EE E6820: Speech & Audio Processing & Recognition

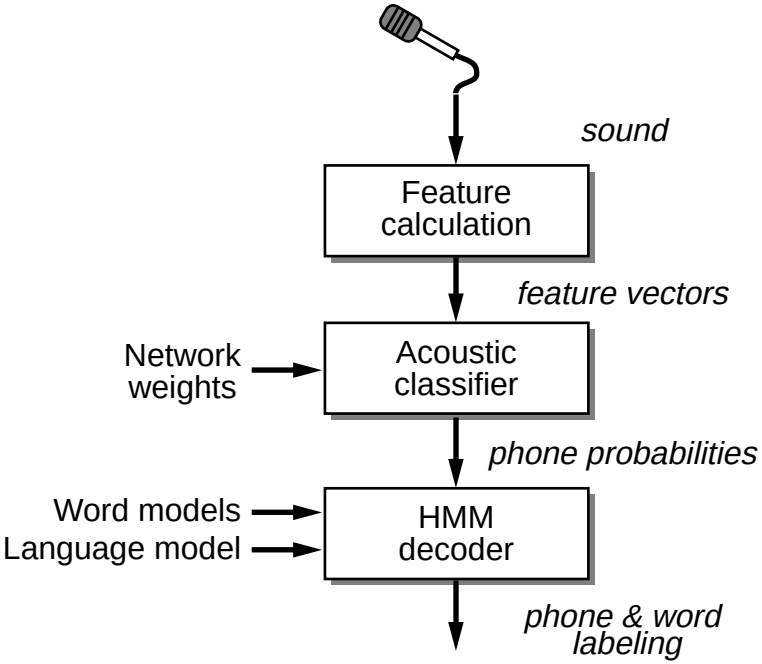
Lecture 10:
ASR: Sequence Recognition

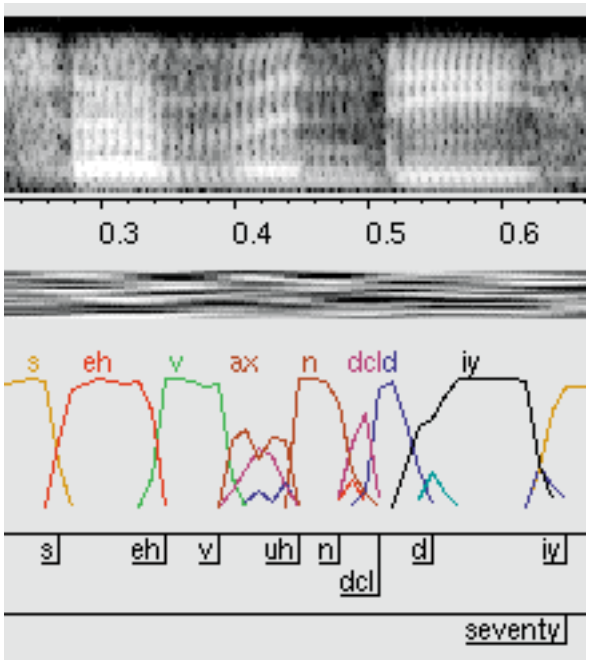
1. **Deterministic sequence recognition: DTW**
2. **Statistical sequence recognition: HMM**
3. **HMM training**
4. **Discriminant models**

Dan Ellis <dpwe@ee.columbia.edu>
http://www.ee.columbia.edu/~dpwe/e6820/

1 Sequence recognition

- **After feature calculation, the problem is to classify the sequences of frames:**

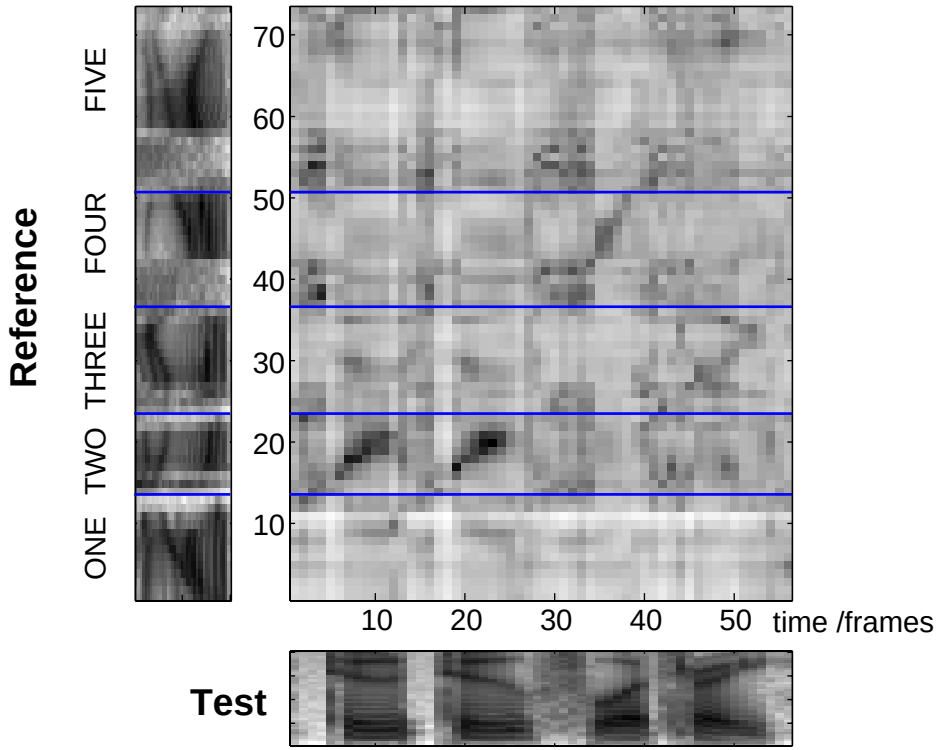




- **Questions**
  - how to represent 'model' sequences
  - how to score matches

Acoustic template matching

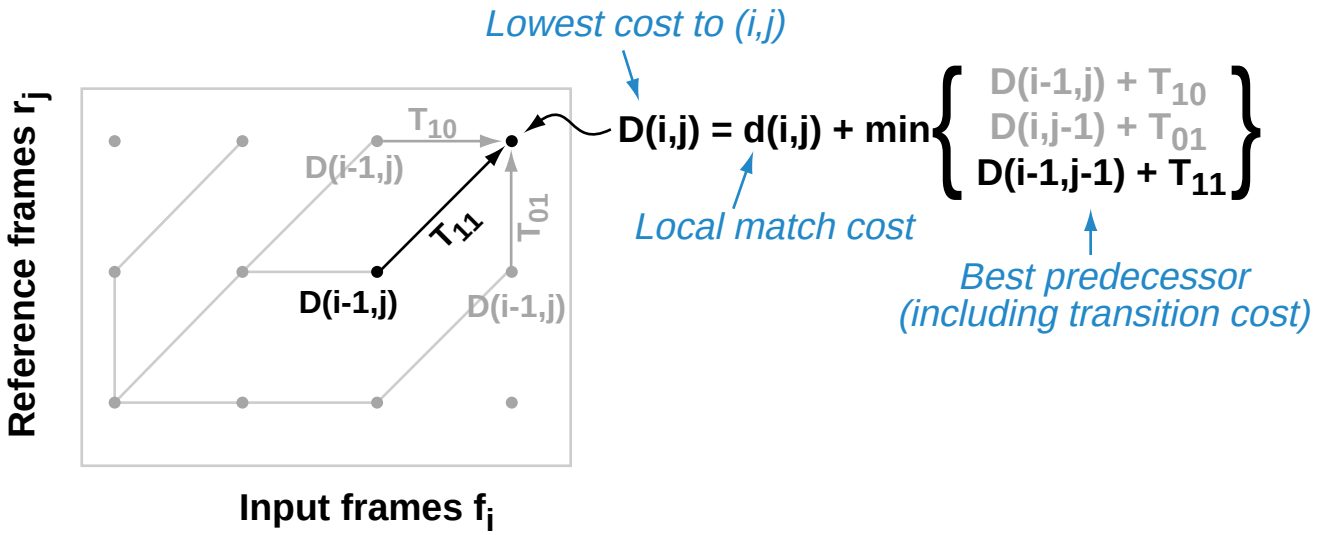
- **Framewise comparison of unknown word and stored templates:**



- distance metric?
- comparison between templates?
- constraints?

Dynamic Time Warp (DTW)

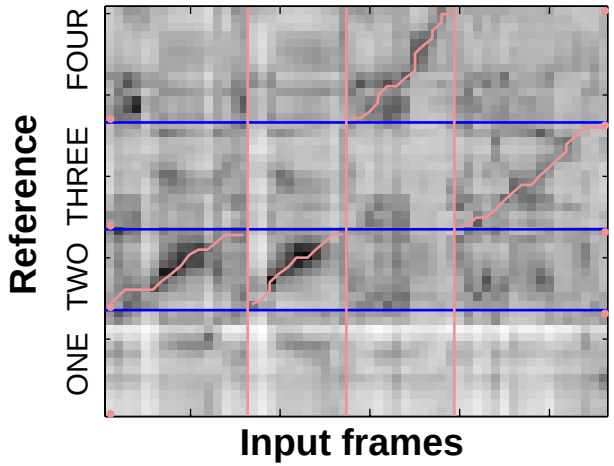
- **Find lowest-cost constrained path:**
  - matrix d(i,j) of distances between input frame f_i and reference frame r_j
  - allowable predecessors & transition costs T_{xy}



- **Best path via traceback from final state**
  - have to store predecessors for (almost) every (i,j)

DTW-based recognition

- **Reference templates for each possible word**
- **Isolated word:**
  - mark endpoints of input word
  - calculate scores through each template (+prune)
  - choose best
- **Continuous speech**
  - one matrix of template slices;
    special-case constraints at word ends



DTW-based recognition (2)

+ **Successfully handles timing variation**
+ **Able to recognize speech at reasonable cost**
- **Distance metric?**
  - pseudo-Euclidean space?
- **Warp penalties?**
- **How to choose templates?**
  - several templates per word?
  - choose ‘most representative’?
  - align and average?

→ **need a *rigorous* foundation...**

Outline

1. Dynamic Time Warp
2. **Hidden Markov Models**
   - Statistical formulation
   - Markov Models
   - Hidden states
3. HMM training
4. Discriminant models

2 Statistical sequence recognition

- **DTW limited because it's hard to optimize**
  - interpretation of distance, transition costs?
- **Need a theoretical foundation: Probability**
- **Formulate as MAP choice among models:**

$$M^* = \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta)$$

  - X = observed features
  - M_j = word-sequence models
  - Θ = all current parameters

Statistical formulation (2)

- **Can rearrange via Bayes' rule (& drop $p(X)$):**

$$M^* = \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta)$$

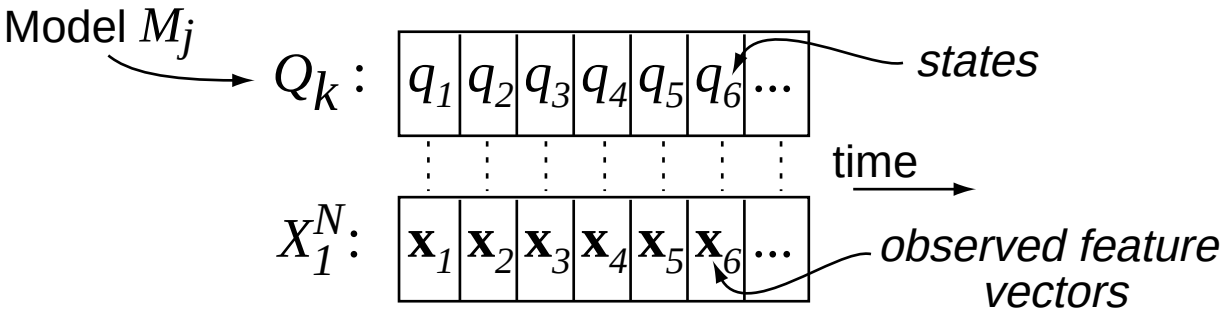
$$= \underset{M_j}{\operatorname{argmax}}\ p(X | M_j, \Theta_A)\, p(M_j | \Theta_L)$$

  - $p(X \mid M_j)$ = likelihood of obs'v'ns under model
  - $p(M_j)$ = prior probability of model
  - Θ_A = acoustics-related model parameters
  - Θ_L = language-related model parameters

- **Questions:**
  - what form of model to use for $p(X | M_j, \Theta_A)$?
  - how to find Θ_A (training)?
  - how to solve for M_j (decoding)?

State-based modeling

- **Assume discrete-state model for the speech:**
  - observations are divided up into time frames
  - model → states → observations:



- **Probability of observations given model is:**

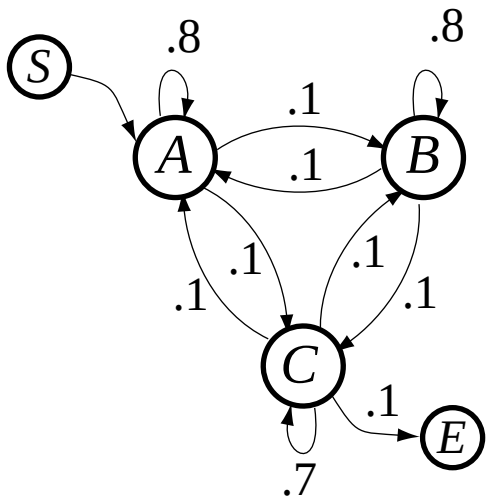
$$p(X|M_j) = \sum_{\text{all } Q_k} p(X_1^N | Q_k, M_j) \cdot p(Q_k | M_j)$$

  - sum over all possible state sequences Q_k

- **How do observations depend on states?**
  How do state sequences depend on model?

Generative Markov models

- **A (first order) Markov model is a finite-state system whose behavior depends only on the current state, e.g.**



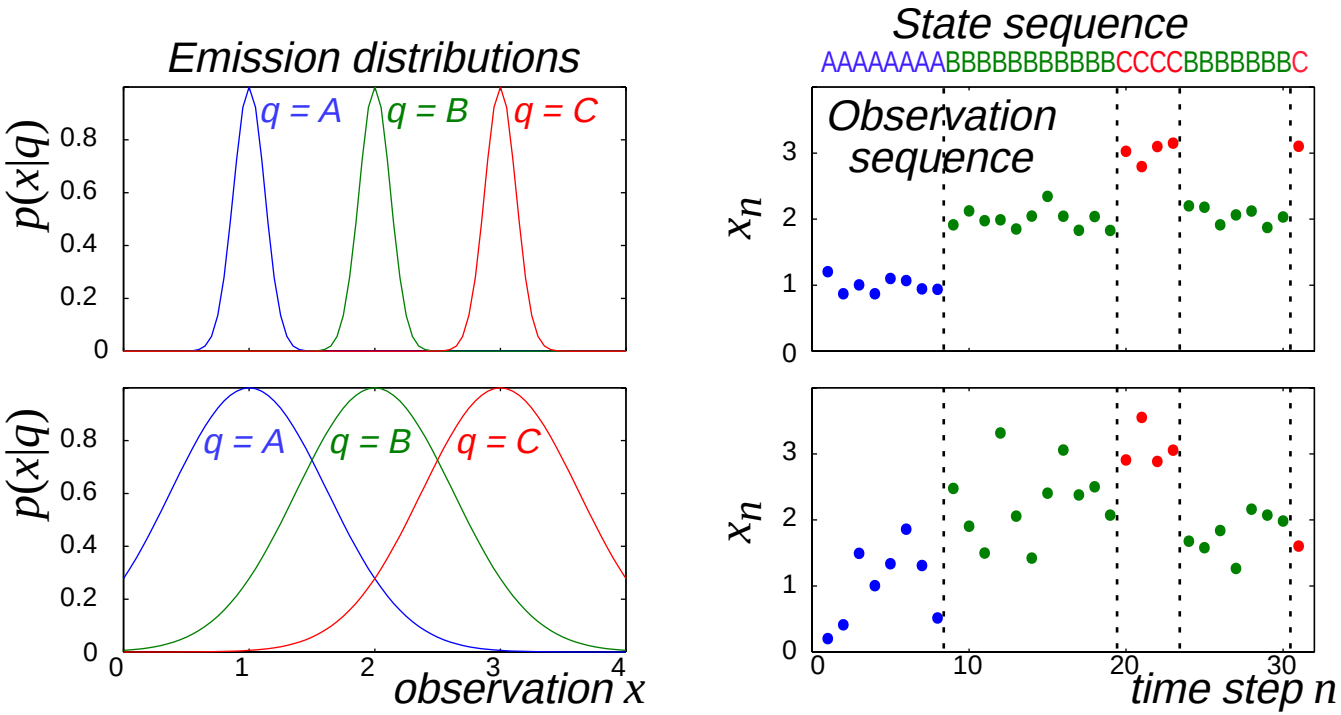
$p(q_{n+1}\|q_n)$		q_{n+1} S	A	B	C	E
q_n	S	0	1	0	0	0
	A	0	.8	.1	.1	0
	B	0	.1	.8	.1	0
	C	0	.1	.1	.7	.1
	E	0	0	0	0	1

S A A A A A A A A B B B B B B B B B C C C C B B B B B B C E

- **Defined by:**
  - state set $\{q^i\}$ (including start & end states)
  - transition matrix $p(q_{n+1} \mid q_n)$ depends *only* on current state

Hidden Markov models

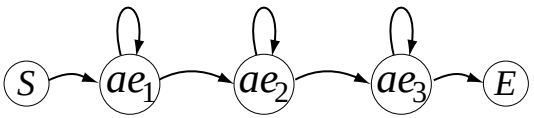
- **Markov models where state sequence $Q = \{q_n\}$ is not directly observable (= 'hidden')**
- **But, observations X *do* depend on Q:**
  - x_n is rv that depends on current state: $p(x|q)$

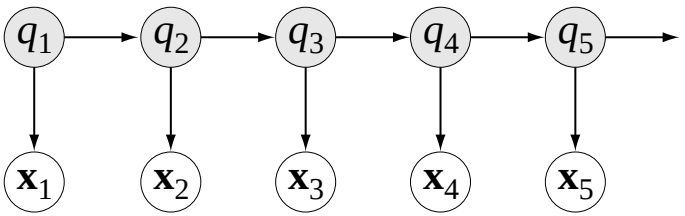


- can still tell *something* about state seq...

Hidden Markov models (2)

- **An HMM is specified by:**
  - emission distributions $p(x|q^i) \equiv b_i(x)$
  - transition probabilities $p(q_n^j | q_{n-1}^i) \equiv a_{ij}$
  - (initial state probabilities $p(q_1^i) \equiv \pi_i$)
- **Speech models** M_j
  - typ. left-to-right HMMs (sequence constraint)
  - observation & evolution are conditionally independent of rest given (hidden) state q_n





  - *self-loops* for time dilation

Markov models for sequence recognition

- **Independence of observations:**
  - observation x_n depends only current state q_n

$$p(X|Q) = p(x_1, x_2, \ldots x_N | q_1, q_2, \ldots q_N)$$
$$= p(x_1|q_1) \cdot p(x_2|q_2) \cdot \ldots p(x_N|q_N)$$
$$= \prod_{n=1}^{N} p(x_n|q_n) = \prod_{n=1}^{N} b_{q_n}(x_n)$$

- **Markov transitions:**
  - transition to next state q_{i+1} depends only on q_i

$$p(Q|M) = p(q_1, q_2, \ldots q_N | M)$$
$$= p(q_N|q_1 \cdots q_{N-1}) p(q_{N-1}|q_1 \cdots q_{N-2}) \ldots p(q_2|q_1) p(q_1)$$
$$= p(q_N|q_{N-1}) p(q_{N-1}|q_{N-2}) \ldots p(q_2|q_1) p(q_1)$$
$$= p(q_1) \prod_{n=2}^{N} p(q_n|q_{n-1}) = \pi_{q_1} \prod_{n=2}^{N} a_{q_{n-1} q_n}$$

Model fit calculation

- **From 'state-based modeling':**

$$p(X|M_j) = \sum_{\text{all } Q_k} p(X_1^N | Q_k, M_j) \cdot p(Q_k | M_j)$$

- **For HMMs:**

$$p(X|Q) = \prod_{n=1}^{N} b_{q_n}(x_n)$$

$$p(Q|M) = \pi_{q_1} \cdot \prod_{n=2}^{N} a_{q_{n-1} q_n}$$

- **Hence, solve for M^* :**
  - calculate $p(X|M_j)$ for each available model, scale by prior $p(M_j) \rightarrow p(M_j|X)$

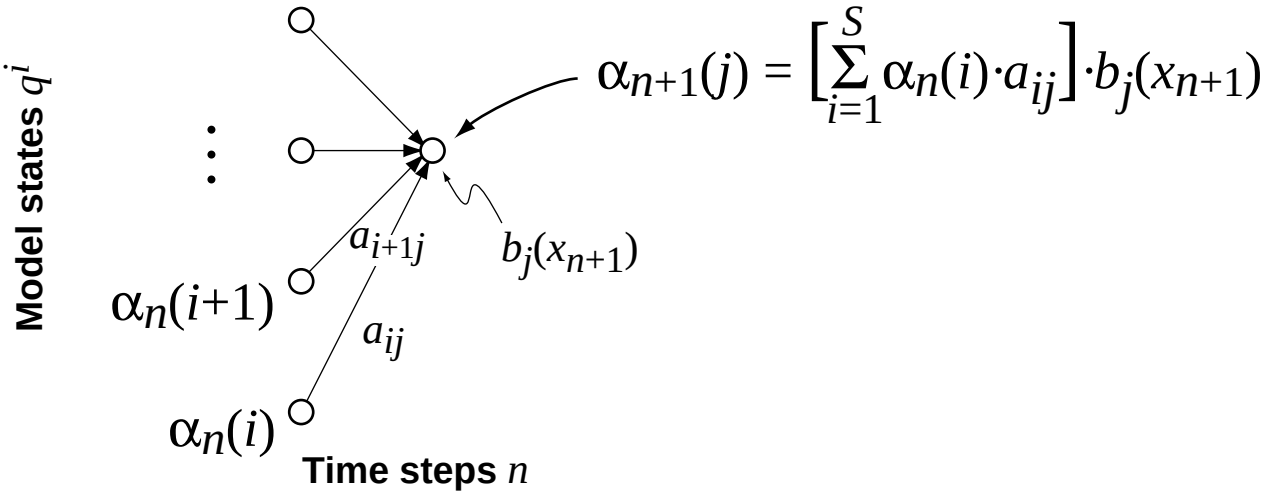
- **Sum over *all* Q_k ???**

The 'forward recursion'

- **Dynamic-programming-like technique to calculate sum over all Q_k**
- **Define $\alpha_n(i)$ as the probability of *getting to* state q^i at time step n (by any path):**

$$\alpha_n(i) = p(x_1, x_2, \ldots x_n, q_n = q^i) \equiv p(X_1^n, q_n^i)$$

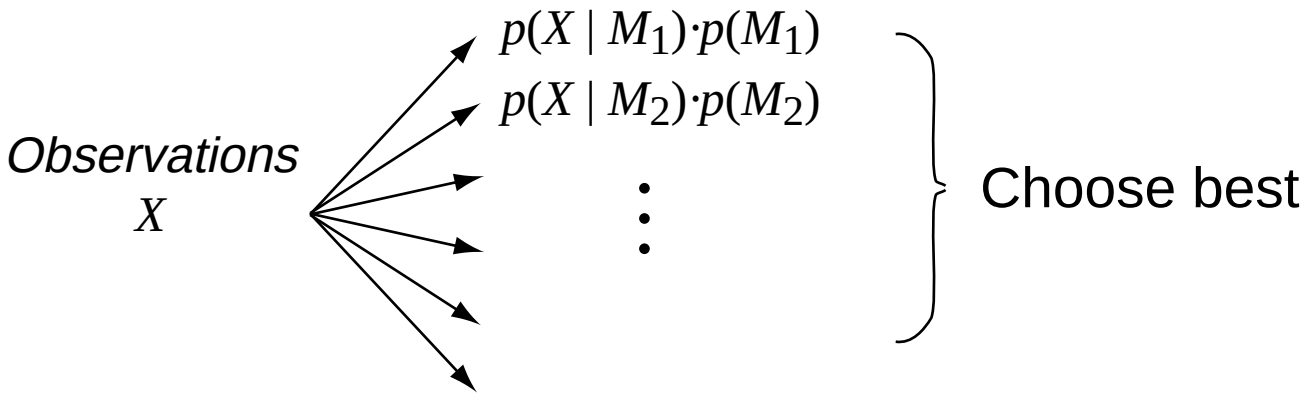
- **Then $\alpha_{n+1}(j)$ can be calculated recursively:**



Forward recursion (2)

- **Initialize** $\alpha_1(i) = \pi_i \cdot b_i(x_1)$
- **Then total probability** $p(X_1^N | M) = \sum_{i=1}^{S} \alpha_N(i)$

$\rightarrow$ **Practical way to solve for** $p(X \mid M_j)$ **and hence perform recognition**



Optimal path

- **May be interested in actual q_n assignments**
  - which state was 'active' at each time frame
  - e.g. phone labelling (for training?)
- **Total probability is over *all* paths...**
- **... but can also solve for single *best* path = "Viterbi" state sequence**
- **Probability along best path to state q_{n+1}^j:**

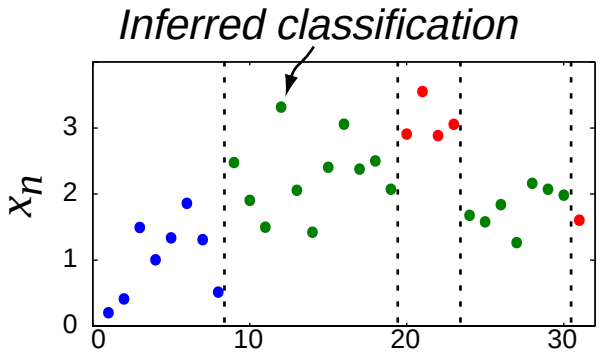
$$\alpha_{n+1}^*(j) = \left[\max_i \{\alpha_n^*(i) a_{ij}\}\right] \cdot b_j(x_{n+1})$$

  - backtrack from final state to get best path
  - final probability is product only (no sum)
    $\rightarrow$ log-domain calculation just summation
- **Total probability often dominated by best path:**

$$p(X, Q^* | M) \approx p(X|M)$$

Interpreting the Viterbi path

- **Viterbi path assigns each x_n to a state q^i**
  - performing classification based on $b_i(x)$
  - ... at the same time as applying transition constraints a_{ij}



- **Can be used for segmentation**
  - train an HMM with 'garbage' and 'target' states
  - decode on new data to find 'targets', boundaries
- **Can use for (heuristic) training**
  - e.g. train classifiers based on labels...

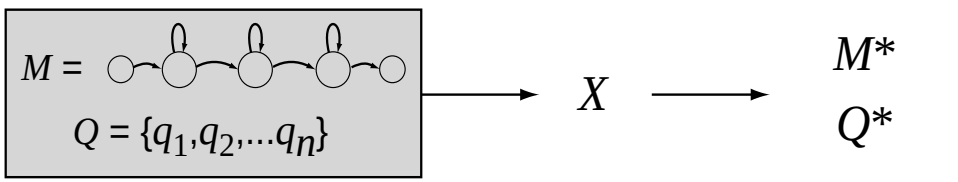
HMM summary

- **HMM M_j is hypothesized *generator* of obs'v'n X**
- **During generation, behavior of model depends only on current state q_n:**
  - transition probabilities $p(q_{n+1} \mid q_n) = a_{ij}$
  - emission distributions $p(x_n \mid q_n) = b_i(x)$
- **Given just observed emissions X we can still calculate probability that they came from M_j:**

$$p(X|M) = \sum_{\text{all } Q} p(X|Q, M)\, p(Q|M)$$

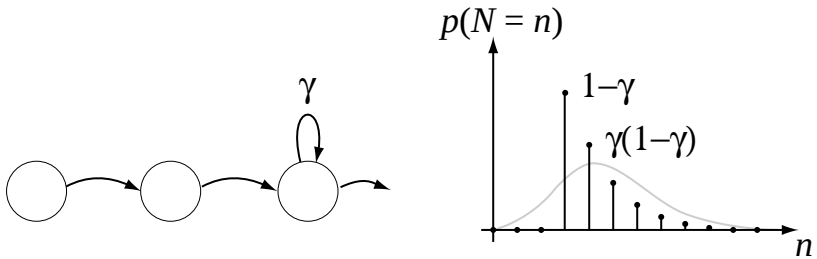
- **Can also infer most likely state sequence**

$$Q^* = \underset{Q}{\text{argmax}}\ p(X, Q|M)$$

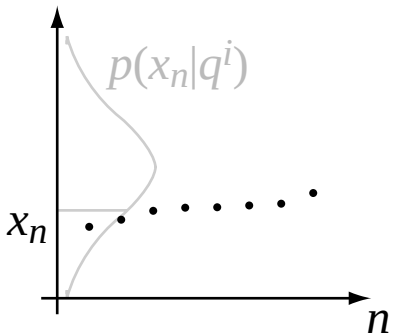


Validity of HMM assumptions

- **Key assumption is *conditional independence*: Given q^i, future evolution & obs. distribution are independent of previous events**
  - duration behavior: self-loops imply exponential distribution



  - independence of successive x_ns



$$p(X) = \prod p(x_n|q^i) \; ?$$

Outline

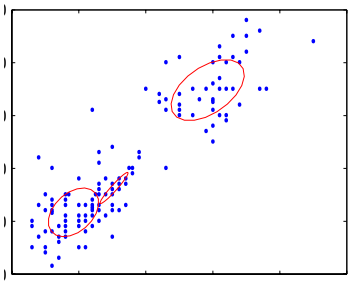
1. Dynamic Time Warp
2. Hidden Markov Models
3. **HMM training**
   - EM for HMM parameters
   - State occupancy probabilities
   - Acoustic parameters
4. Discriminative models

3 Training models

- **We know how to *use* an HMM, but where does it come from?**
- **In fact, HMM's assumption of stationarity within states seems a worse model than DTW templates**
- **Objection to DTW was the lack of a principled optimization procedure**
- **Probabilistic foundation of HMMs supports optimization:**
  - maximum-likelihood - adjust Θ to maximize likelihood of training data under model
  - (really want minimum error rate or ...)

EM for HMMs

- **Expectation-Maximization (EM) was introduced for Gaussian mixture models (GMMs)**
  - fit GMMs to arbitrary data
  - EM finds locally-optimal model parameters Θ that maximize data likelihood $p(x_{train}|\Theta)$
  - makes (some) sense for decision rules like $p(x|M_j) \cdot p(M_j)$



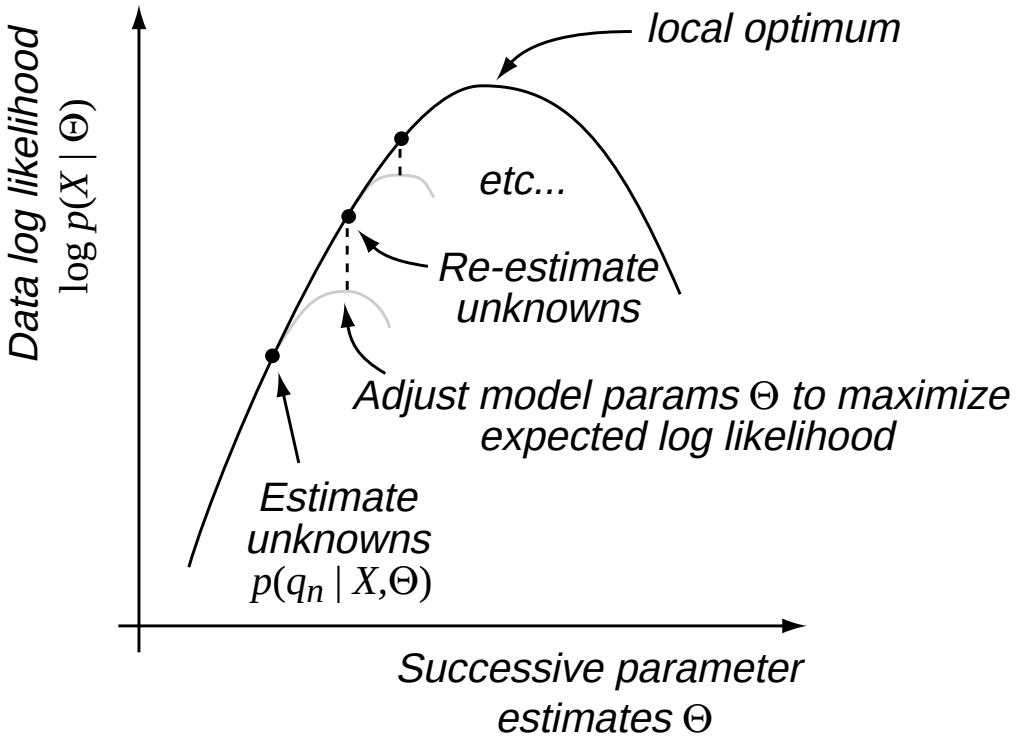
- **Principle: adjust Θ to maximize expected log likelihood of known x & unknown u:**

$$E[\log p(x, u|\Theta)] = \sum_u p(u|x, \Theta_{old}) \log[p(x|u, \Theta) p(u|\Theta)]$$

  - for GMMs, unknowns = mix assignments k
  - for HMMs, unknowns = hidden state q_n (take Θ to include M_j)

What EM does

- **Maximize data likelihood by repeatedly estimating unknowns and re-maximizing expected log likelihood:**



EM for HMMs (2)

- **Expected log likelihood for HMM:**

$$\sum_{\text{all } Q_k} p(Q_k | X, \Theta_{\text{old}}) \log [p(X | Q_k, \Theta) p(Q_k | \Theta)]$$

$$= \sum_{\text{all } Q_k} p(Q_k | X, \Theta_{\text{old}}) \log \left[\prod_n p(x_n | q_n) \cdot p(q_n | q_{n-1}) \right]$$

$$= \sum_{n=1}^{N} \sum_{i=1}^{S} p(q_n^i | X, \Theta_{\text{old}}) \log p(x_n | q_n^i, \Theta)$$

$$+ \sum_{i=1}^{S} p(q_1^i | X, \Theta_{\text{old}}) \log p(q_1^i | \Theta)$$

$$+ \sum_{n=2}^{N} \sum_{i=1}^{S} \sum_{j=1}^{S} p(q_{n-1}^i, q_n^j | X, \Theta_{\text{old}}) \log p(q_n^j | q_{n-1}^i, \Theta)$$

- closed-form maximization by differentiation etc.

EM update equations

- **For acoustic model (e.g. 1-D Gauss):**

$$\mu_i = \frac{\sum_n p(q_n^i \mid X, \Theta_{\text{old}}) \cdot x_n}{\sum_n p(q_n^i \mid X, \Theta_{\text{old}})}$$

- **For transition probabilities:**

$$p(q_n^j \mid q_{n-1}^i) = \frac{\sum_n p(q_{n-1}^i, q_n^j \mid X, \Theta_{\text{old}})}{\sum_n p(q_{n-1}^i \mid X, \Theta_{\text{old}})}$$

- **'Normalized expectations' as for GMMs**
- **Require 'state occupancy probabilities',**

$$p(q_n^i \mid X_1^N, \Theta_{\text{old}})$$

The forward-backward algorithm

- **We need $p(q_n^i \mid X_1^N)$ for EM updates (Θ implied)**
- **Forward algorithm gave us $\alpha_n(i) = p(q_n^i, X_1^n)$**
  - excludes influence of remaining data X_{n+1}^N
- **Hence, define $\beta_n(i) = p(X_{n+1}^N \mid q_n^i, X_1^n)$**

  so that $\alpha_n(i) \cdot \beta_n(i) = p(q_n^i, X_1^N)$

  then $p(q_n^i \mid X_1^N) = \dfrac{\alpha_n(i) \cdot \beta_n(i)}{\sum_j \alpha_n(j) \cdot \beta_n(j)}$
- **Recursive definition for β:**

  $\beta_n(i) = \sum_j \beta_{n+1}(j) a_{ij} b_j(x_{n+1})$
  - recurses *backwards* from final state N

Estimating a_{ij} from α & β

- **From EM equations:**

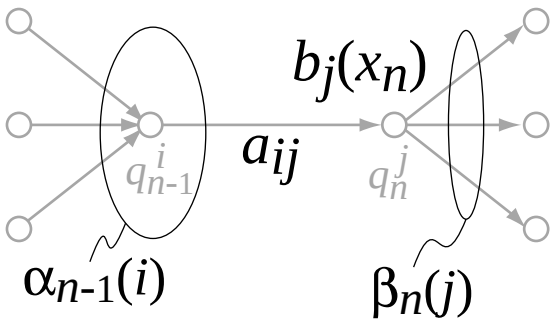
$$p(q_n^j \,|\, q_{n-1}^i) \;=\; a_{ij}^{\text{new}} \;=\; \frac{\sum_n p(q_{n-1}^i, q_n^j \,|\, X, \Theta_{\text{old}})}{\sum_n p(q_{n-1}^i \,|\, X, \Theta_{\text{old}})}$$

  - prob. of transition normalized by prob. in start

- **Obtain from** $p(q_{n-1}^i, q_n^j, X \,|\, \Theta_{\text{old}})$

$$= p(X_{n+1}^N \,|\, q_n^j)\, p(x_n \,|\, q_n^j)\, p(q_n^j \,|\, q_{n-1}^i)\, p(q_{n-1}^i, X_1^{n-1})$$

$$= \beta_n(j) \cdot b_j(x_n) \cdot a_{ij} \cdot \alpha_{n-1}(i)$$



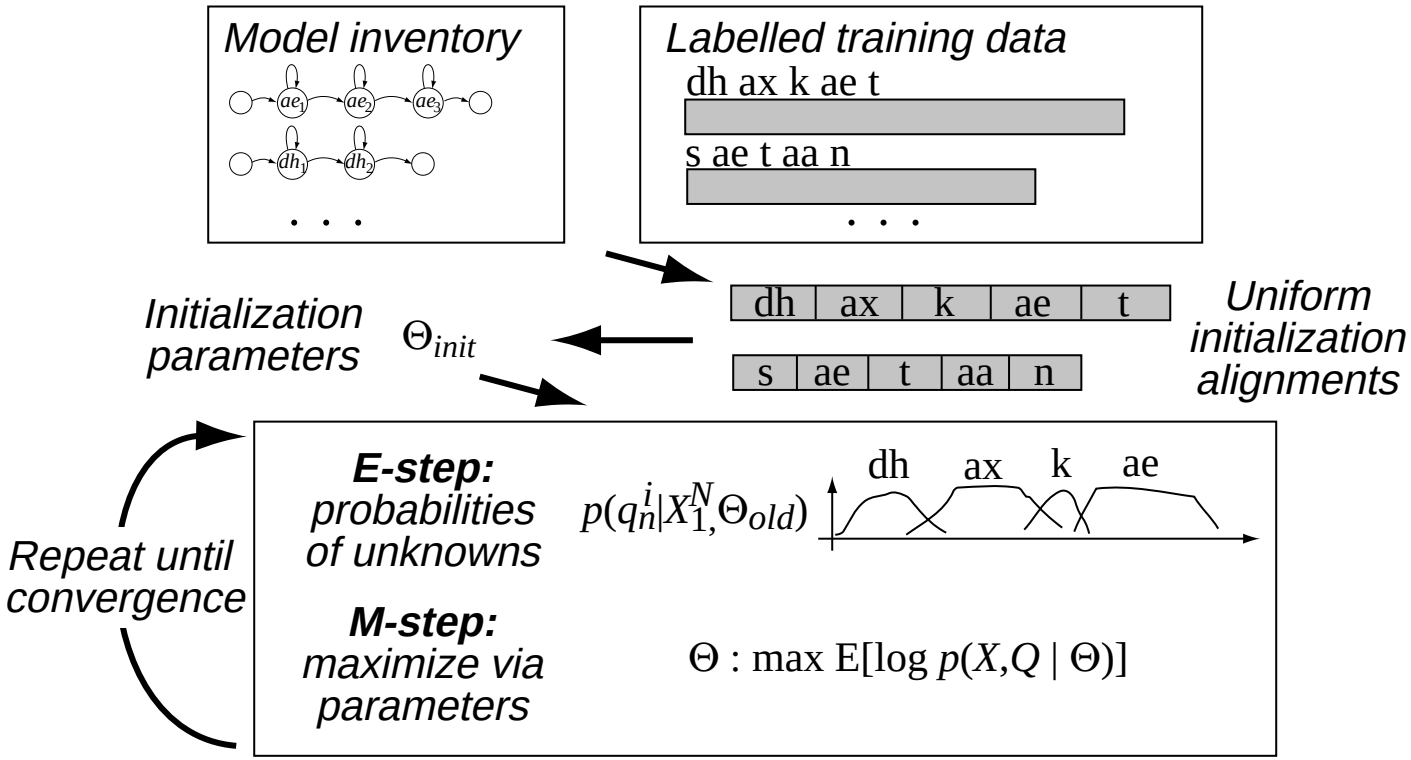
GMM-HMMs in practice

- **GMMs as acoustic models:**
  train by including mixture indices as unknowns
  - just more complicated equations...
- **Practical GMMs:**
  - 9 to 39 feature dimensions
  - 2 to 64 Gaussians per mixture
    depending on number of training examples
- **Training constraint: define state *sequence***
  - decide words & pronunciations for training data
  - EM will then figure out boundaries
- **Lots of data $\rightarrow$ can model more classes**
  - e.g context-independent (CI): q^i = ***ae*** ***aa*** ***ax*** ...

  $\rightarrow$context-dependent (CD): q^i = *b*-***ae***-*b* *b*-***ae***-*k* ...

HMM training in practice

- **EM only finds local optimum**
  → critically dependent on initialization
  - approximate parameters / rough alignment



- **Applicable for more than just words...**

Outline

1. Dynamic Time Warp
2. Hidden Markov Models
2. HMM training
4. **Discriminant models**
   - Being discriminant
   - Neural-net recognizers

3 Discriminant models

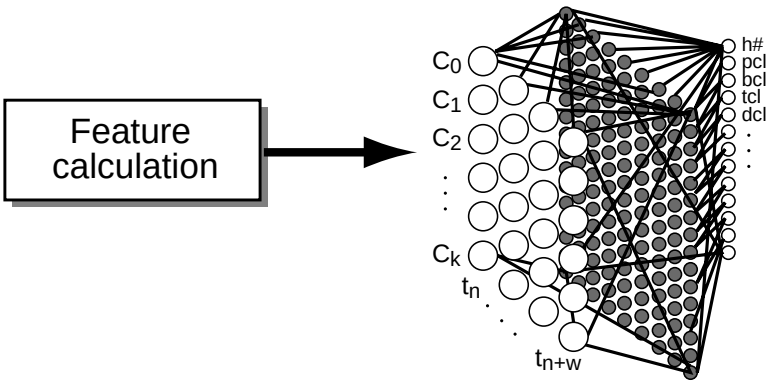
- **EM training of HMMs is *maximum likelihood***
  - i.e. local $\max p(X_{trn} \mid \Theta)$
  - converges to Bayes optimum ... in the limit
- **Decision rule is** $\max p(X \mid M) \cdot p(M)$
  - training will increase $p(X \mid M_{correct})$
  - may also increase $p(X \mid M_{wrong})$ (more!)
- **Discriminant training tries directly to increase discrimination between right & wrong models**
  - e.g. Maximum Mutual Information (MMI)

$$I(M_j, X|\Theta) = \log \frac{p(M_j, X|\Theta)}{p(M_j|\Theta)p(X|\Theta)}$$

$$= \log \frac{p(X|M_j, \Theta)}{\sum p(X|M_k, \Theta)p(M_k|\Theta)}$$

Neural Network Acoustic Models

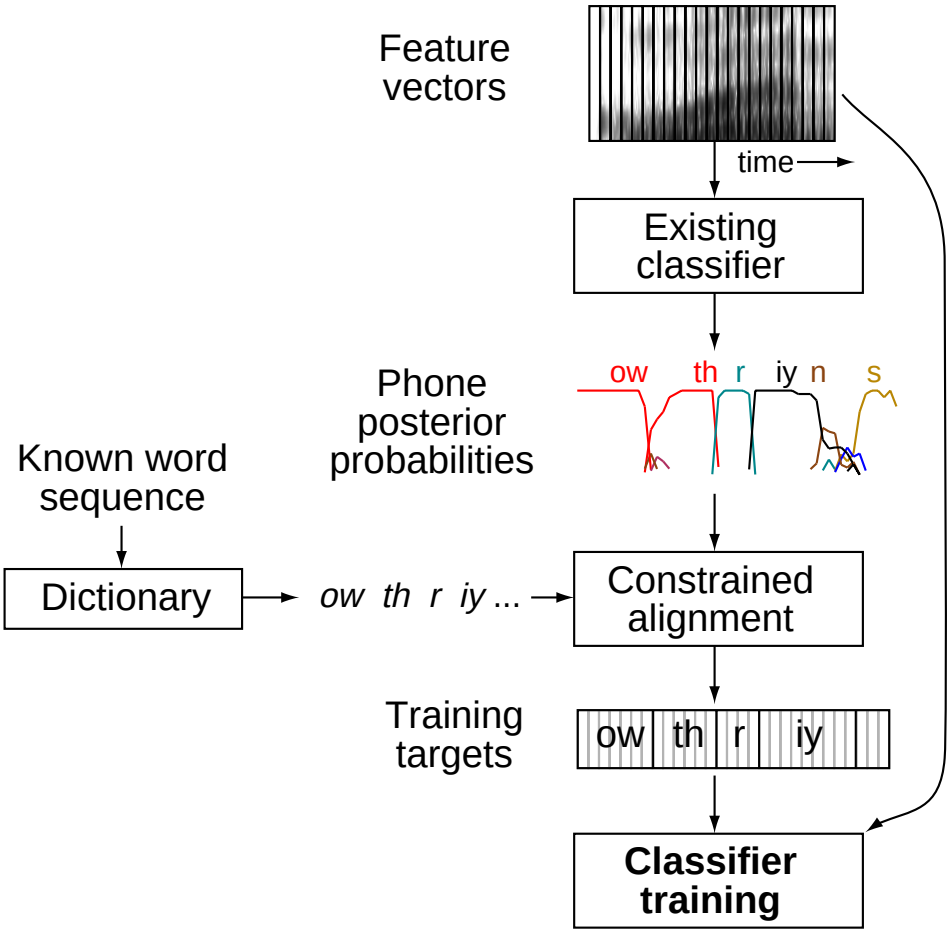
- **Single model generates posteriors directly for all classes at once = *frame*-discriminant**
- **Use regular HMM *decoder* for recognition**
  - set $b_i(x_n) = p(x_n | q^i) \approx p(q^i | x_n) / p(q^i)$
- **Nets are less sensitive to input representation**
  - skewed feature distributions
  - correlated features
- **Can use temporal context window to let net 'see' feature dynamics:**



posteriors $p(q^i \mid X)$

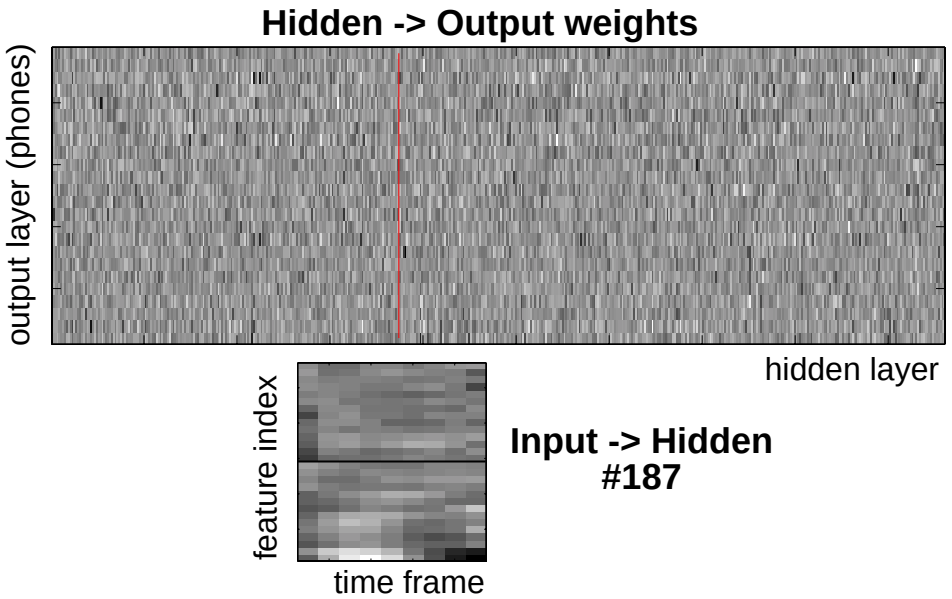
Forced alignment

- **Best (Viterbi) labeling given existing classifier constrained by known word sequence**

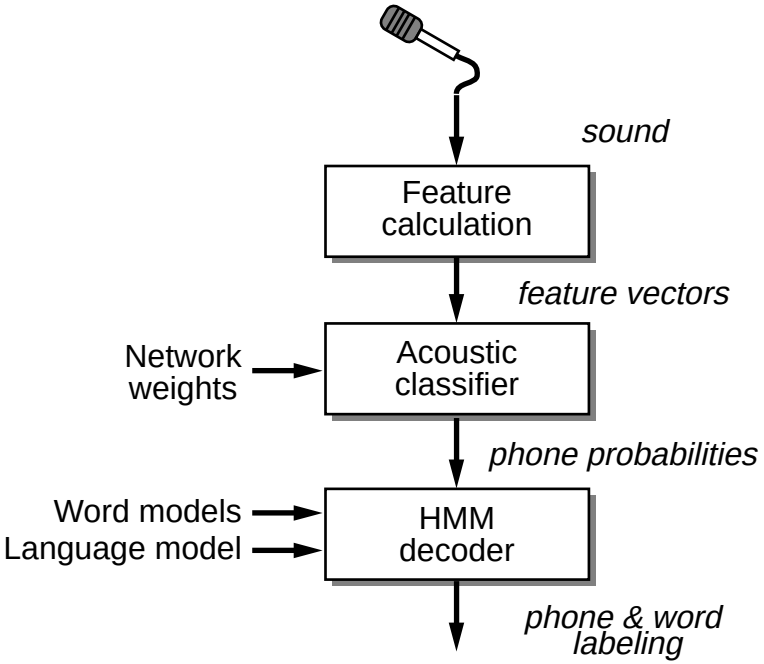


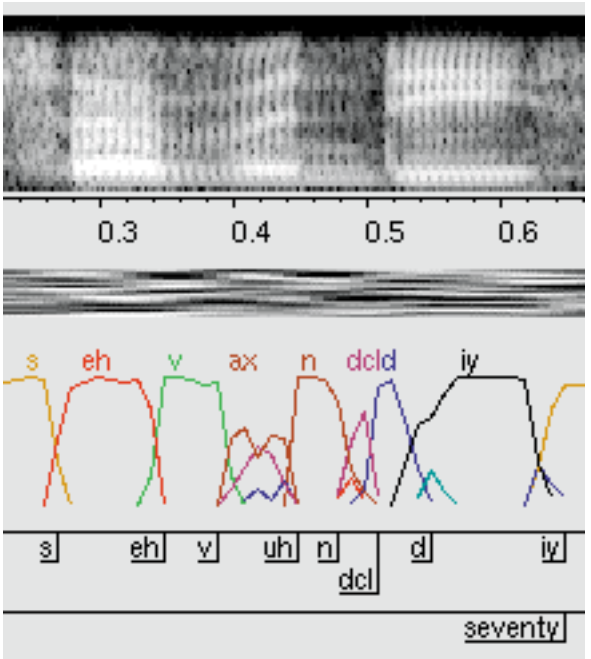
Neural nets: Practicalities

- **Typical net sizes:**
  - input layer: 9 frames x 9-40 features ~ 300 units
  - hidden layer: 100-8000 units, dep. training data
  - output layer: 30-60 context-independent phones
- **Hard to make context dependent**
  - problems training many classes that are similar?
- **Representation is opaque:**



Recap: Recognizer Structure





- **Know how to execute each state**
- **Know how to train each stage**
- **.. except language/word models**

Summary

- **Speech is modeled as a *sequence* of features**
  - need temporal aspect to recognition
  - best time-alignment of templates = DTW
- **Hidden Markov models are rigorous solution**
  - self-loops allow temporal dilation
  - exact, efficient likelihood calculations
- **HMMs can be trained via EM**
  - guaranteed to maximize data likelihood (locally)
  - acoustic models (GMMs) can be trained too
- **Discriminant training considers 'wrong' models**
  - recognition requires improved margin
  - frame-discriminant neural nets give better model discrimination?