

HMMs

Extensions & Practical Issues

Professor Marie Roch

Huang et al.: 8.3 – end of chapter

Continuous-Mixture Density HMMs

- In continuous-mixture HMMs, the discrete alphabet O is replaced with a continuous one of arbitrary dimension.
- This permits us to model multi-dimensional feature data without resorting to vector quantization.

State-dependent pdfs revisited

- Discrete pdf $b_j(\cdot)$ replaced with a M mixture GMM:

$$b_j(x) = \sum_{k=1}^M c_{jk} b_{jk}(x)$$

$$\text{where } b_{jk}(x) = N(x | \mu_{jk}, \Sigma_{jk})$$

- $k_t=p$ denotes the p^{th} mixture at time t
(similar to our use of $s_t=j$ for j^{th} state at time t).

Independence assumption

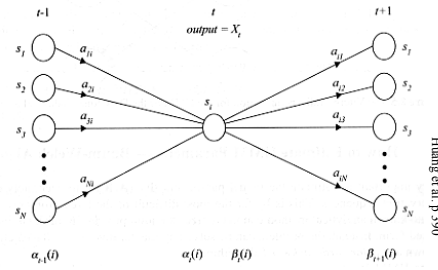
- In practice, Σ_{jk} is usually constrained to be a diagonal matrix (features are assumed independent).
 - Variances only reduces the cost of pdf evaluation
 - Feature sets are chosen to have small covariances.

Expectation

- Let us define

$$\zeta_t(i) = \frac{\Pr(s_t = i | X, \Phi)}{\Pr(X, s_t = i | \Phi)}$$

$$= \frac{\Pr(X | \Phi)}{\Pr(X | \Phi)}$$



Numerator $\zeta_t(i)$

Expectation

- We can extend $\zeta_t(j)$ to pertain to only to the probability of transitioning through state j at time t using mixture k :

$$\zeta_t(j, k) = \frac{\Pr(s_t = j, k_t = k | X, \Phi)}{\Pr(X, s_t = j, k_t = k | \Phi)}$$

$$= \frac{\Pr(X | \Phi)}{\Pr(X | \Phi)}$$

$$= \frac{\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} c_{jk} b_{jk}(x_t) \beta_t(j)}{\sum_{k=1}^N \alpha_T(k)}$$

Maximization

- Estimation for π , A is the same except that $\gamma_t(i, j)$ is computed using the Gaussian mixtures.
- Recall:

$$\hat{\pi}_i = \sum_{k=1}^N \gamma_1(i, k) \quad \text{and} \quad \hat{a}_{ij} = \frac{\sum_{t=2}^T \gamma_t(i, j)}{\sum_{t=2}^T \sum_{k=1}^N \gamma_t(i, k)}$$

Maximization

- Mean reestimation – Weighted average of x_t where the weights are the expectations for transitioning through state j and mixture k :

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^T \zeta_t(j, k) x_t}{\sum_{t=1}^T \zeta_t(j, k)}$$

Maximization

- Variance-Covariance matrix similar:

$$\hat{\Sigma}_{jk} = \frac{\sum_{t=1}^T \zeta_t(j, k)(x_t - \hat{\mu}_{jk})(x_t - \hat{\mu}_{jk})'}{\sum_{t=1}^T \zeta_t(j, k)}$$

Maximization

- weights are ratio of transitions through state j , mixture k to transitions through state j and all mixtures:

$$\hat{c}_{jk} = \frac{\sum_{t=1}^T \zeta_t(j, k)}{\sum_{t=1}^T \sum_{k=1}^M \zeta_t(j, k)}$$

Semicontinuous HMMs

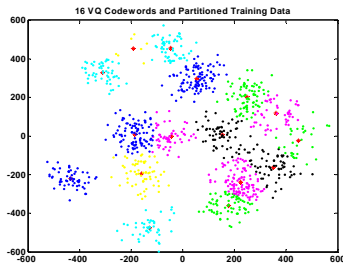
- When using vector quantization for discrete HMMs, the clustering was done independently of the model.
- All HMMs shared the same discrete observation space.

Semicontinuous HMMs

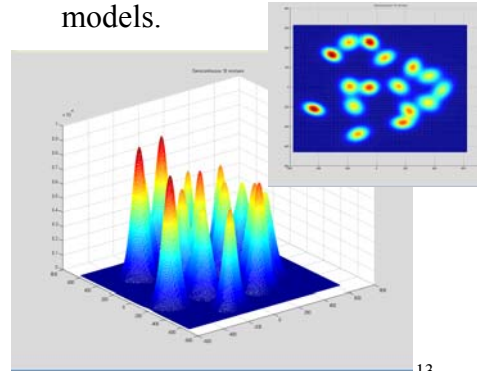
- In a semicontinuous HMM, mixtures are estimated across *all* training data.
 - Construct global GMM
 - GMM independent of class
 - Requires many more mixtures
- Within each model, we estimate the contribution of each mixture to the given state.

Relationship between VQ and Semicontinuous HMMs

VQ Codebook



Mixtures – Common set of densities shared between models.



Semicontinuous HMM

- One can think of the weight as being similar to the discrete state-dependent distribution $b_j(k)$.
- Thus:

$$b_j(x) = \sum_{k=1}^M \underbrace{b_j(k)}_{\text{contribution of k'th mixture}} N(x | \mu_k, \Sigma_k)$$

(essentially a discrete pdf on $[1, 2, \dots, M]$)

Semicontinuous HMMs

- For most mixtures, $b_j(k)$ values very small.
- Approximate $b_j(x)$ by subset $\eta(x)$.
- $\eta(x)$ contains largest $b_j(k)$ values:

$$b_j(x) \cong \sum_{k \in \eta(x)} b_j(k) N(x | \mu_k, \Sigma_k)$$

Semicontinuous HMMs

- Advantages
 - Ability to cache likelihood computations across models
 - Larger data sets to estimate the mixtures
- Also called tied-mixtures as mixtures from different states share Gaussians
 - Common to tie a subset of the space.
e.g. models for /T AY M/ and /T AY T/ have common subset

Initial estimates

- Zero probabilities will always remain zero.
- Discrete HMMs
 - Reasonably insensitive to starting parameters.
 - Uniform distributions for A , π , b are reasonable.
- Continuous/Semicontinuous HMMs
 - A , π as with discrete HMMs
 - Mixture parameters much more sensitive.

Continuous HMM State-dependent distribution B

- Construct a discrete HMM
- Segment the data using the Viterbi algorithm into N states.
- Partition the data from each state using a clustering method such as VQ.
- Estimate the initial means, covariances, and weights from the segmented data.

Semicontinuous HMM (SCHMM)

- Start with a discrete HMM
- Similar to continuous case
 - Initial means are codewords
 - Covariance matrix is estimated from the sample covariance of training words belonging to each codeword.
 - Continuous HMM could also be estimated this way by taking only the top M most frequent codewords in each state as the mixtures.

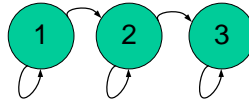
Semicontinuous HMM

- Also possible to use the LBG binary splitting algorithm to grow the mixtures.

Topology

- The state transition matrix defines the model topology.
- Here, we see a left to right model without any skips allowed (i.e. state 1 to state 3):

$$\begin{bmatrix} .3 & .7 & 0 \\ 0 & .2 & .8 \\ 0 & 0 & 1 \end{bmatrix}$$



Topology

- Left to right models are appropriate when the signal being modeled has temporal structure (i.e. a phone, syllable, word, ...)
- Models in which every state can always be reached are ergodic.
 - Temporal structure is not as well defined as in a left to right model.
 - Well suited for silence, text-independent speaker recognition, etc.

Topology & state count

- The optimal number of states is difficult to predict.
- For structured speech using left to right models, about 15-25 states per second is appropriate.
- Unstructured sounds such as silence, can get by with fewer states (i.e. one state may be sufficient for a silence model)

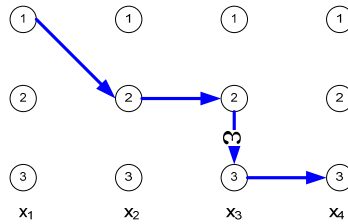
Null transition

- A notational convenience is the null transition.
- In a null transition, the state changes but no observation symbol is emitted/observed.

a_{ij}^{ϵ} - null transition from state i to j

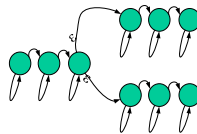
$$\sum_j a_{ij} + a_{ij}^{\epsilon} = 1 \quad \text{for all } i$$

Null transition example



Null transition

- Why bother with null transitions?
 - Permits us to easily construct networks of HMMs



- Null transitions are only a notational convenience. They do not add to the modeling capability.

Null transitions

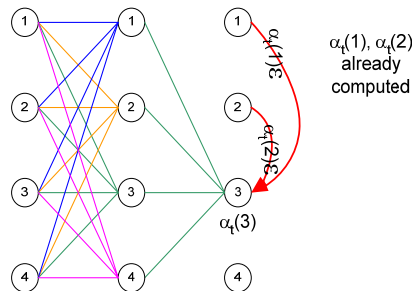
- Requires a minor modification to the forward algorithm

$$\alpha_t(j) = \sum_{i=1}^N \alpha_{t-1}(i) a_{ij} b_j(x_t) + \sum_{i=1}^N \alpha_t(i) a_{ij}^{\varepsilon} \quad \begin{array}{l} 1 \leq t \leq T \\ 1 \leq j \leq N \end{array}$$

- Algorithm is recursive, but easily handled if the null transitions are only left to right:

$$a_{ij}^{\varepsilon}, i \geq j \rightarrow a_{ij}^{\varepsilon} = 0$$

Null transitions



$$\alpha_t(j) = \sum_{i=1}^N \alpha_{t-1}(i) a_{ij} b_j(x_t) + \sum_{i=1}^N \alpha_t(i) a_{ij}^{\varepsilon}$$

Training criteria

- We assume distribution chosen to model the speech really fits the speech and that there is sufficient data to model it.
- Baum-Welch equations use a maximum-likelihood estimator of the distribution parameters.
 - Parameters are chosen to maximize the probability with respect to the training data.

Training criteria

- Our distribution choices are merely simplifications, and we do not always have enough data.
- Other criteria may help us find a better estimator given the problems of approximate distributions & lack of data.

Training criteria

- Other criteria are possible:
 - minimum classification error (MCE)
 - maximum mutual information estimation (MMIE)
 - maximum *a-posteriori* estimation (MAP)

Methods to cope with insufficient data

- Obtain more data.
- Reduce the number of parameters
 - Use less states/mixtures (not always desirable)
 - Use diagonal covariance matrices
 - Tie parameters, i.e. SCHMM
- Interpolate
 - deleted interpolation
 - parameter flooring

Deleted interpolation

- Suppose we want to create a model which is specific to an individual.
- We might find that we have insufficient training data to do so.
- One possibility is to create two models
 - Speaker independent
 - Speaker dependent

Deleted interpolation

- Speaker independent model – Sufficient training data
- Speaker dependent model – Insufficient training data, but hopefully a better fit to the distribution of speaker-specific data.

Deleted interpolation

- Interpolate the two models:

$$\Pr_{DI}(X) = \lambda \Pr(X | \Phi_{SpkrDep}) + (1 - \lambda) \Pr(X | \Phi_{SpkrIndep})$$

- How do we determine the weight λ ?

Deleted interpolation: estimating λ

- In the example of building a speaker specific model, we want λ to be appropriate for speaker-specific data.
- We cannot use the speaker-specific training data to estimate λ , it would match the speaker-specific model too closely.
- Consequently, we need speaker-specific data not used in training.

Deleted interpolation: estimating λ

- We hold back, or delete, some of the training data and build models with the remaining data.
- Using an N-fold cross validation, we can create multiple models.
- We can determine the likelihood of the two models with respect to the deleted (withheld) data.

Deleted interpolation: estimating λ

- Given an initial estimate for λ , we can use the EM algorithm with each of the N cross-fold validation models to determine an appropriate value λ_i for each partition i .
- Set λ as the average of the λ_i 's.

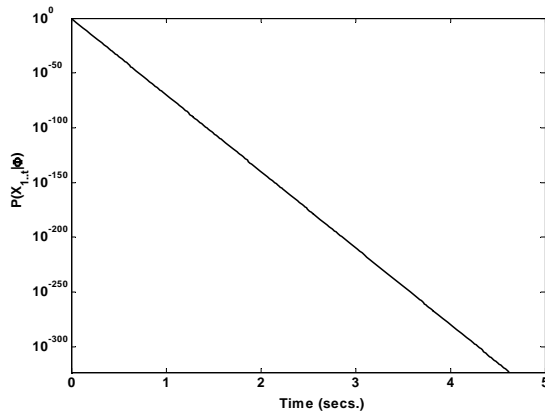
Parameter flooring

- Never let a parameter value fall beneath some threshold.
- Can be seen as a special case of interpolating with a uniform distribution.
- This is especially useful for covariance matrices
 - Typical approach: Find the covariance matrix Σ of all the data and do not let any covariance matrix entry fall beneath some percentage of Σ .

Probability representations: Scaling

- Suppose that we have a five second phrase which we would like to recognize.
- Assuming that feature extraction is at 100 frames per second (10 MS frame advance), this consists of 500 frames.
- Suppose that for a given HMM network, the maximum probability for any observation and state transition is .2.

The need for scaling:



Probability tends towards zero as the number of frames increase.

Scaling

- By introducing scaling coefficients, we can keep the probabilities within machine precision.

$$\bullet \quad S_t = \frac{1}{\sum_{j=1}^N \alpha'_t(j)} \rightarrow \sum_{j=1}^N S_t \alpha'_t(j) = 1$$

Scaling the forward & backward alg.

- We can multiply each α_t and β_t by the scale factor S_t and use the scaled α_{t-1} and β_{t+1} :

$$\hat{\alpha}_t(j) = \sum_{i=1}^N \alpha'_{t-1}(i) a_{ij} b_j(x_t)$$

$$S_t = \frac{1}{\sum_{i=1}^N \hat{\alpha}_t(i)}$$

$$\alpha'_t(j) = S_t \hat{\alpha}_t(j)$$

- Prevents hardware underflow.

Scaling

- At each step, we scale.

$$\alpha'_1(j) = S_1 \alpha_1(j) \quad \text{by definition}$$

$$\alpha'_2(j) = S_2 \sum_{i=1}^N \alpha'_1(i) a_{ij} b_j(x_2)$$

$$\alpha'_2(j) = S_2 \sum_{i=1}^N S_1 \alpha_1(i) a_{ij} b_j(x_2)$$

$$\alpha'_2(i) = S_1 S_2 \sum_{i=1}^N \alpha_1(i) a_{ij} b_j(x_2) \quad S_2 \text{ constant across } i$$

- Leads to: $\alpha'_t(j) = S_1 S_2 \dots S_t \sum_{i=1}^N \alpha_{t-1}(i) a_{ij} b_j(x_t)$
 $= \left(\prod_{\tau=1}^t S_\tau \right) \alpha_t(j)$
 $= \text{Scale}_\alpha(T) \alpha_t(j)$ where $\text{Scale}_\alpha(T) = \left(\prod_{\tau=1}^t S_\tau \right)$

Recovering probability from scaling

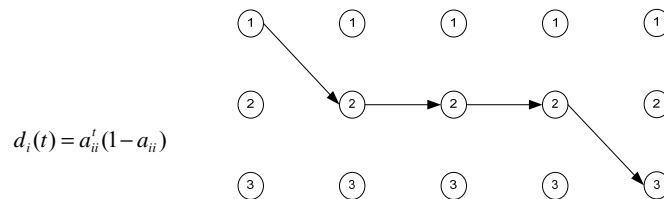
$$\begin{aligned}\sum_{i=1}^N \alpha'_T(i) &= \sum_{i=1}^N \text{Scale}_\alpha(T) \alpha_T(i) && \text{from previous slide} \\ &= \text{Scale}_\alpha(T) \sum_{i=1}^N \alpha_T(i) && \text{Scale}_\alpha(T) \text{ independent of } i \\ &= \text{Scale}_\alpha(T) \Pr(X | \Phi) && \Pr(X | \Phi) = \sum_{i=1}^N \alpha_T(i)\end{aligned}$$
$$\Pr(X | \Phi) = \frac{\sum_{i=1}^N \alpha'_T(i)}{\text{Scale}_\alpha(T)} = \frac{1}{\text{Scale}_\alpha(T)} \text{ would still result in underflow}$$
$$\log \Pr(X | \Phi) = -\sum_{t=1}^T \log S_t$$

Probability representations

- As previously discussed, the Viterbi algorithm does not use addition, and we can rewrite the Viterbi equation in the log domain. Hence scaling is not needed.
- However, for the Baum-Welch reestimation equations, scaling is preferable to trying to implement addition in the log domain.
- Finally, due to the small numbers, it is common to use double precision floating point for probability computations.

Duration and HMMs

- When a model stays in a certain state for a period of time, the probability decays exponentially:



Duration

- Many researchers have argued that the exponential distribution is not appropriate to model duration and have proposed semi-Markov models where duration is modeled explicitly.

Higher-order assumptions

- Another criticism of HMMs is that the first-order Markov assumption is inadequate to capture the state transition probabilities.
- One possibility is to permit higher-order assumptions, i.e.:

$$\Pr(X_i | X_1, \dots, X_{i-1}) = \Pr(X_i | X_{i-1}, X_{i-2})$$

- This model is not without drawbacks:
 - Additional parameters to estimate
 - Additional algorithmic complexity

Relaxing the conditional independence assumption.

- We assume that X_t and X_{t-1} are uncorrelated.
- In practice this is usually not true.
- It is possible to include the dependence upon the previous frame(s):

$$\Pr(X | S, \Phi) = \prod_{t=1}^T \Pr(X_t | X_{t-1}, s_t, \Phi)$$

- As with higher-order assumptions, this complicates the model.

Summary of HMM extensions addressing criticisms of HMMs

- All three techniques
 - duration modeling
 - relaxation of the first-order assumption
 - and relaxation of the conditional-independence assumptioncan be shown to provide improvements.
- In practice, they are rarely used as the improvements are not significant with respect to the additional cost.