i \geq 0, \quad \sum_i \lambda_i = 1$

$$L(\theta) > L(\theta_n) \Rightarrow \underbrace{L(\theta) - L(\theta_n)}_{\text{want to maximize this delta}} = \log \sum_z P(x, z|\theta) - \log P(x|\theta_n)$$

$P(z|x,\theta_n)$ can serve as a λ

$$L(\theta) - L(\theta_n) = \log \sum_z P(x|z,\theta) P(z|\theta) \frac{P(z|x,\theta_n)}{P(z|x,\theta_n)} - \log P(x|\theta_n)$$

$$\geq \underbrace{\sum_z P(z|x,\theta_n) \log \frac{P(x|z,\theta)P(z|\theta)}{P(z|x,\theta_n)} - \log P(x|\theta_n)}_{\beta(\theta,\theta_n)}$$

Choose a θ that maximizes $L(\theta) \geq L(\theta_n) + \beta(\theta, \theta_n)$

$\geq Q(\theta, \theta_n)$

$$\theta_{n+1} = \arg\max_\theta Q(\theta, \theta_n) = \arg\max_\theta \underbrace{\sum_z P(z|x,\theta_n) \log P(x,z|\theta)}_{E_{z|x,\theta_n} \log P(x,z|\theta)}$$

How do we use EM for HMM parameter estimation?

Forward and Backward Probabilities

Baum-Welch algorithm iteratively estimates transition & observation probabilities and uses these values to derive even better estimates.

Forward and Backward Probabilities

Baum-Welch algorithm iteratively estimates transition & observation probabilities and uses these values to derive even better estimates.

Require two probabilities to compute estimates for the transition and observation probabilities:

Forward and Backward Probabilities

Baum-Welch algorithm iteratively estimates transition & observation probabilities and uses these values to derive even better estimates.

Require two probabilities to compute estimates for the transition and observation probabilities:

1. **Forward** probability: Recall $\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$

Forward and Backward Probabilities

Baum-Welch algorithm iteratively estimates transition & observation probabilities and uses these values to derive even better estimates.

Require two probabilities to compute estimates for the transition and observation probabilities:

1. **Forward** probability: Recall $\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j | \lambda)$
2. **Backward** probability: $\beta_t(i) = P(o_{t+1}, o_{t+2} \ldots o_T | q_t = i, \lambda)$

Backward probability

1. **Initialization:**

$$\beta_T(i) \;=\; 1, \quad 1 \leq i \leq N$$

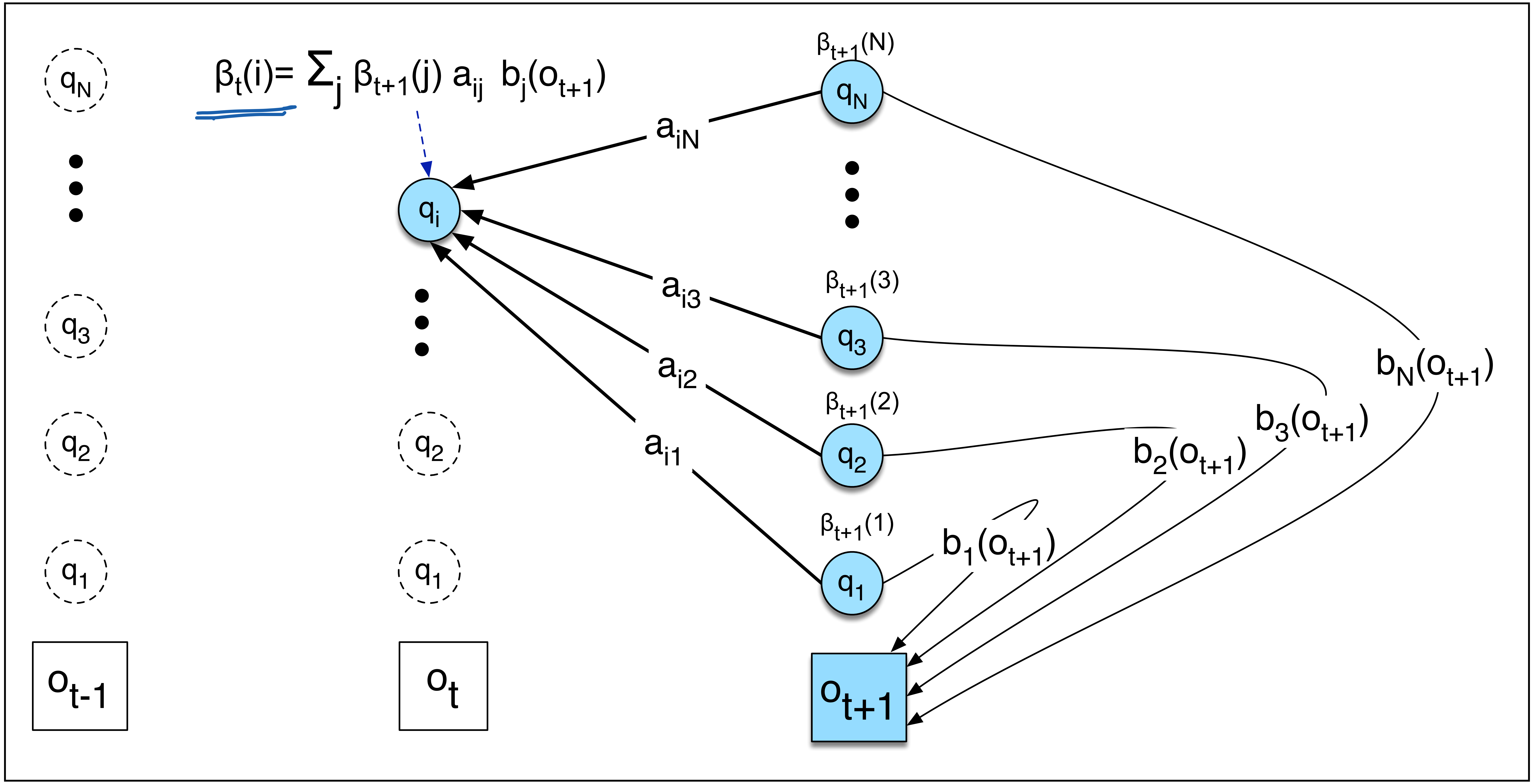
2. **Recursion**

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j), \quad 1 \leq i \leq N, 1 \leq t < T$$

3. **Termination:**

$$P(O|\lambda) = \sum_{j=1}^{N} \pi_j\, b_j(o_1)\, \beta_1(j)$$

Visualising backward probability computation



1. Baum-Welch: Estimating a_{ij}

We need to define $\xi_t(i, j)$ to estimate a_{ij}

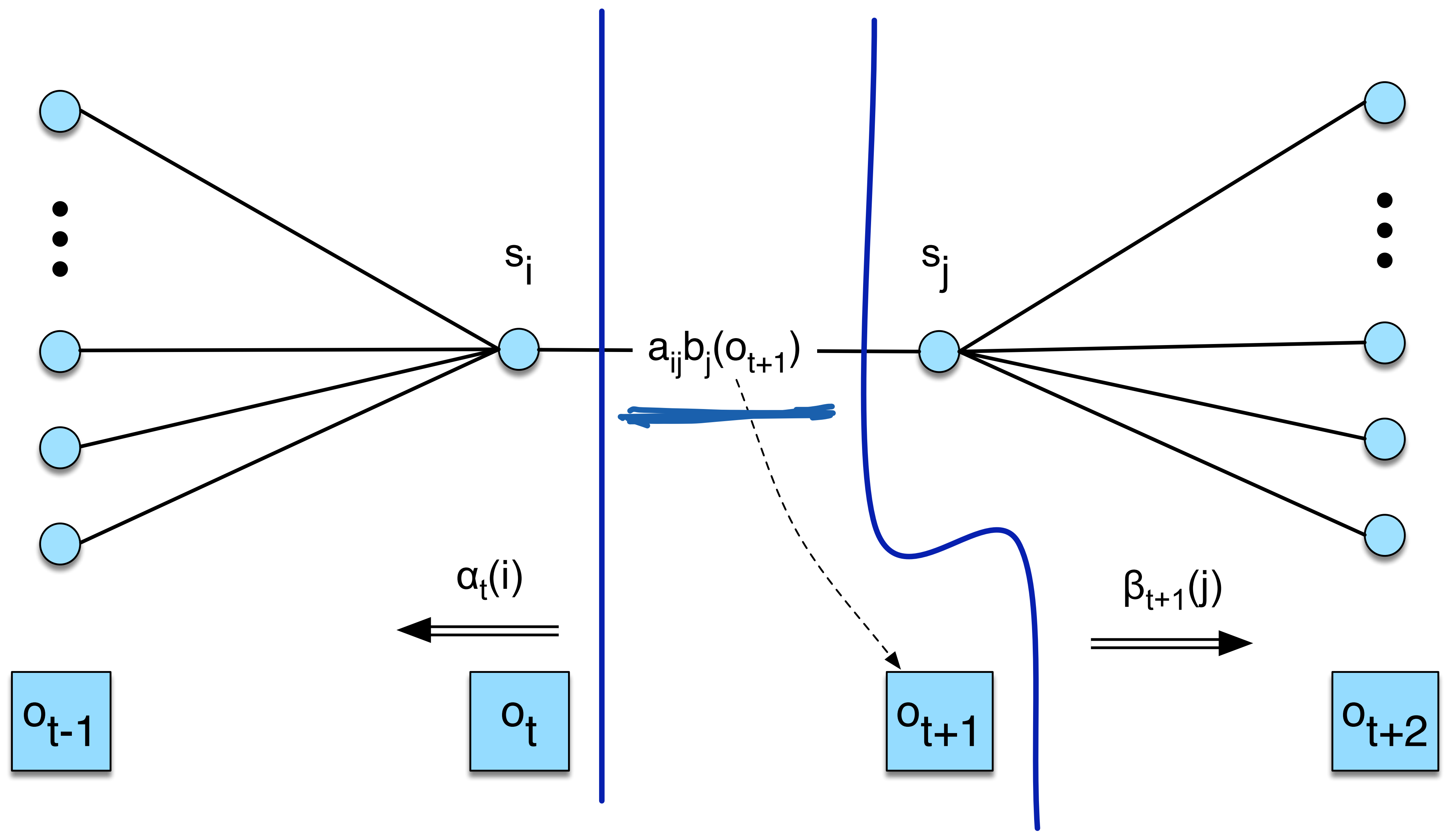
where $\xi_t(i, j) = P(q_t = i, q_{t+1} = j | O, \lambda)$

1. Baum-Welch: Estimating a_{ij}

We need to define $\xi_t(i,j)$ to estimate a_{ij}

where $\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda)$

which works out to be $\xi_t(i,j) = \dfrac{\alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$ $= \dfrac{P(q_t = i, q_{t+1} = j, O | \lambda)}{P(O|\lambda)}$



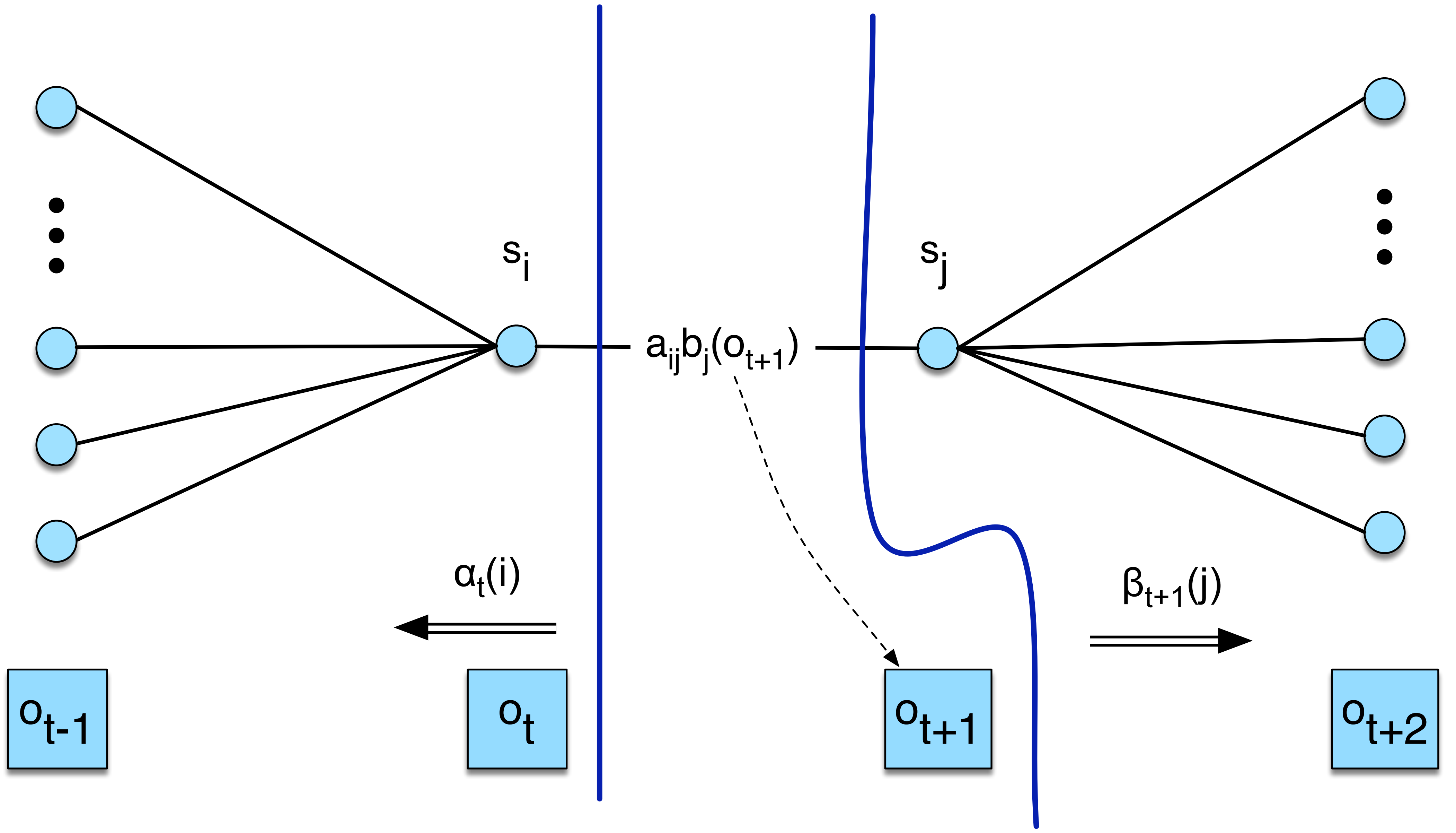
1. Baum-Welch: Estimating a_{ij}

We need to define $\xi_t(i,j)$ to estimate a_{ij}

where $\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda)$

which works out to be $\xi_t(i,j) = \dfrac{\alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$

Then, $\hat{a}_{ij} = \dfrac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$



2. Baum-Welch: Estimating $b_j(v_k)$

$z_t \in V$

We need to define $\gamma_t(j)$ to estimate $b_j(v_k)$

where $\gamma_t(j) = P(q_t = j | O, \lambda)$

2. Baum-Welch: Estimating $b_j(v_k)$

We need to define $\gamma_t(j)$ to estimate $b_j(v_k)$

where $\gamma_t(j) = P(q_t = j|O,\lambda)$ $= \dfrac{P(q_t = j, O|\lambda)}{P(O|\lambda)}$

which works out to be $\gamma_t(j) = \dfrac{\alpha_t(j)\beta_t(j)}{P(O|\lambda)}$

2. Baum-Welch: Estimating $b_j(v_k)$

We need to define $\gamma_t(j)$ to estimate $b_j(v_k)$

where $\gamma_t(j) = P(q_t = j | O, \lambda)$

which works out to be $\gamma_t(j) = \dfrac{\alpha_t(j)\beta_t(j)}{P(O|\lambda)}$

State occupancy probability

2. Baum-Welch: Estimating $b_j(v_k)$

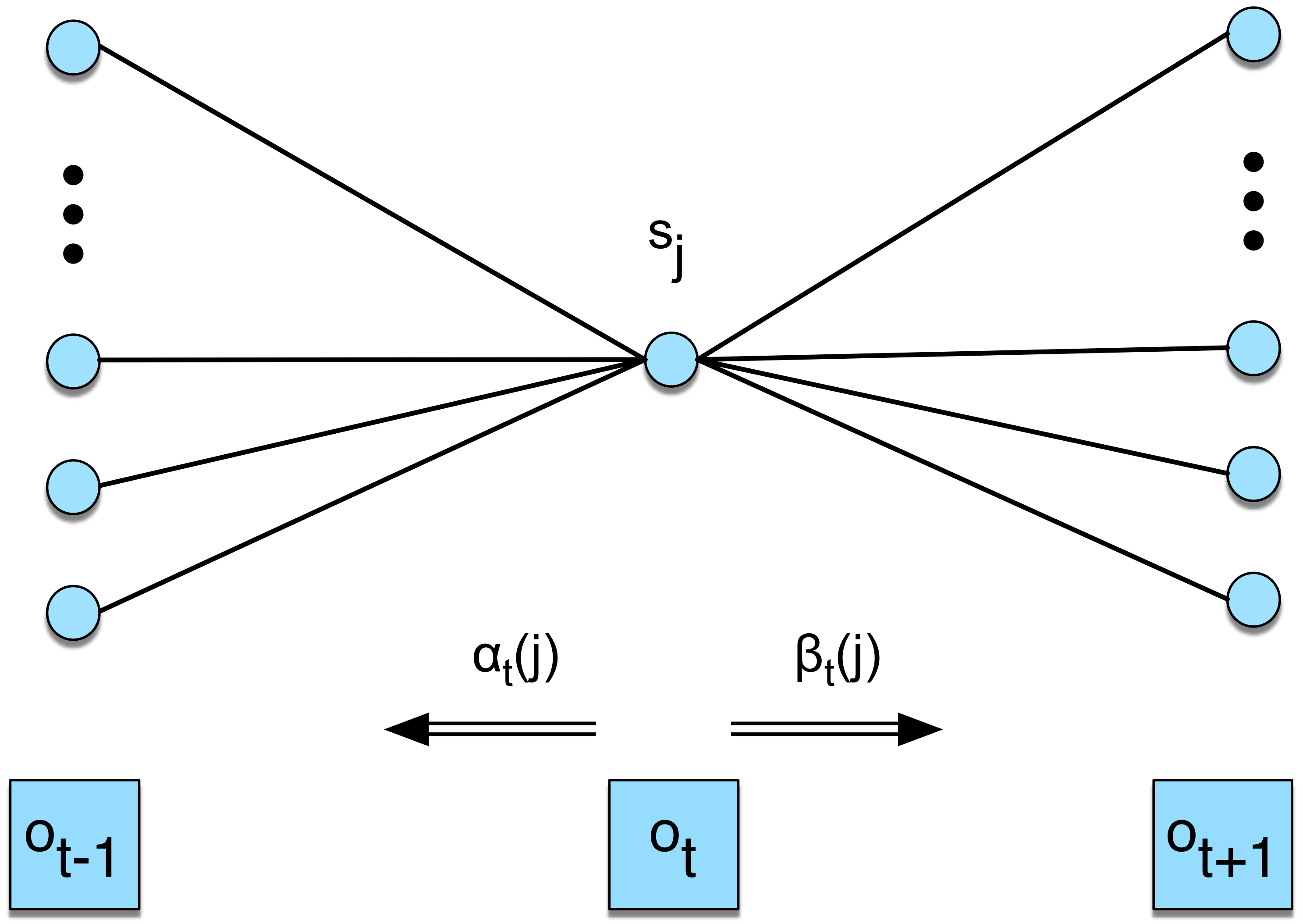
We need to define $\gamma_t(j)$ to estimate $b_j(v_k)$

where $\gamma_t(j) = P(q_t = j | O, \lambda)$

which works out to be $\gamma_t(j) = \dfrac{\overbrace{\alpha_t(j)\beta_t(j)}^{P(q_t = j,\, O | \lambda)}}{P(O|\lambda)}$

State occupancy probability

$P(O|\lambda) \rightarrow \sum_j \alpha_t(j)\beta_t(j)$



2. Baum-Welch: Estimating $b_j(v_k)$

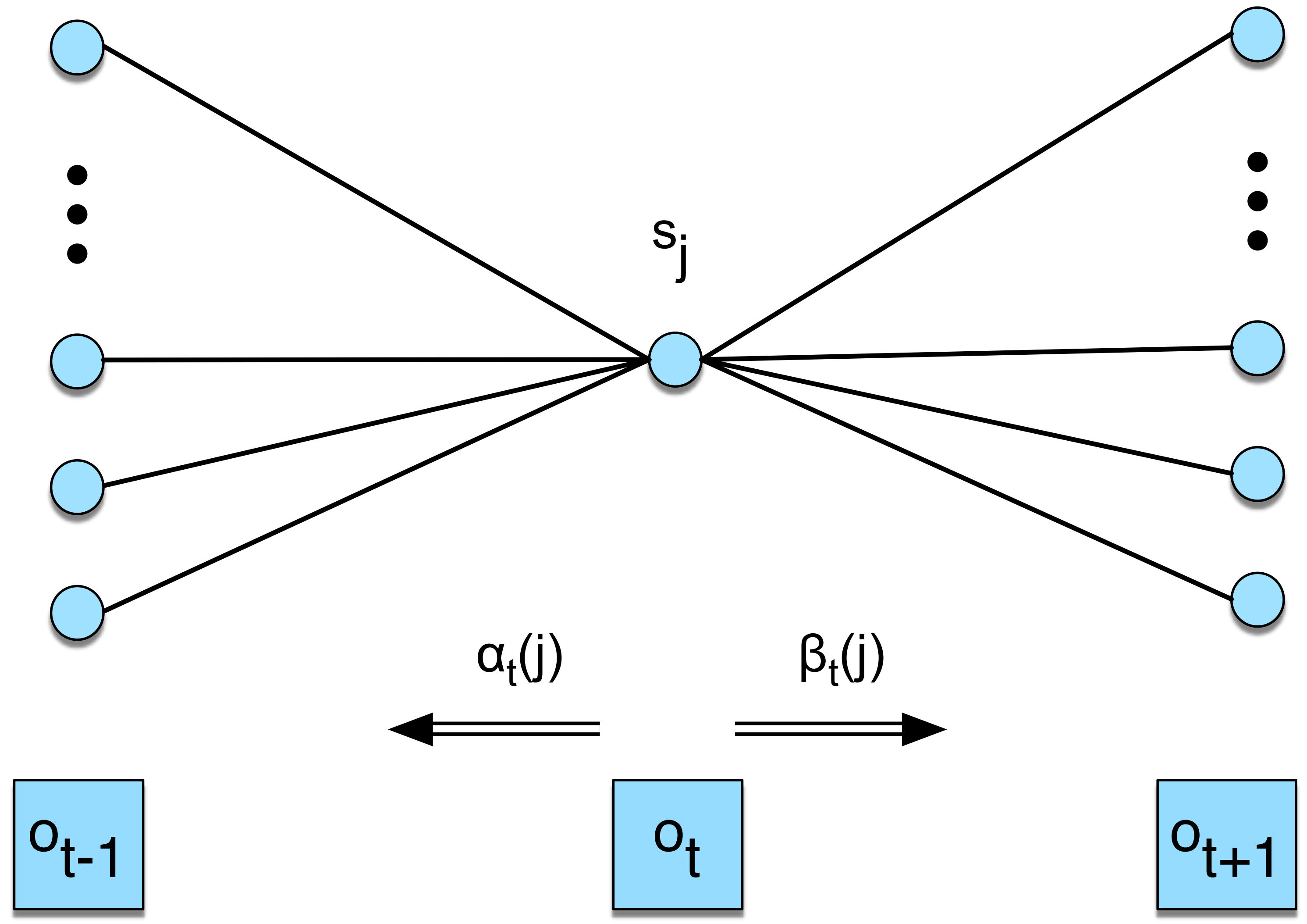
We need to define $\gamma_t(j)$ to estimate $b_j(v_k)$

where $\gamma_t(j) = P(q_t = j | O, \lambda)$

which works out to be $\gamma_t(j) = \dfrac{\alpha_t(j)\beta_t(j)}{P(O|\lambda)}$

State occupancy probability

Then, $\hat{b}_j(v_k) = \dfrac{\sum_{t=1 s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$ for discrete outputs



Bringing it all together: Baum-Welch

Estimating HMM parameters iteratively using the EM algorithm.
For each iteration, do:

Bringing it all together: Baum-Welch

Estimating HMM parameters iteratively using the EM algorithm.
For each iteration, do:

> **E step:** For all time-state pairs, compute the state occupation probabilities $\gamma_t(j)$ and $\xi_t(i, j)$

Bringing it all together: Baum-Welch

Estimating HMM parameters iteratively using the EM algorithm.
For each iteration, do:

E step: For all time-state pairs, compute the state occupation probabilities $\gamma_t(j)$ and $\xi_t(i, j)$

M step: Reestimate HMM parameters, i.e. transition probabilities, observation probabilities, based on the estimates derived in the E step

Baum-Welch algorithm (pseudocode)

function FORWARD-BACKWARD(*observations* of len T, *output vocabulary V*, *hidden state set Q*) **returns** *HMM=(A,B)*

initialize A and B
iterate until convergence
E-step

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(q_F)} \quad \forall\, t \text{ and } j$$

$$\xi_t(i,j) = \frac{\alpha_t(i)\, a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\alpha_T(q_F)} \quad \forall\, t,\ i, \text{ and } j$$

M-step

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

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t. O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

return A, B

Baum-Welch Algorithm as EM

Observed data: *N sequences,* x_i, i=1…N where $x_i \in \mathrm{V}$

Baum-Welch Algorithm as EM

Observed data: *N sequences,* x_i, $i=1\ldots N$ where $x_i \in \mathrm{V}$

Parameters θ : transition matrix *A*, observation probabilities *B*

Baum-Welch Algorithm as EM

Observed data: *N sequences,* x_i, $i=1..N$ where $x_i \in V$

Parameters θ : transition matrix A, observation probabilities B

[EM Iteration, E-step]

Compute quantities involved in $Q(\theta, \theta^{\ell-1})$

$\gamma_{i,t}(j) = \Pr(z_t = j \mid x_i ; \theta^{\ell-1})$

$\xi_{i,t}(j,k) = \Pr(z_t = j, z_{t+1} = k \mid x_i ; \theta^{\ell-1})$

Baum-Welch Algorithm as EM

Observed data: *N sequences,* x_i, $i=1\ldots N$ where $x_i \in \mathrm{V}$

Parameters θ : transition matrix *A*, observation probabilities *B*

[EM Iteration, M-step]

Find θ which maximises $Q(\theta,\theta^{\ell-1})$

Baum-Welch Algorithm as EM

Observed data: *N sequences*, x_i, $i=1\ldots N$ where $x_i \in \mathrm{V}$

Parameters θ : transition matrix A, observation probabilities B

[EM Iteration, M-step]

Find θ which maximises $Q(\theta, \theta^{\ell-1})$

$$A_{j,k} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i-1} \xi_{i,t}(j,k)}{\sum_{i=1}^{N} \sum_{t=1}^{T_i-1} \sum_{k'} \xi_{i,t}(j,k')}$$

$$B_{j,v} = \frac{\sum_{i=1}^{N} \sum_{t:x_{it}=v} \gamma_{i,t}(j)}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j)}$$

Discrete to continuous outputs

We derived Baum-Welch updates for discrete outputs.

However, HMMs in acoustic models emit real-valued vectors as observations.

Use probability density functions to define observation probabilities

Discrete to continuous outputs

We derived Baum-Welch updates for discrete outputs.

However, HMMs in acoustic models emit real-valued vectors as observations.

Use probability density functions to define observation probabilities

If x were 1D values, HMM observation probabilities: $b_j(x) = \mathcal{N}(x \mid \mu_j, \sigma_j^2)$
where μ_j is the mean associated with state j and σ_j^2 is its variance

Discrete to continuous outputs

We derived Baum-Welch updates for discrete outputs.

However, HMMs in acoustic models emit real-valued vectors as observations.

Use probability density functions to define observation probabilities

If x were 1D values, HMM observation probabilities: $b_j(x) = \mathcal{N}(x \mid \mu_j, \sigma_j^2)$
where μ_j is the mean associated with state j and σ_j^2 is its variance

If $\mathbf{x} \in \mathbb{R}^d$, then we use multivariate Gaussians, $b_j(\mathbf{x}) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_j, \Sigma_j)$
where Σ_j is the covariance matrix associated with state j

BW for Gaussian Observation Model

Observed data: *N sequences,* $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1 \ldots N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix *A*, observation prob. **$B = \{(\mu_j, \Sigma_j)\}$ for all j**

BW for Gaussian Observation Model

Observed data: *N sequences,* $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1\ldots N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix *A*, observation prob. $B = \{(\mu_j, \Sigma_j)\}$ for all j

BW for Gaussian Observation Model

Observed data: *N sequences*, $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1\ldots N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix A, observation prob. $B = \{(\mu_j, \Sigma_j)\}$ for all j

[EM Iteration, M-step]

Find θ which maximises $Q(\theta, \theta^{\ell-1})$

A same as with discrete outputs

$$\mu_j = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j) x_{it}}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j)}$$

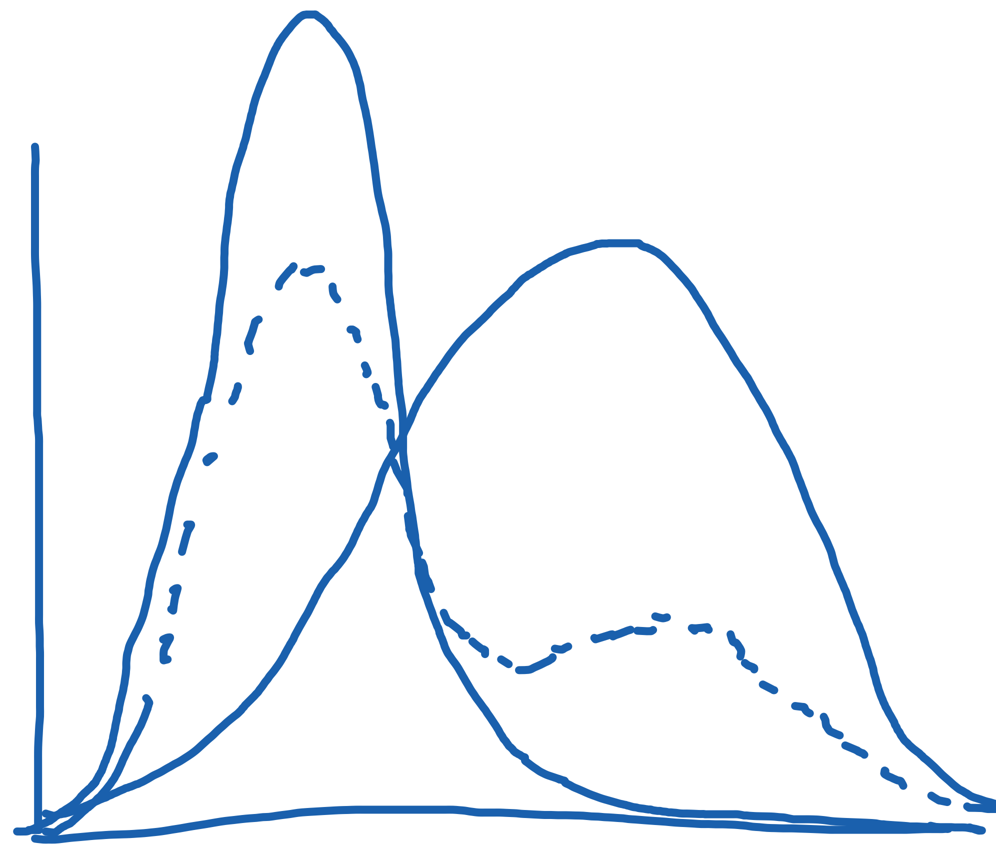
$$\Sigma_j = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j) (x_{it} - \mu_j)(x_{it} - \mu_j)^T}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j)}$$

Gaussian Mixture Model

- Assuming that observations associated with a state follow a Gaussian distribution is too simplistic.

Gaussian Mixture Model

- Assuming that observations associated with a state follow a Gaussian distribution is too simplistic.
- More generally, we use a "mixture of Gaussians" to allow for acoustic vectors associated with a state to be non-Gaussian.



Gaussian Mixture Model

- Assuming that observations associated with a state follow a Gaussian distribution is too simplistic.
- More generally, we use a "mixture of Gaussians" to allow for acoustic vectors associated with a state to be non-Gaussian.
- Instead of $b_j(\mathbf{x}) = \mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}_j, \Sigma_j)$ in the single Gaussian case, $b_j(\mathbf{x})$ can be an M-component mixture model:

$$b_j(\mathbf{x}) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_{jm}, \Sigma_{jm})$$

where c_{jm} is the mixing probability for Gaussian component m of state j

$$\sum_{m=1}^{M} c_{jm} = 1, \quad c_{jm} \geq 0$$

BW for Gaussian Mixture Model

Observed data: *N sequences,* $x_i = (x_{i1}, \ldots, x_{iT_i})$, i=$1 \ldots N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix A, observation prob. $B = \{(\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ for all j,m

BW for Gaussian Mixture Model

Observed data: *N sequences,* $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1\ldots N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix *A*, observation prob. $B = \{(\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ for all j,m

BW for Gaussian Mixture Model

Observed data: *N sequences*, $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1..N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix A, observation prob. $B = \{(\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ for all j,m

[EM Iteration, M-step]

Find θ which maximises $Q(\theta, \theta^{\ell-1})$

$$\mu_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m) x_{it}}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$\Sigma_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)(x_{it} - \mu_{jm})(x_{it} - \mu_{jm})^T}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$c_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \sum_{m'=1}^{M} \gamma_{i,t}(j,m')}$$

BW for Gaussian Mixture Model

Observed data: *N sequences*, $x_i = (x_{i1}, \ldots, x_{iT_i})$, $i=1..N$ where $x_{it} \in \mathbb{R}^d$

Parameters θ : transition matrix A, observation prob. $B = \{(\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ for all j,m

[EM Iteration, M-step]

Find θ which maximises $Q(\theta, \theta^{\ell-1})$

$$\mu_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m) x_{it}}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$\Sigma_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)(x_{it} - \mu_{jm})(x_{it} - \mu_{jm})^T}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$c_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \sum_{m'=1}^{M} \gamma_{i,t}(j,m')}$$

Prob. of component m of state j at time t

Baum Welch: In summary

Baum Welch: In summary

[Every EM Iteration]

Compute $\theta = \{ A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm}) \}$ for all j,k,m

Baum Welch: In summary

[Every EM Iteration]

Compute $\theta = \{ A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm}) \}$ for all j,k,m

$$A_{j,k} = \frac{\sum_{i=1}^{N} \sum_{t=2}^{T_i} \xi_{i,t}(j,k)}{\sum_{i=1}^{N} \sum_{t=2}^{T_i} \sum_{k'} \xi_{i,t}(j,k')}$$

Baum Welch: In summary

[Every EM Iteration]

Compute $\theta = \{ A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm}) \}$ for all j,k,m

$$A_{j,k} = \frac{\sum_{i=1}^{N} \sum_{t=2}^{T_i} \xi_{i,t}(j,k)}{\sum_{i=1}^{N} \sum_{t=2}^{T_i} \sum_{k'} \xi_{i,t}(j,k')}$$

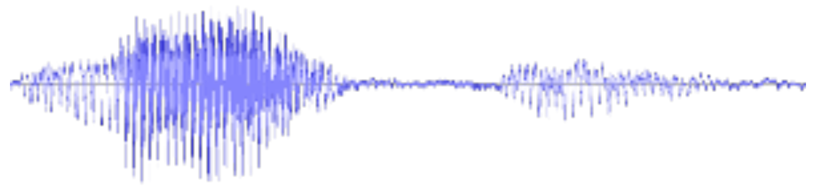
$$\mu_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m) x_{it}}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$\Sigma_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)(x_{it} - \mu_{jm})(x_{it} - \mu_{jm})^T}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}$$

$$c_{jm} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j,m)}{\sum_{i=1}^{N} \sum_{t=1}^{T_i} \sum_{m'=1}^{M} \gamma_{i,t}(j,m')}$$

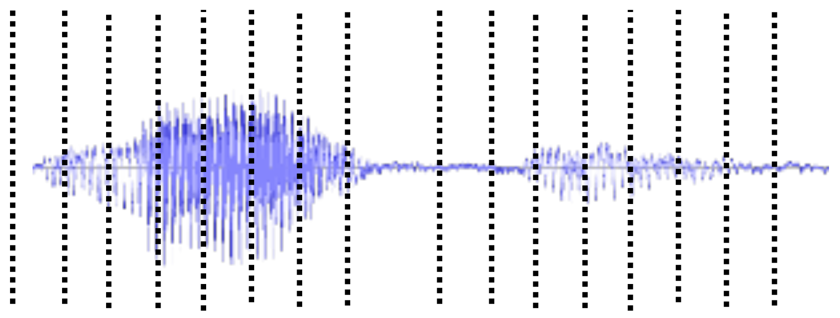
Overall Summary

Training



Overall Summary

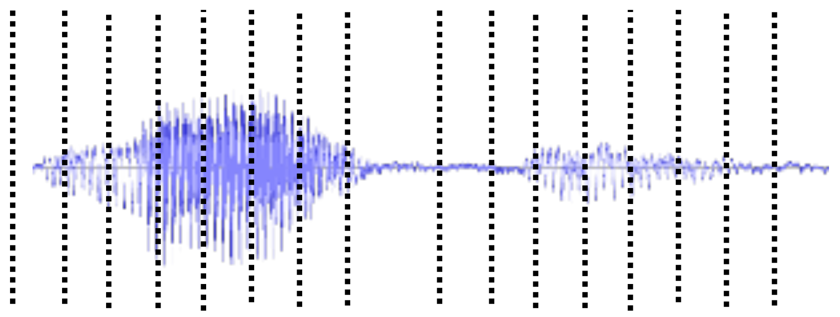
Training



… …

Overall Summary

Training

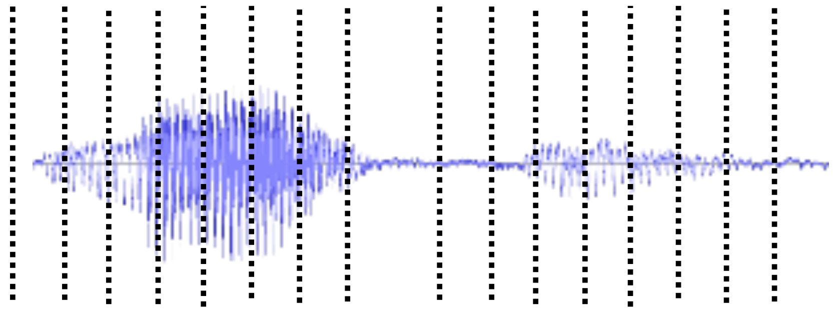


$\rightarrow O_1^1, \ldots, O_{t1}^1$

… …

Overall Summary

Training

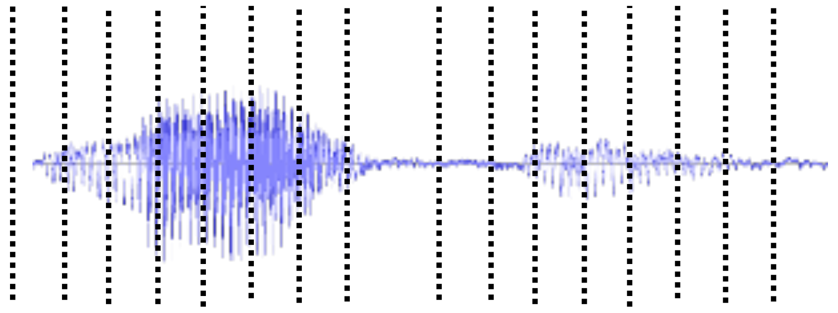


$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

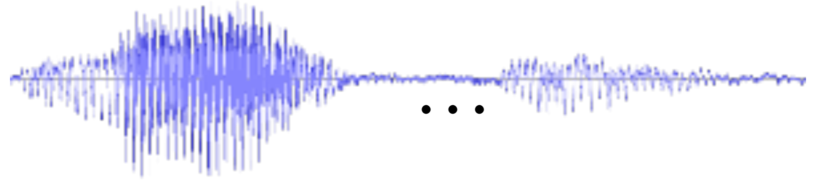
$\ldots$ $\ldots$

Overall Summary

Training

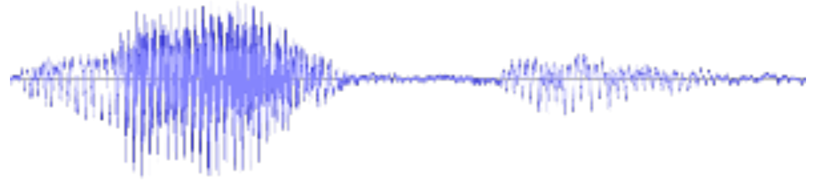


$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$



$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

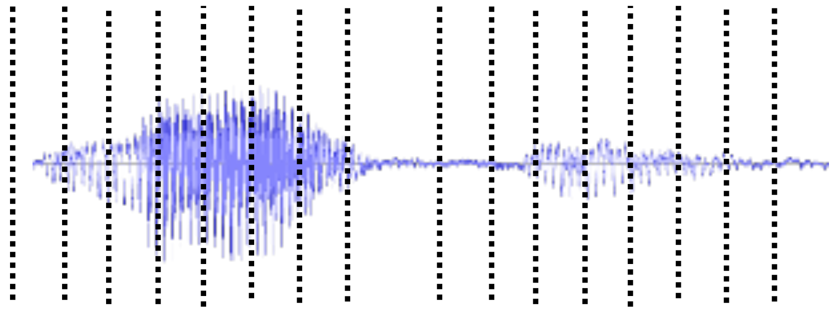
$\vdots$



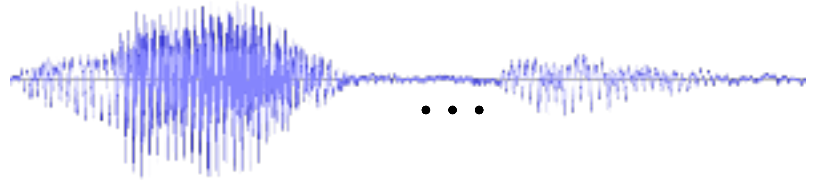
$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Overall Summary

Training

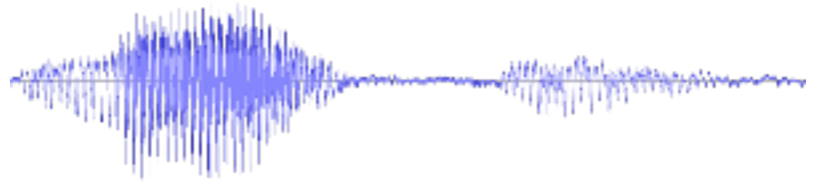


$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$



$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

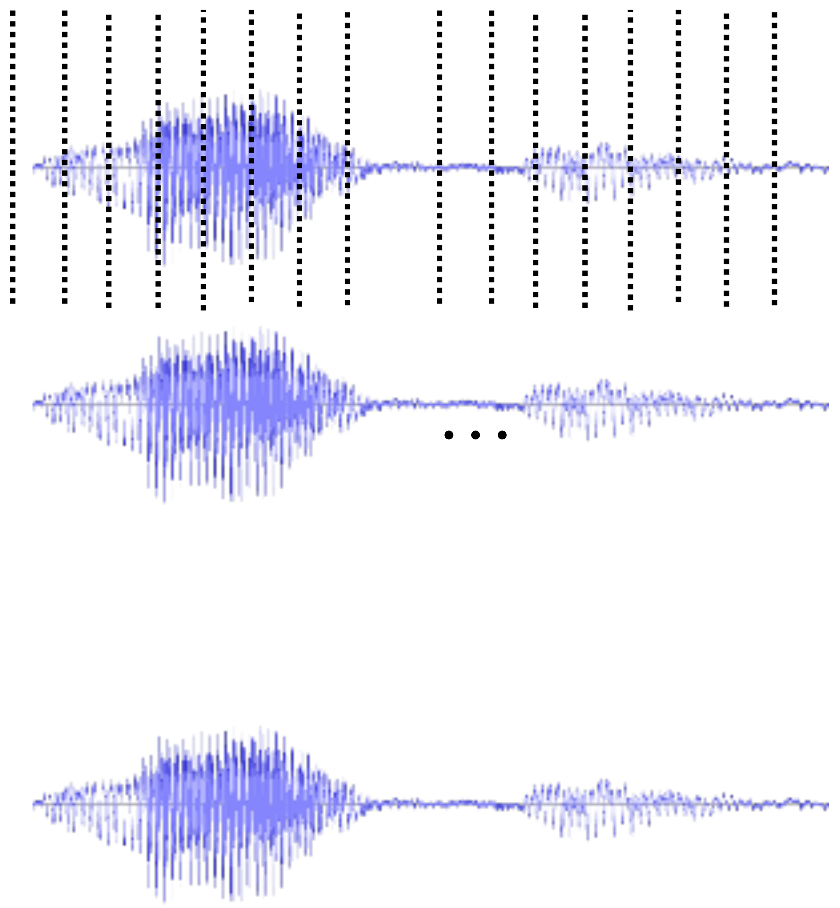


$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

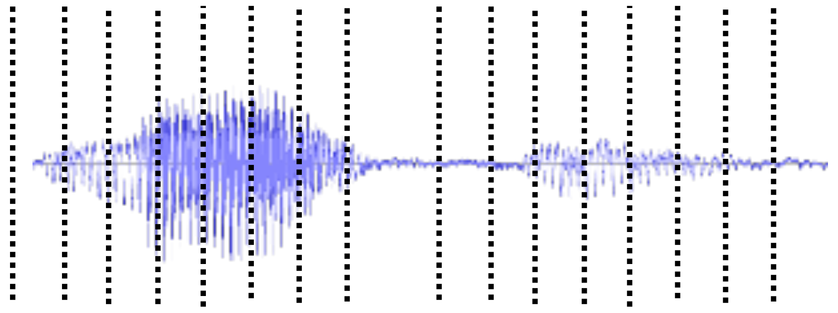
$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

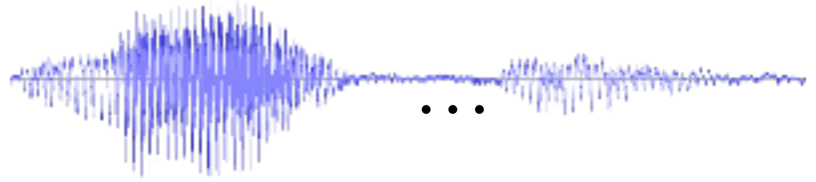
Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of i*th training utterance determined by using a word-to-phone mapping applied to* $w_1^i, \ldots, w_{\ell i}^i$

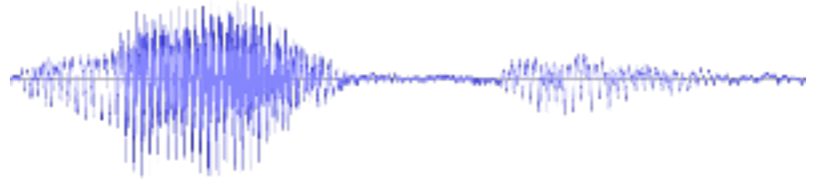
Overall Summary

Training

 $\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

 $\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

 $\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

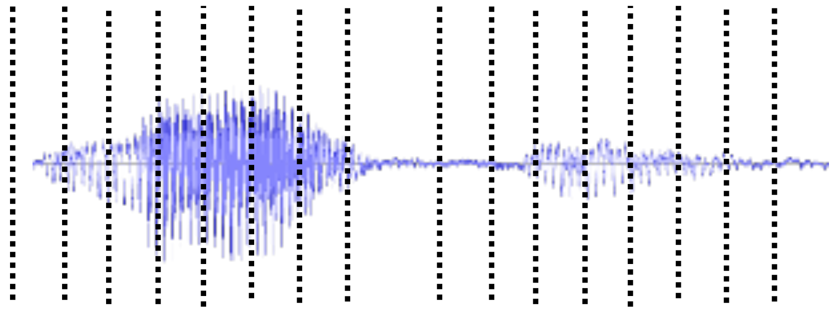
Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

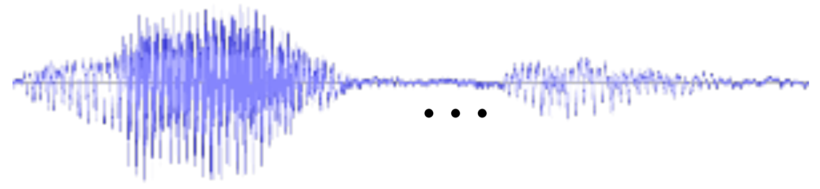
Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Overall Summary

Training

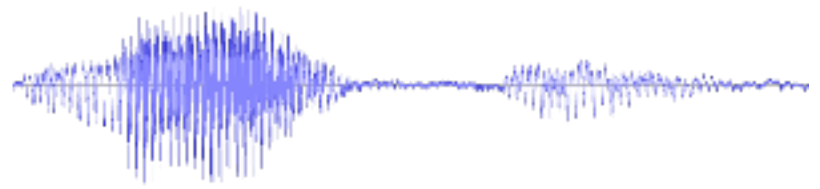


$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$



$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$



$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

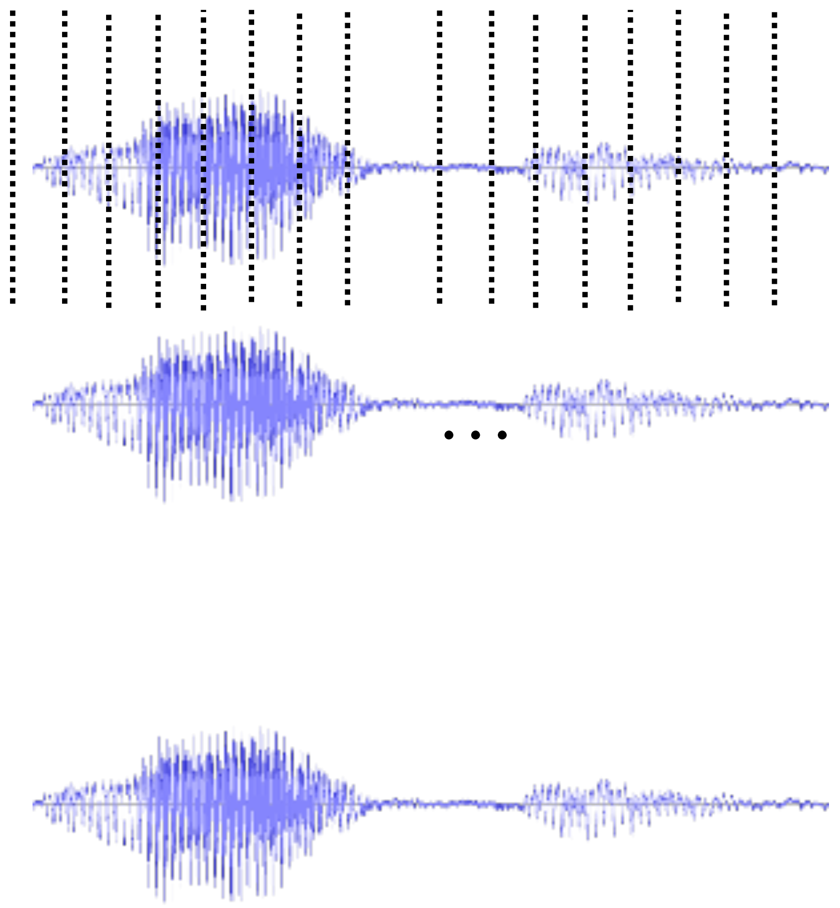
Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

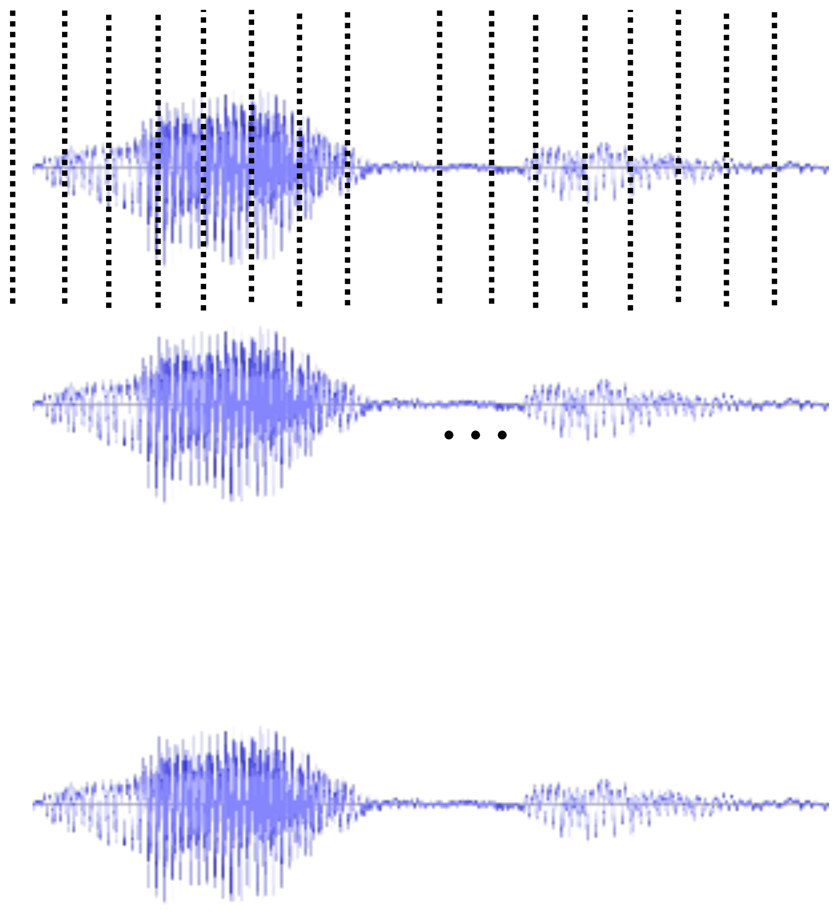
Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Test

$\mathbf{O} = O_1, \ldots, O_T$

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

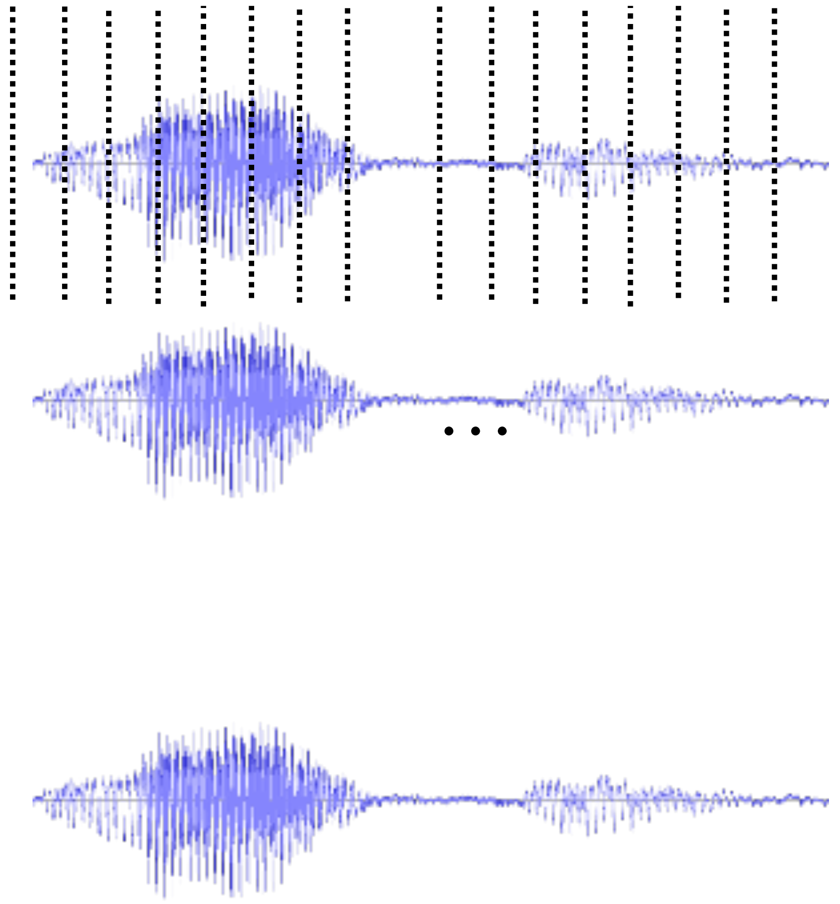
Test

$\mathbf{O} = O_1, \ldots, O_T$

$$W^* = \arg\max_W P(W \mid O)$$

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Test

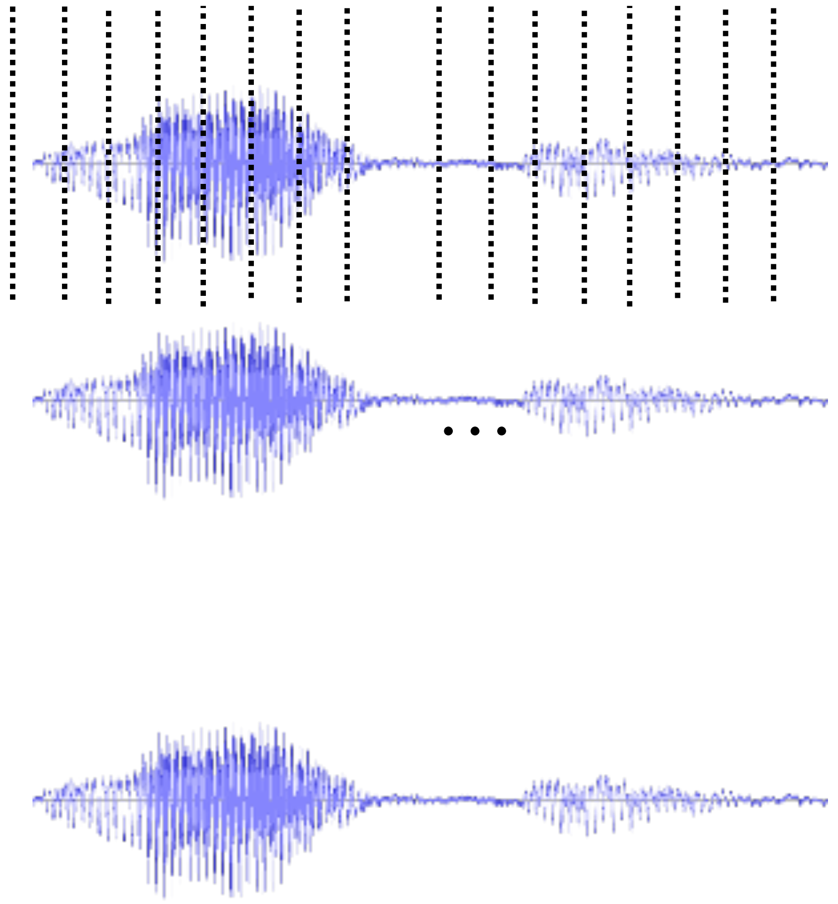
$\mathbf{O} = O_1, \ldots, O_T$

$$W^* = \arg\max_W P(W \mid O)$$

Compute using Viterbi algorithm

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots \rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Test

$\mathbf{O} = O_1, \ldots, O_T$

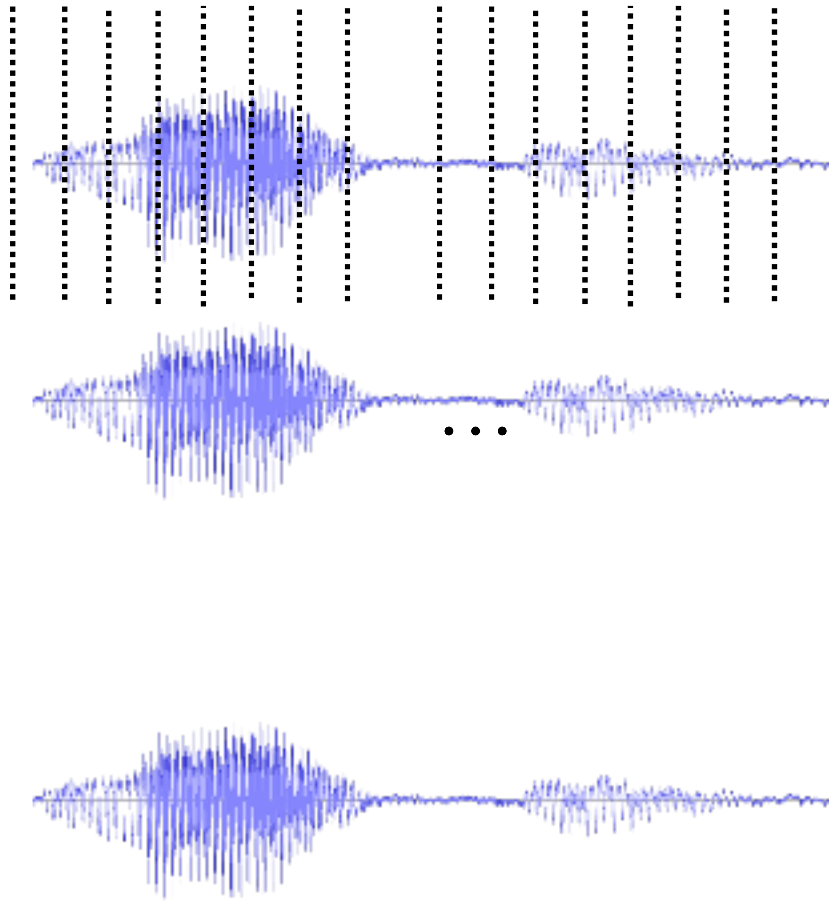
$$W^* = \arg\max_W P(W \mid O)$$

Compute using Viterbi algorithm

Search all possible state sequences arising from all word sequences most likely to have generated $\mathbf{O}$

Overall Summary

Training



$\rightarrow O_1^1, \ldots, O_{t1}^1$ **and** $\mathbf{w}^1 = w_1^1, \ldots, w_{\ell 1}^1$

$\ldots\rightarrow O_1^2, \ldots, O_{t2}^2$ **and** $\mathbf{w}^2 = w_1^2, \ldots, w_{\ell 2}^2$

$\vdots$

$\rightarrow O_1^N, \ldots, O_{tN}^N$ **and** $\mathbf{w}^N = w_1^N, \ldots, w_{\ell N}^N$

Estimate $\theta = \{A_{jk}, (\mu_{jm}, \Sigma_{jm}, c_{jm})\}$ over all phone states. Use Baum-Welch.

HMM of ith training utterance determined by using a word-to-phone mapping applied to $w_1^i, \ldots, w_{\ell i}^i$

Train LM using $\mathbf{w}^1, \ldots, \mathbf{w}^N$

Test

$\mathbf{O} = O_1, \ldots, O_T$

$$W^* = \arg\max_W P(W \mid O)$$

Compute using Viterbi algorithm

Search all possible state sequences arising from all word sequences most likely to have generated $\mathbf{O}$

Computationally intractable for continuous speech! Efficient algorithms in later classes.