Algorithms for NLP

Speech Inference

Taylor Berg-Kirkpatrick – CMU

Slides: Dan Klein – UC Berkeley

Project Announcements

- Due date postponed: now due Sat 9/23 at 11:59pm
- Will be using Canvas for jar and write-up submission
  - We will test as soon as this is set up
  - Invites will be sent to everyone (will announce)
- Extra jar submission of your best system
  - No spot-checks for extra jar… feel free to use approximations
- Instructions for submission will be added to website
- If using open-address w/ long keys, try this hash:
  - `int hash = ((int) (key ^ (key >>> 32)) * 3875239);`

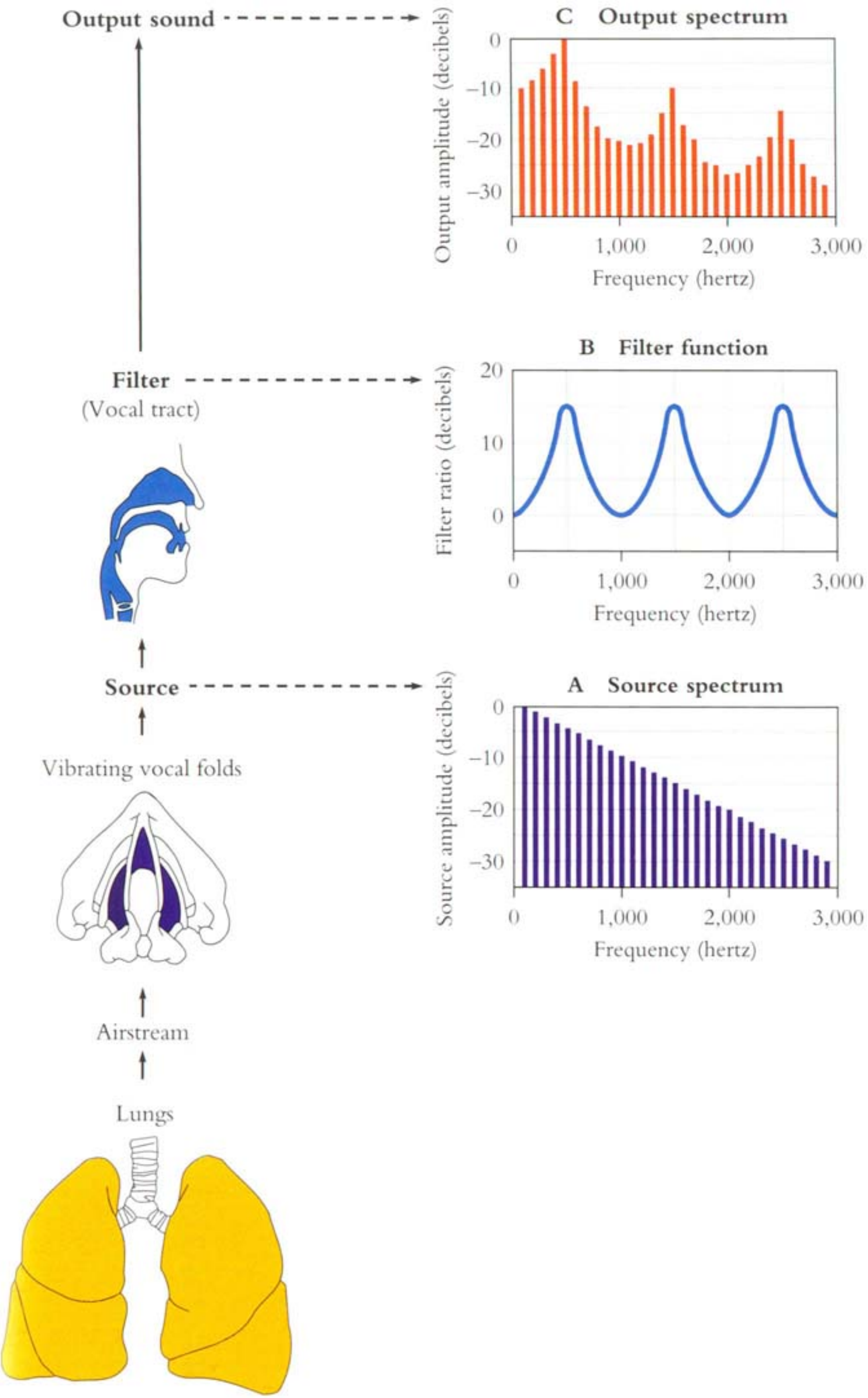
Project Grading

- Late days: 5 total, use whenever
  - But no credit for late submissions when you run out of late days!
  - (Be careful!)

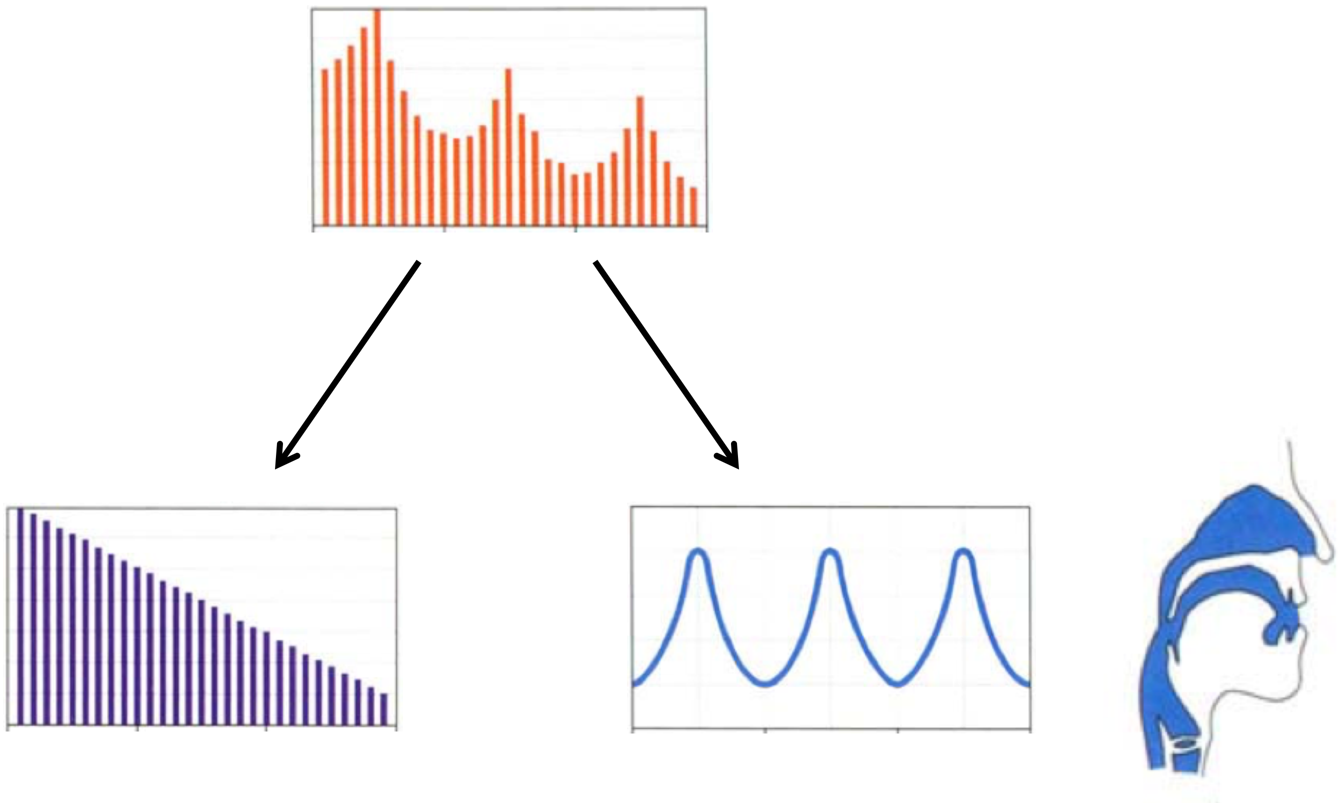
- Grading: Projects out of 10
  - 6 Points: Successfully implemented what we asked
  - 2 Points: Submitted a reasonable write-up
  - 1 Point: Write-up is written clearly
  - 1 Point: Substantially exceeded minimum metrics
  - Extra Credit: Did non-trivial extension to project

Source / Filter

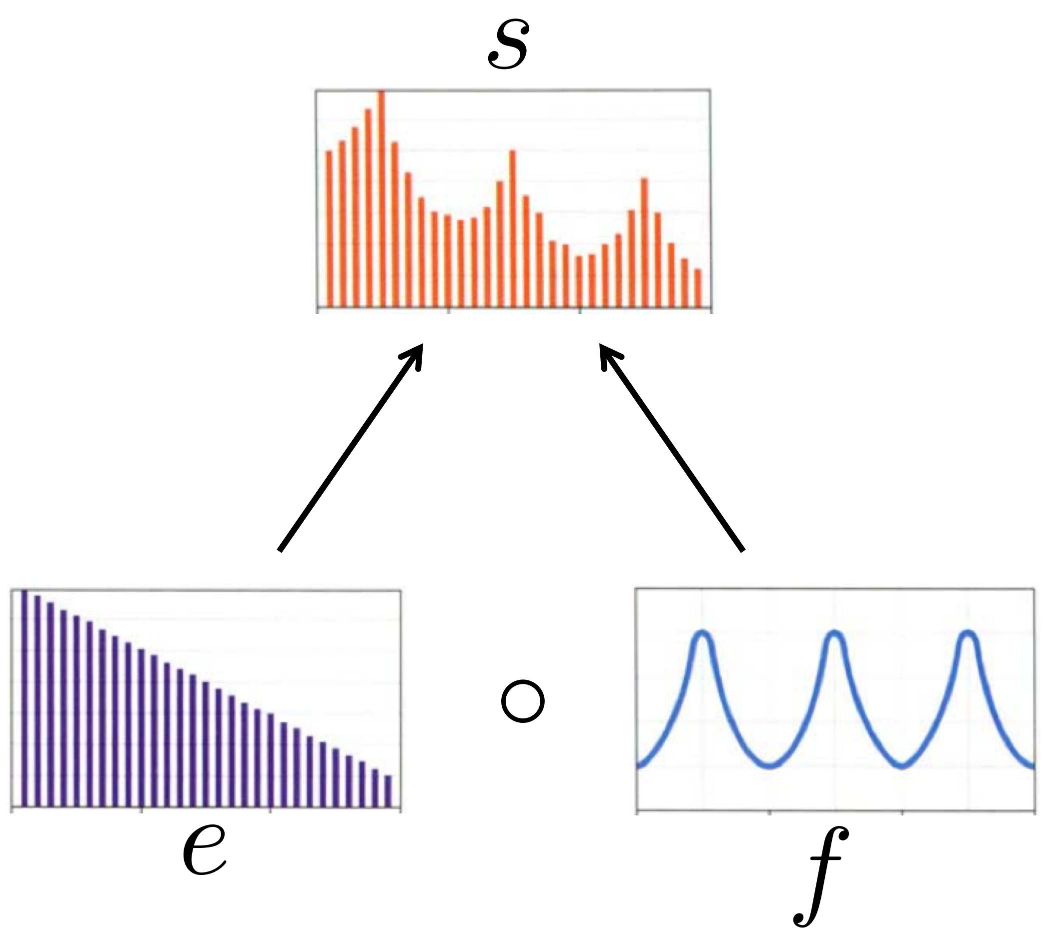
- Articulation process:
  - The vocal cord vibrations create harmonics
  - The mouth is an amplifier
  - Depending on shape of mouth, some harmonics are amplified more than others



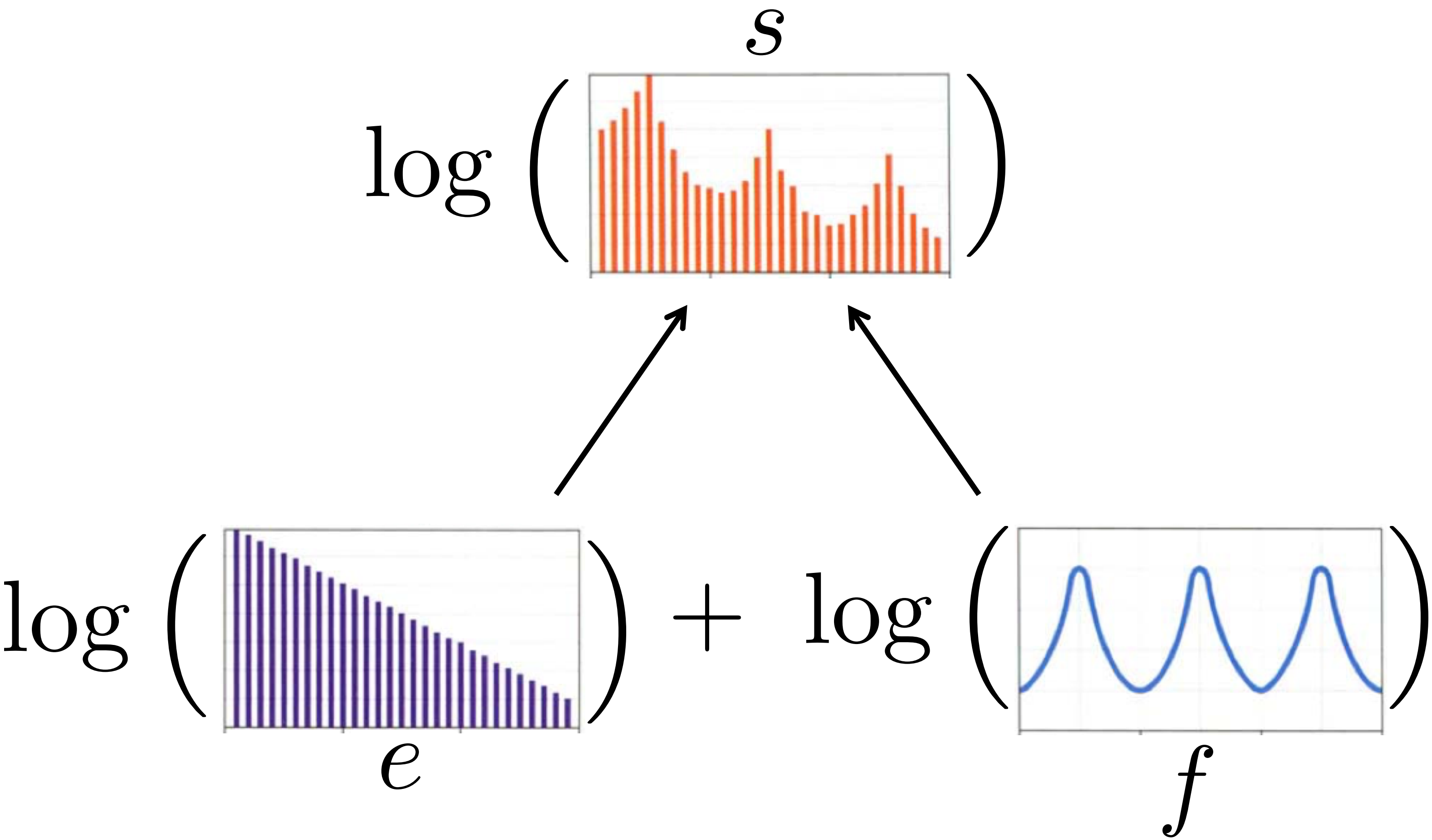
Deconvolution / Liftering



Deconvolution / Liftering



Deconvolution / Liftering

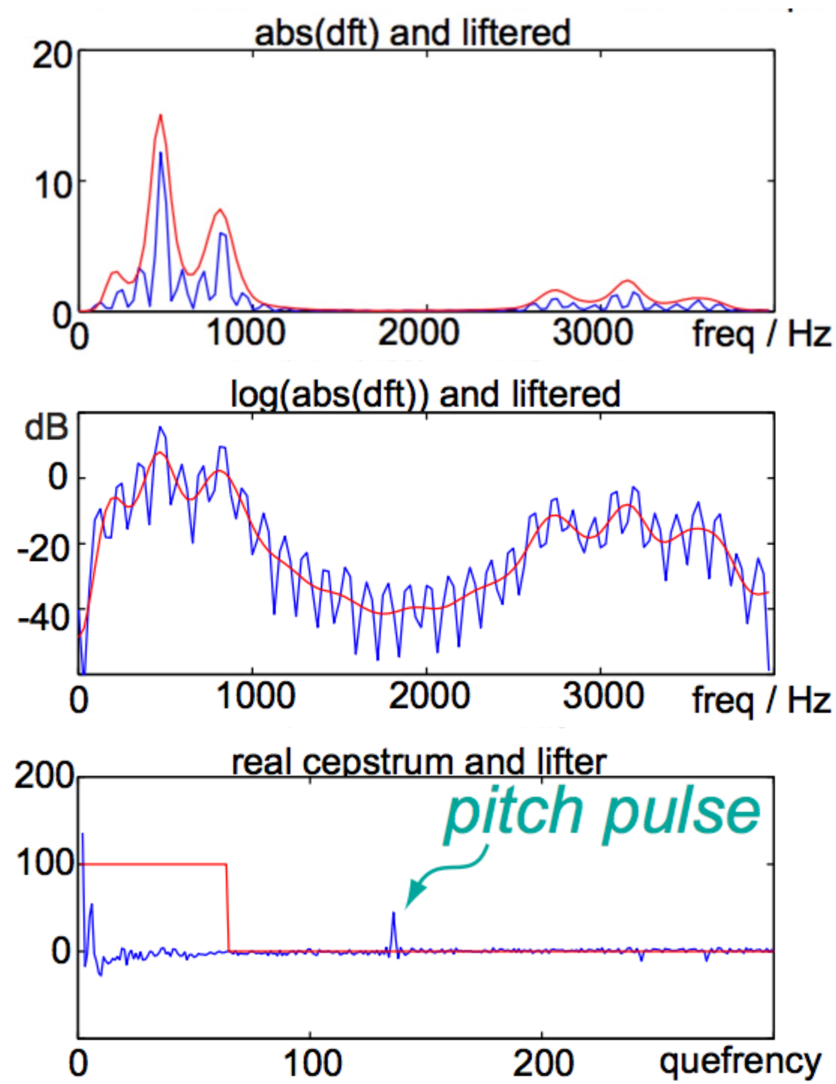


Deconvolution / Liftering

$$s = e \circ f$$

$$\log(s) = \log(e) + \log(f)$$

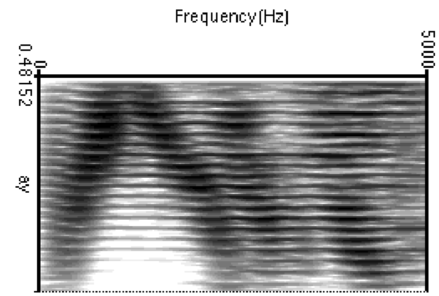
$$\mathrm{IDFT}(\log(s))$$

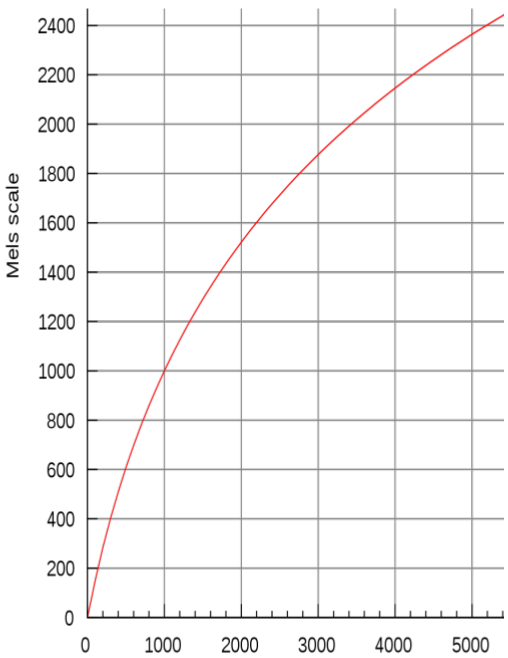


Graphs from Dan Ellis

Mel Freq. Cepstral Coefficients

- Do FFT to get spectral information
  - Like the spectrogram we saw earlier
- Apply Mel scaling (New)
  - Models human ear; more sensitivity in lower freqs
  - Approx linear below 1kHz, log above, equal samples above and below 1kHz
- Take Log
- Do discrete cosine transform





[Graph: Wikipedia]

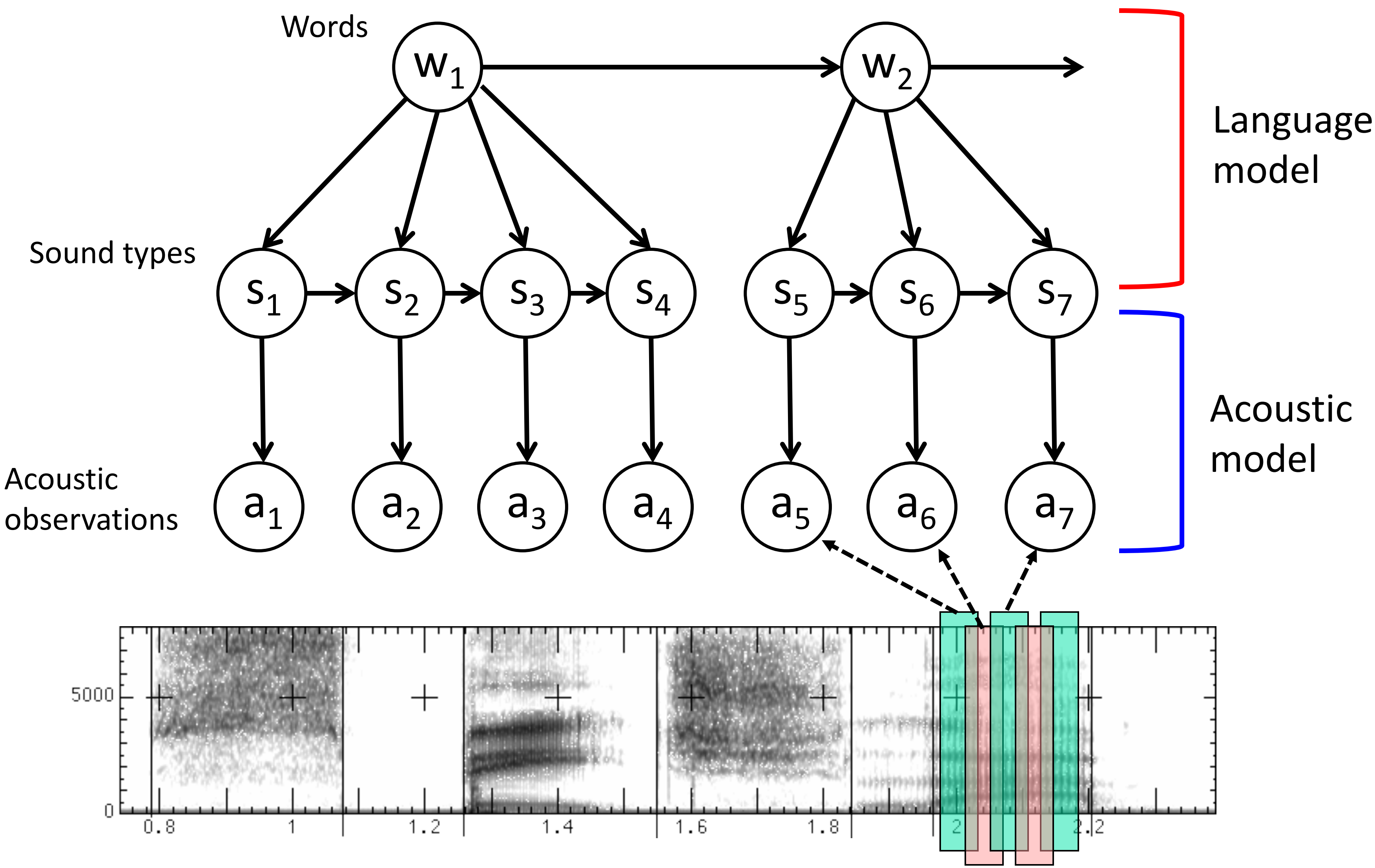
Final Feature Vector

- 39 (real) features per 10 ms frame:
  - 12 MFCC features
  - 12 delta MFCC features
  - 12 delta-delta MFCC features
  - 1 (log) frame energy
  - 1 delta (log) frame energy
  - 1 delta-delta (log frame energy)

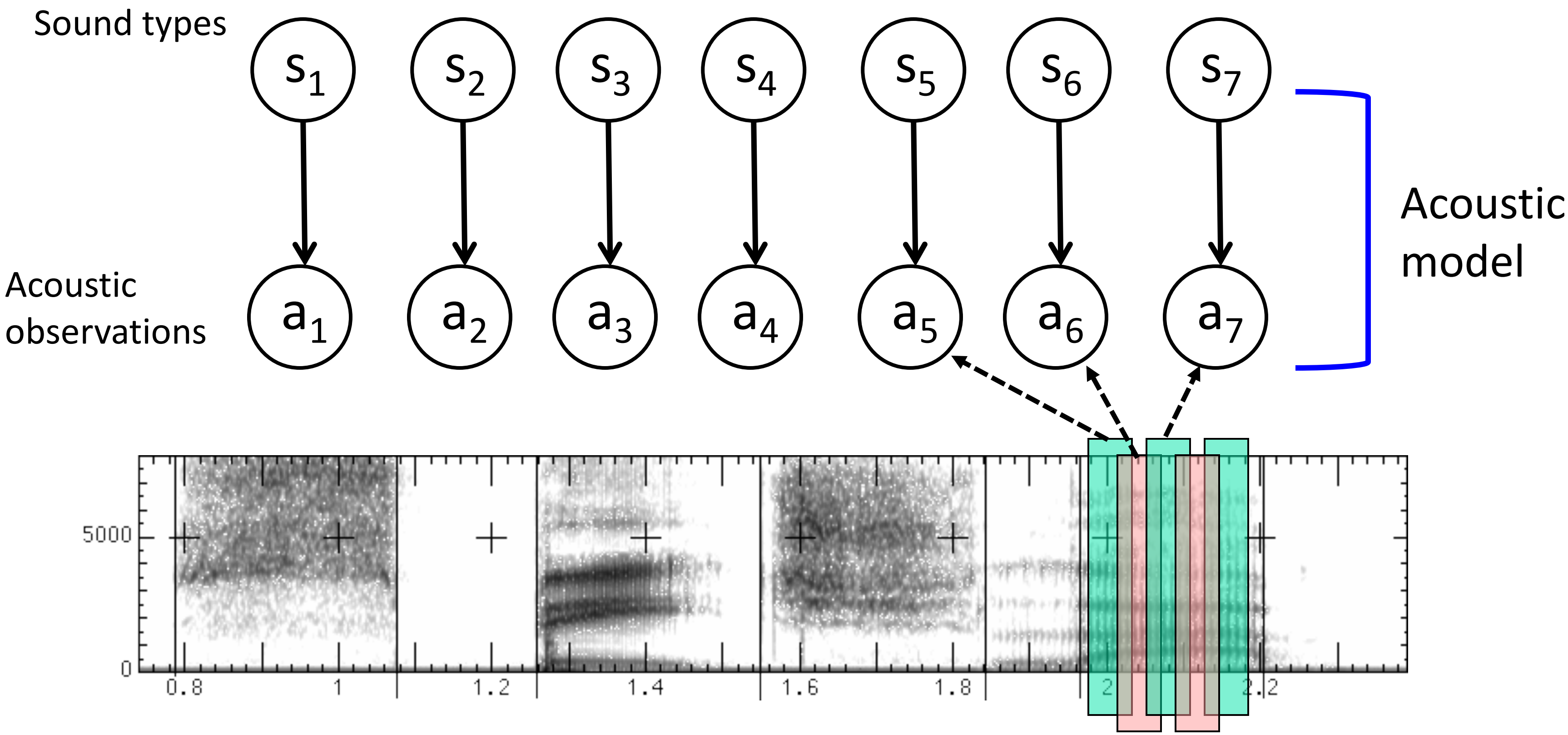
- So each frame is represented by a 39D vector

Acoustic Model

Speech Model

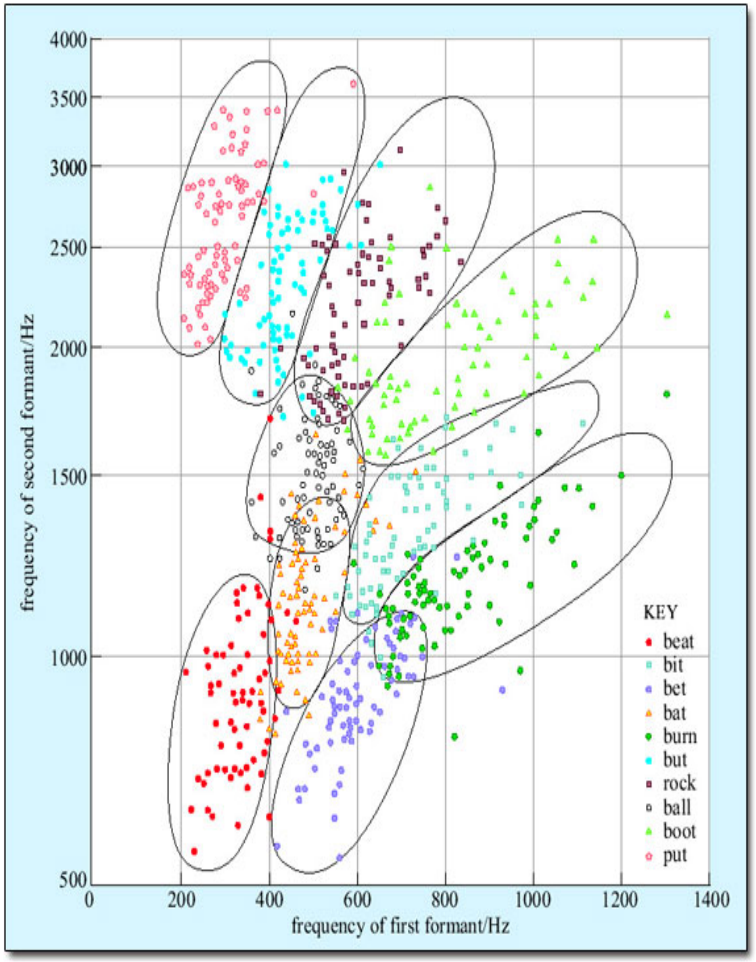


Acoustic Model



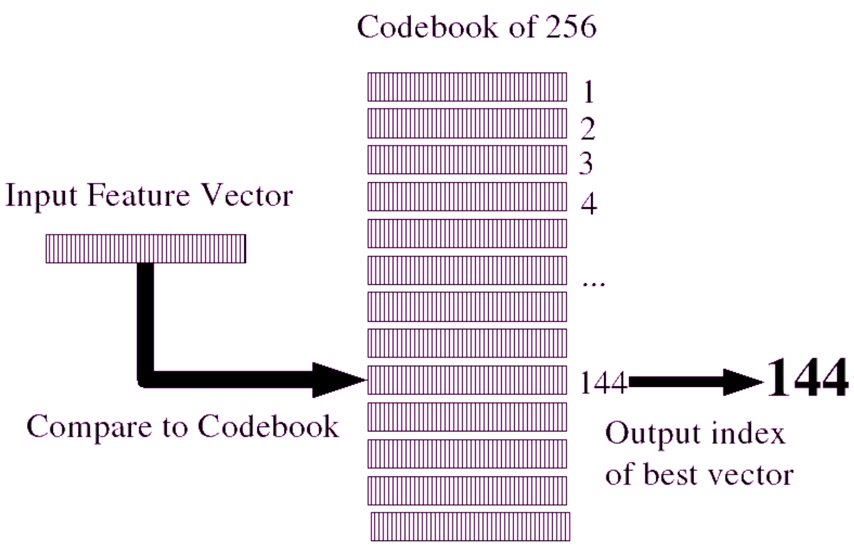
HMMs for Continuous Observations

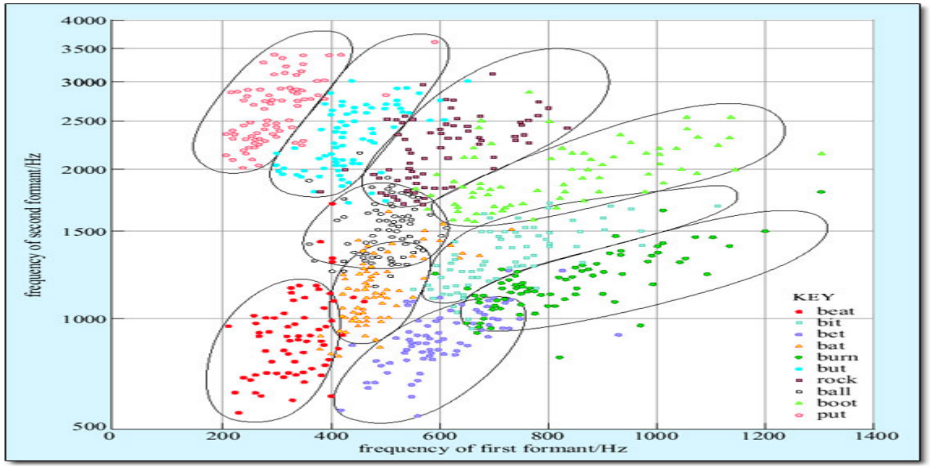
- Before: discrete set of observations
- Now: feature vectors are real-valued
- Solution 1: discretization
- Solution 2: continuous emissions
  - Gaussians
  - Multivariate Gaussians
  - Mixtures of multivariate Gaussians
- A state is progressively
  - Context independent subphone (~3 per phone)
  - Context dependent phone (triphones)
  - State tying of CD phone



Vector Quantization

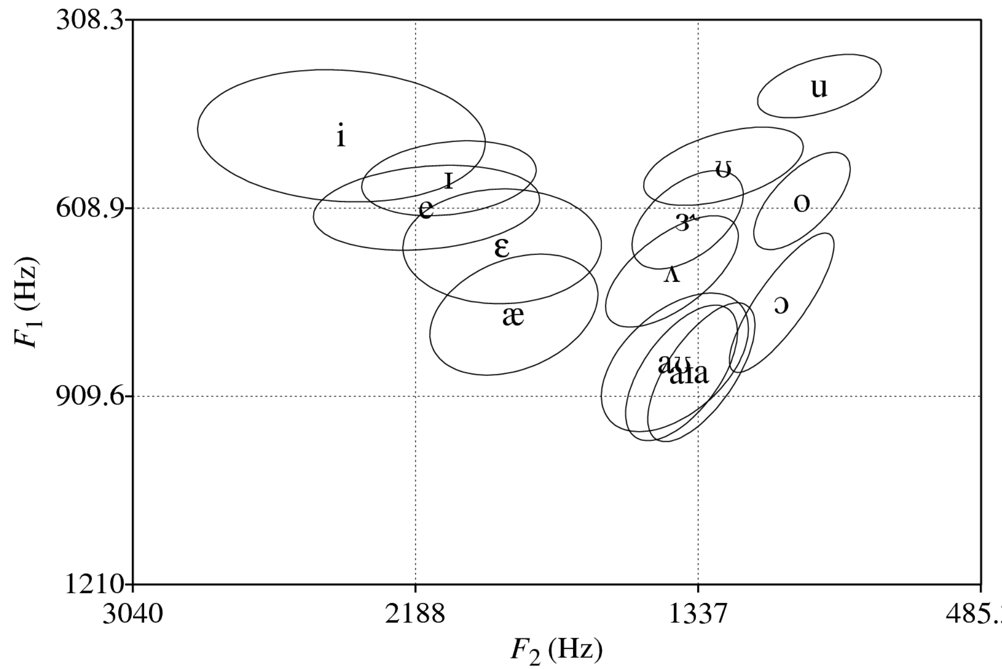
- Idea: discretization
  - Map MFCC vectors onto discrete symbols
  - Compute probabilities just by counting
- This is called vector quantization or VQ
- Not used for ASR any more
- But: useful to consider as a starting point





Gaussian Emissions

- VQ is insufficient for top-quality ASR
  - Hard to cover high-dimensional space with codebook
  - Moves ambiguity from the model to the preprocessing
- Instead: assume the possible values of the observation vectors are normally distributed.
  - Represent the observation likelihood function as a Gaussian?



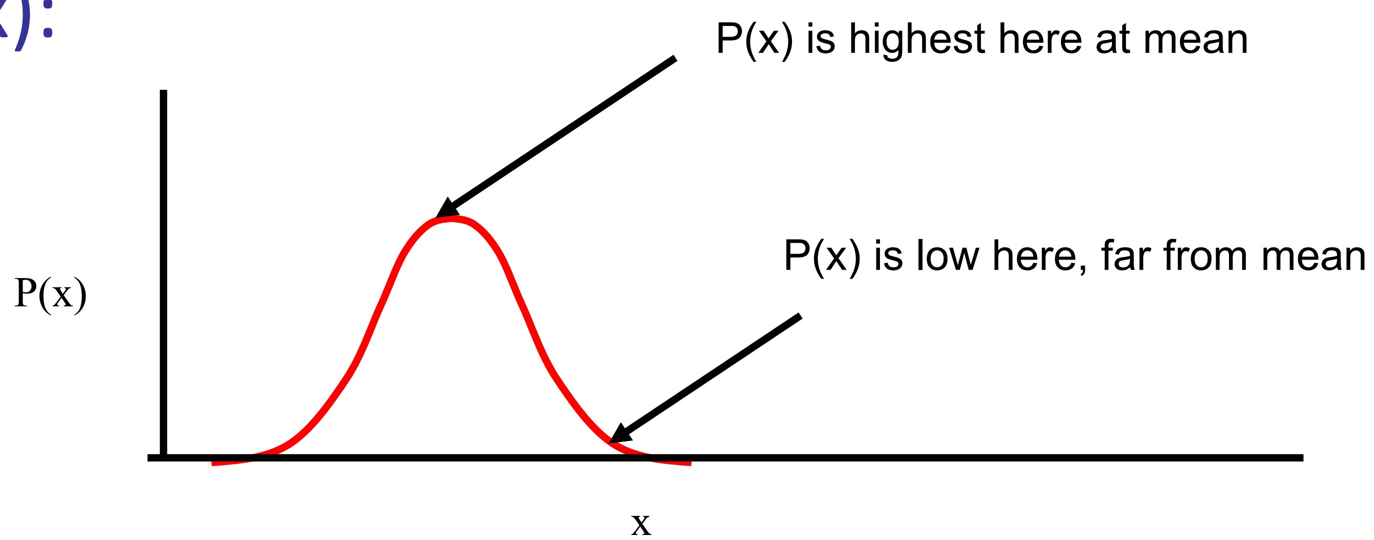
From bartus.org/akustyk

Gaussians for Acoustic Modeling

A Gaussian is parameterized by a mean and a variance:

$$P(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- P(x):



Multivariate Gaussians

- Instead of a single mean μ and variance σ^2:

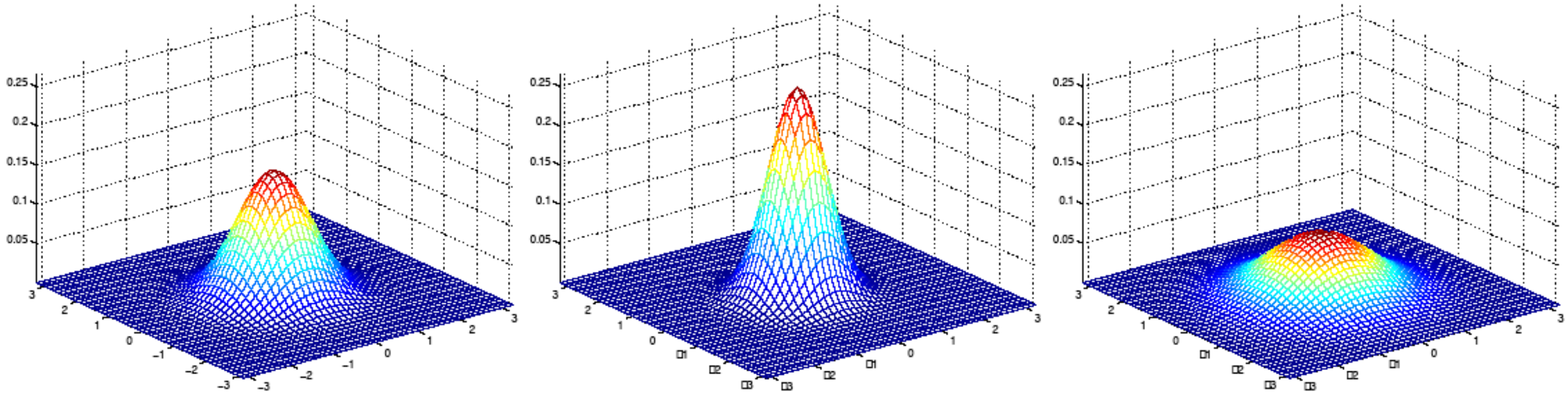
$$P(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Vector of means μ and covariance matrix Σ

$$P(x|\mu,\Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$$

- Usually assume diagonal covariance (!)
  - This isn't very true for FFT features, but is less bad for MFCC features

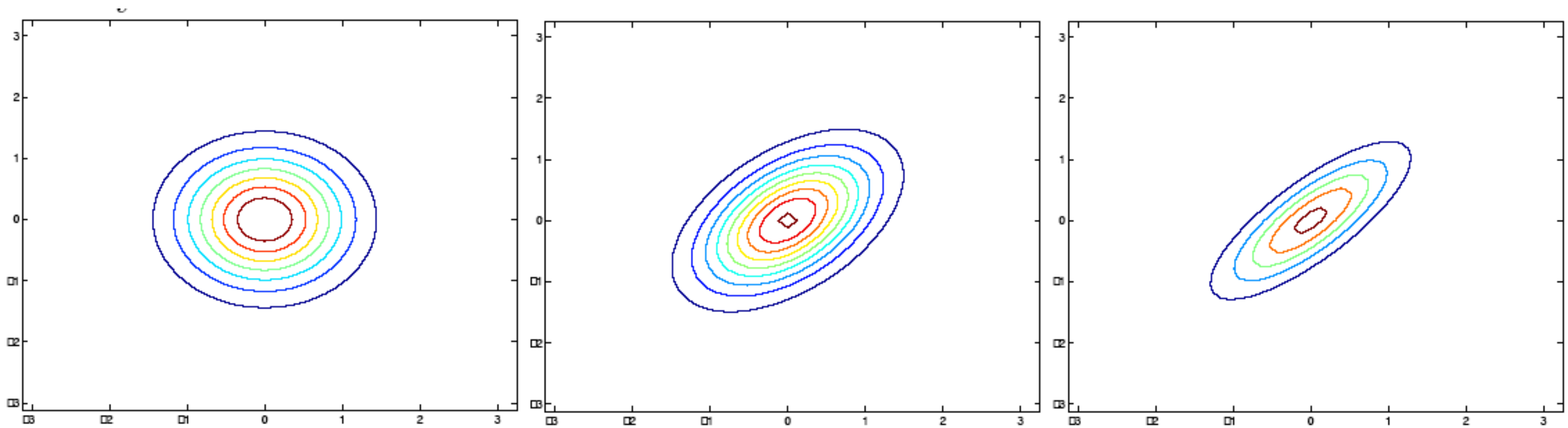
Gaussians: Size of Σ



- $\mu = [0\ 0]$ $\qquad$ $\mu = [0\ 0]$ $\qquad$ $\mu = [0\ 0]$
- $\Sigma = I$ $\qquad$ $\Sigma = 0.6I$ $\qquad$ $\Sigma = 2I$
- As Σ becomes larger, Gaussian becomes more spread out; as Σ becomes smaller, Gaussian more compressed

Text and figures from Andrew Ng

Gaussians: Shape of Σ



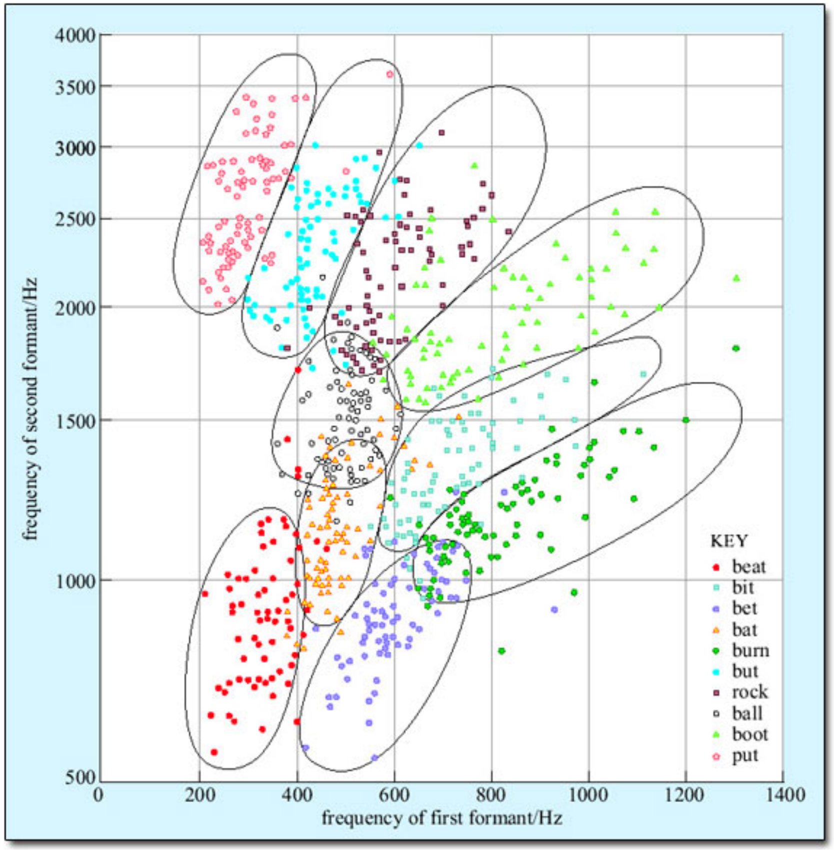
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}; \quad .\Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

- As we increase the off diagonal entries, more correlation between value of x and value of y

Text and figures from Andrew Ng

But we're not there yet

- Single Gaussians may do a bad job of modeling a complex distribution in any dimension
- Even worse for diagonal covariances
- Solution: mixtures of Gaussians



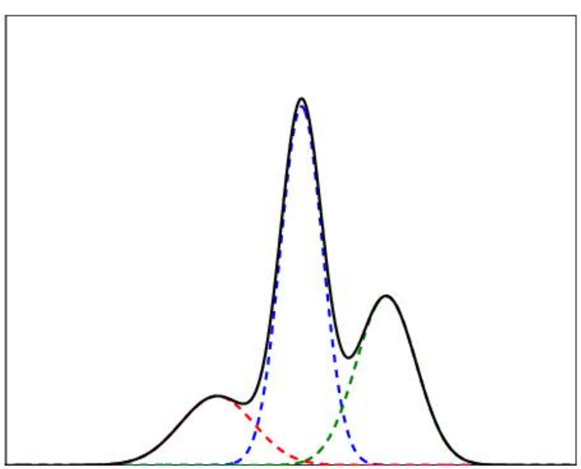
From openlearn.open.ac.uk

Mixtures of Gaussians

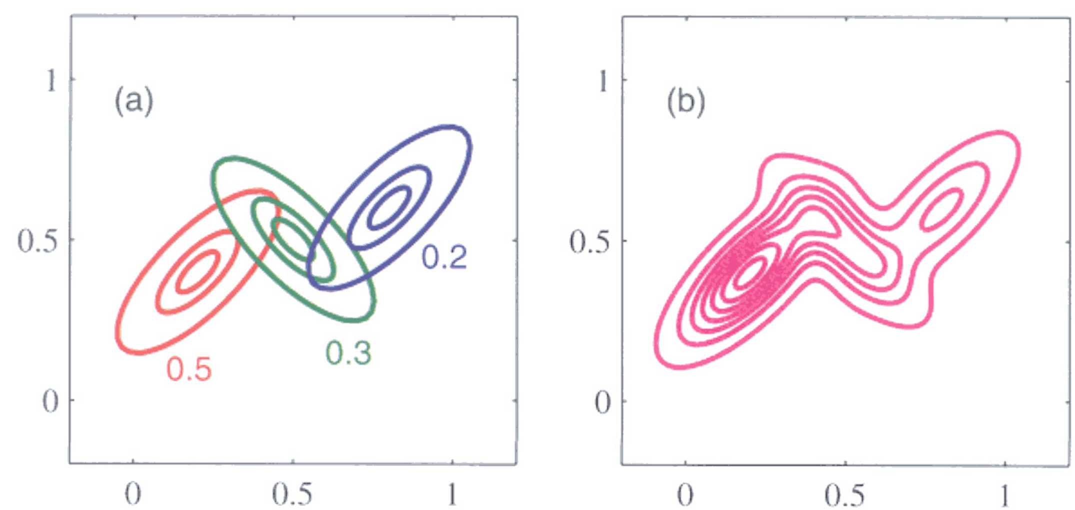
- Mixtures of Gaussians:

$$P(x|\mu_i,\Sigma_i) = \frac{1}{(2\pi)^{k/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^\top \Sigma_i^{-1}(x-\mu_i)\right)$$

$$P(x|\mu,\mathbf{\Sigma},\mathbf{c}) = \sum_i c_i P(x|\mu_i,\Sigma_i)$$



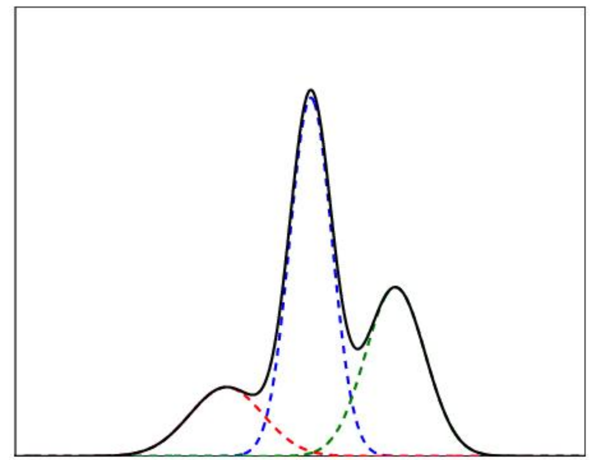
From robots.ox.ac.uk



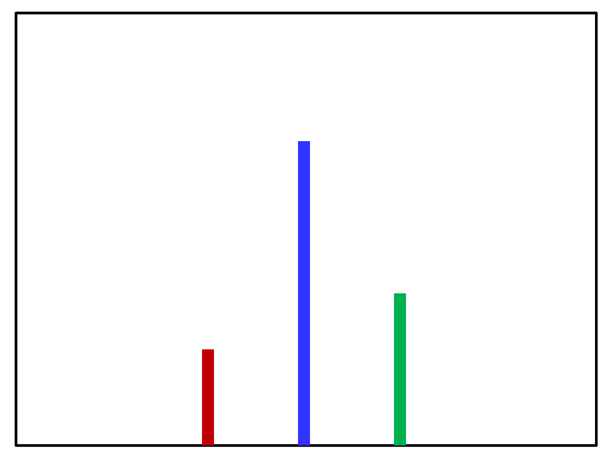
http://www.itee.uq.edu.au/~comp4702

GMMs

- Summary: each state has an emission distribution P(x|s) (likelihood function) parameterized by:
  - M mixture weights
  - M mean vectors of dimensionality D
  - Either M covariance matrices of DxD or M Dx1 diagonal variance vectors



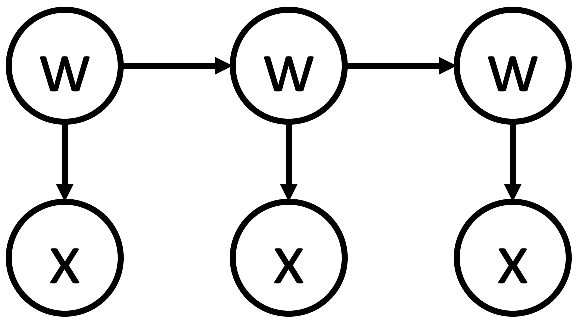
- Like soft vector quantization after all
  - Think of the mixture means as being learned codebook entries
  - Think of the Gaussian densities as a learned codebook distance function
  - Think of the mixture of Gaussians like a multinomial over codes
  - (Even more true given shared Gaussian inventories… more soon)



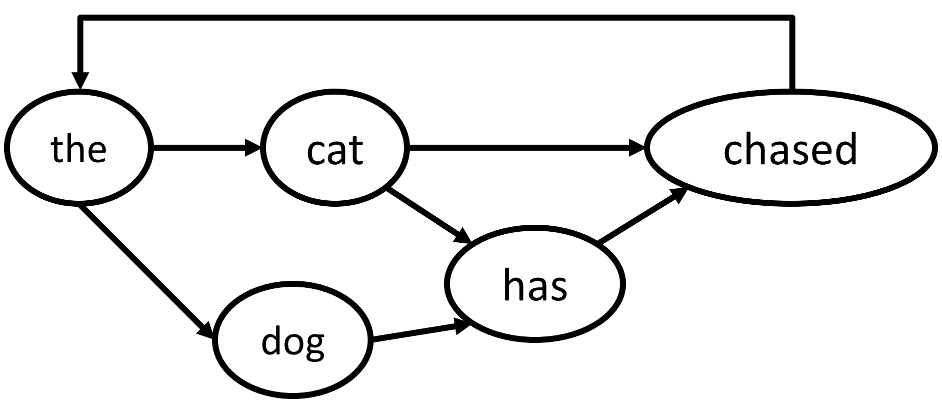
State Model

State Transition Diagrams

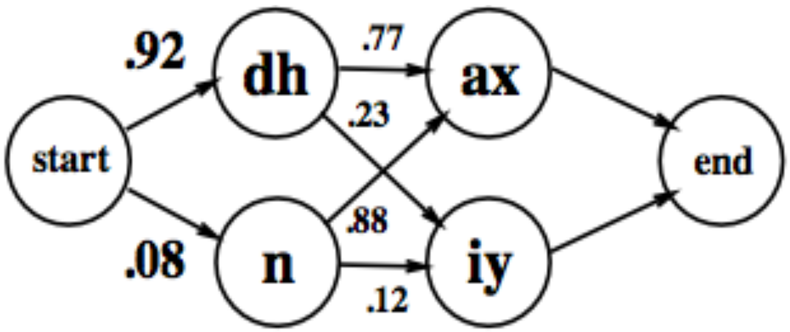
- Bayes Net: HMM as a Graphical Model



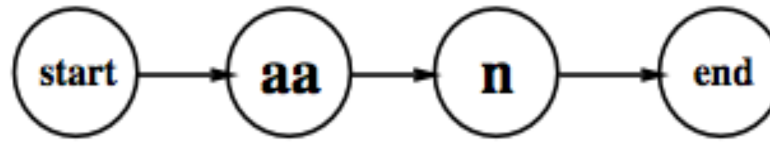
- State Transition Diagram: Markov Model as a Weighted FSA



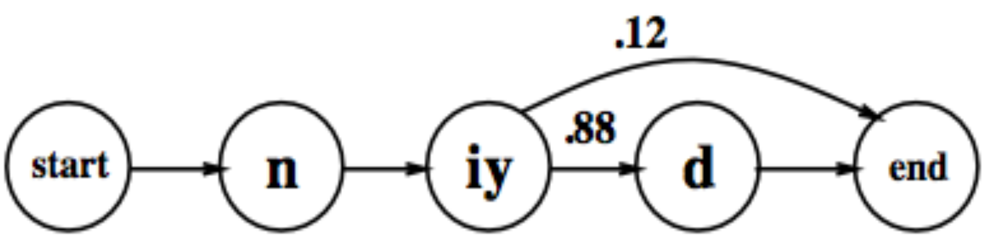
ASR Lexicon



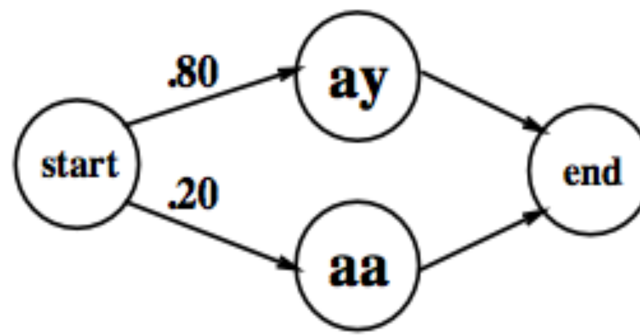
Word model for "the"



Word model for "on"



Word model for "need"



Word model for "I"

Figure: J & M

Lexical State Structure

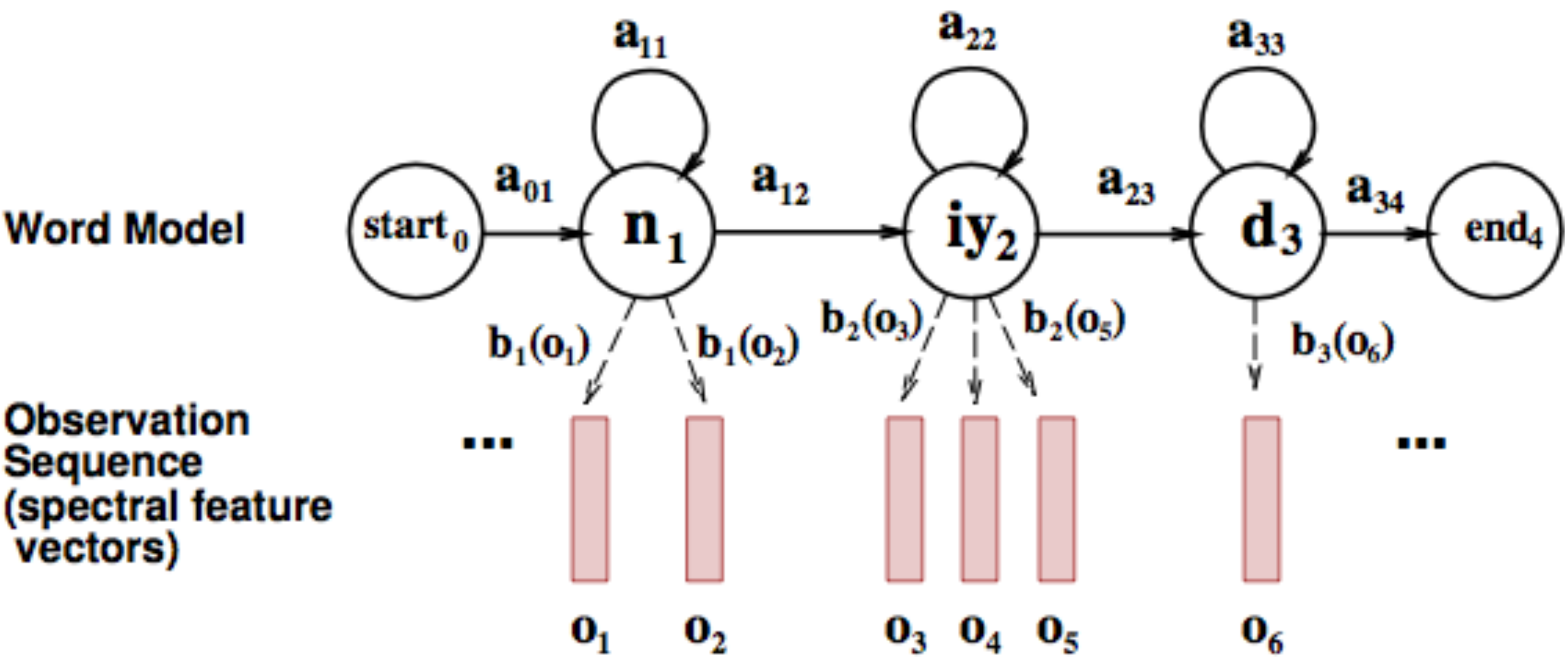


Figure: J & M

Adding an LM

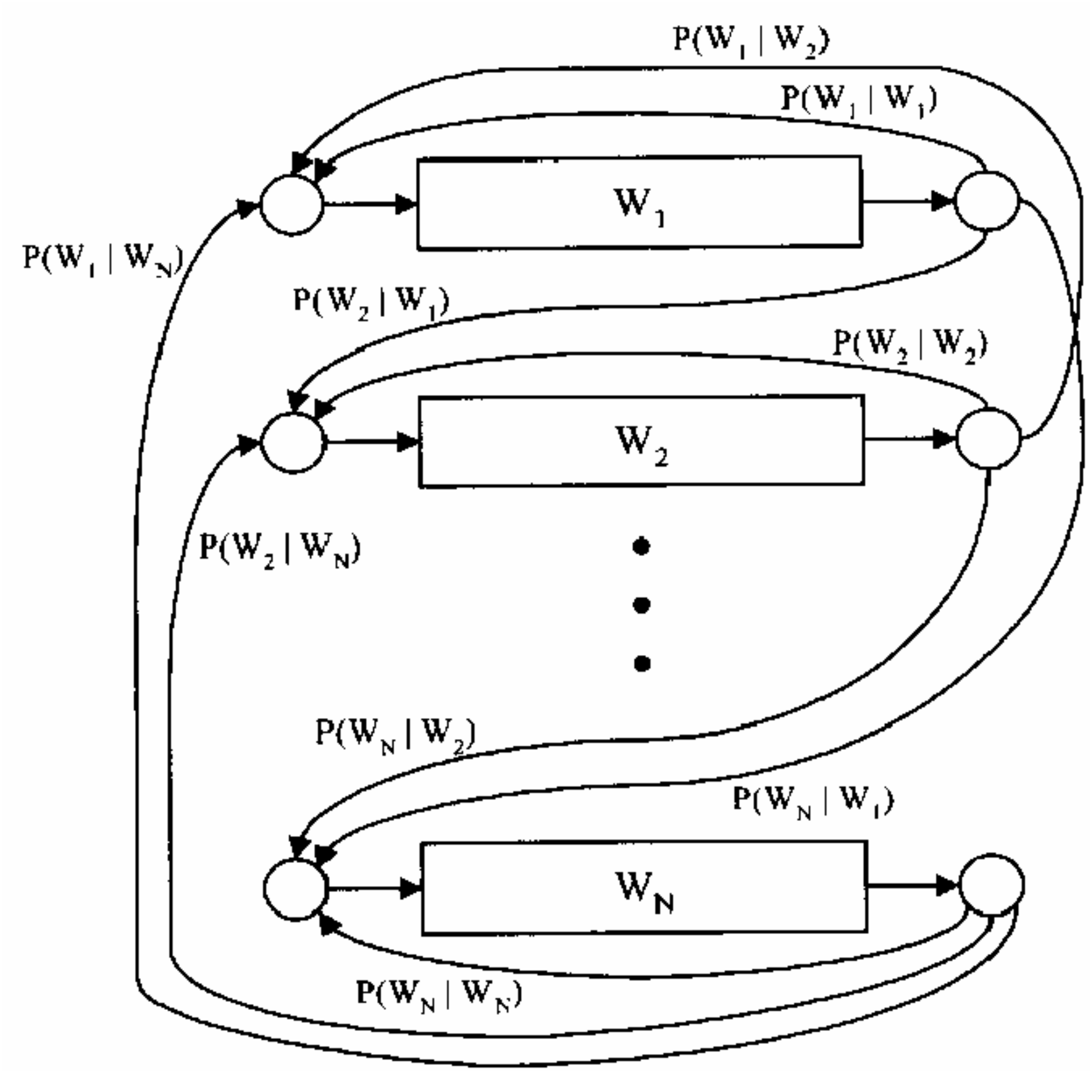


Figure from Huang et al page 618

State Space

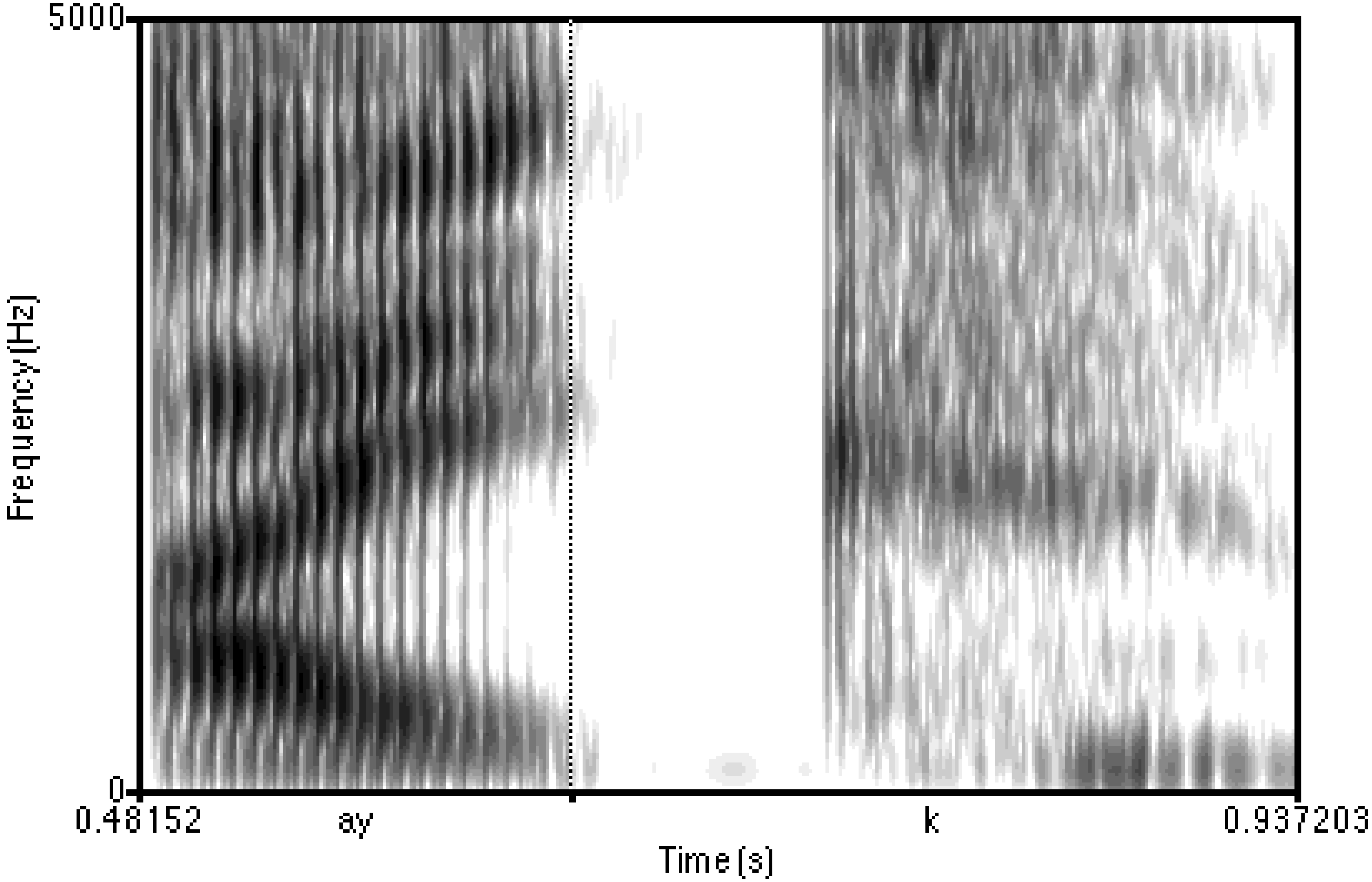
- State space must include
  - Current word (|V| on order of 20K+)
  - Index within current word (|L| on order of 5)
  - E.g. (lec[t]ure) (though not in orthography!)

- Acoustic probabilities only depend on phone type
  - E.g. P(x|lec[t]ure) = P(x|t)

- From a state sequence, can read a word sequence

State Refinement

Phones Aren't Homogeneous



Need to Use Subphones

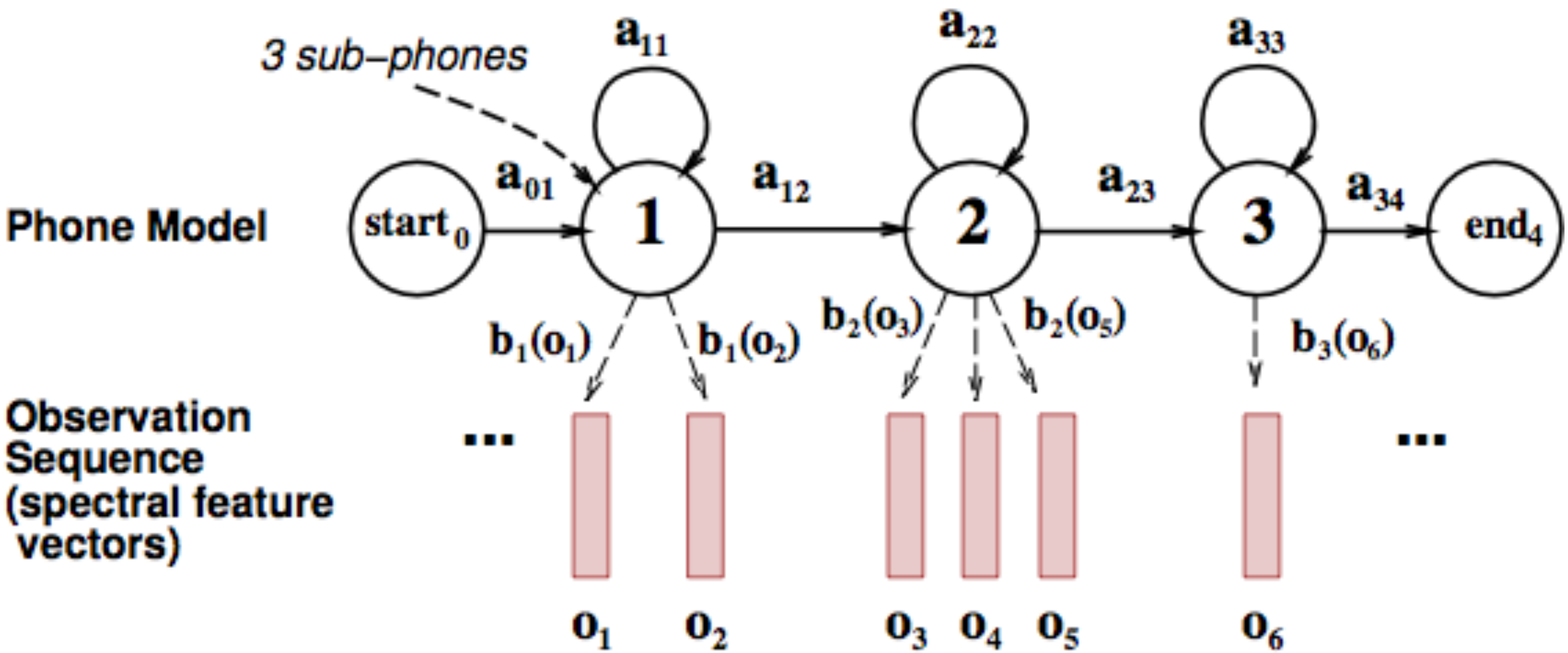


Figure: J & M

A Word with Subphones

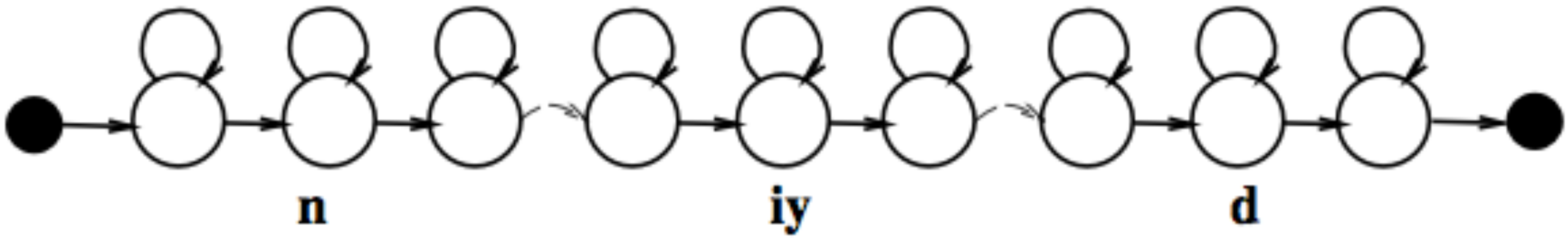
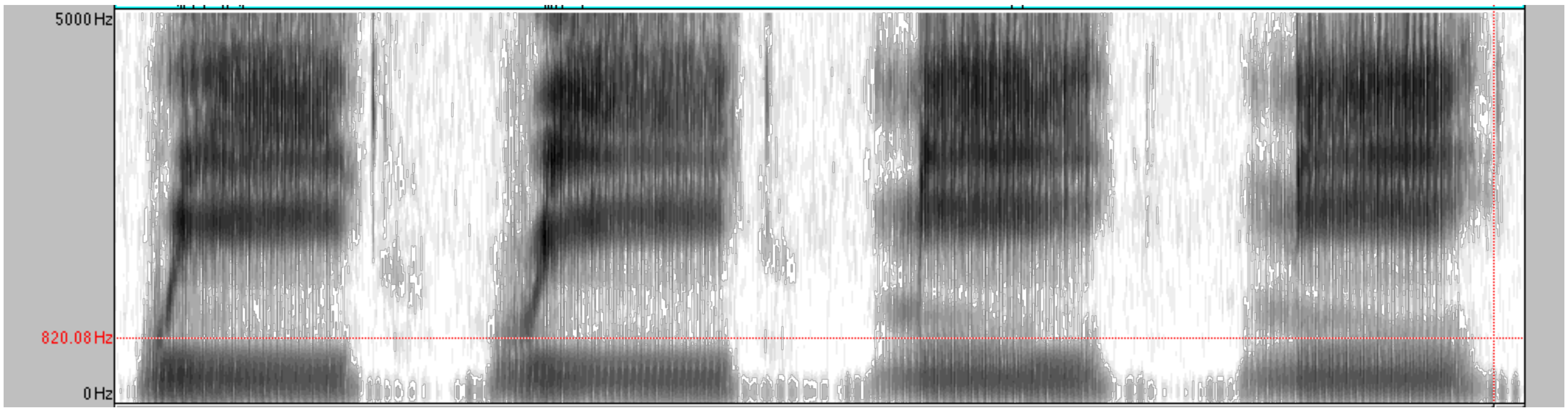


Figure: J & M

Modeling phonetic context



w iy r iy m iy n iy

“Need” with triphone models

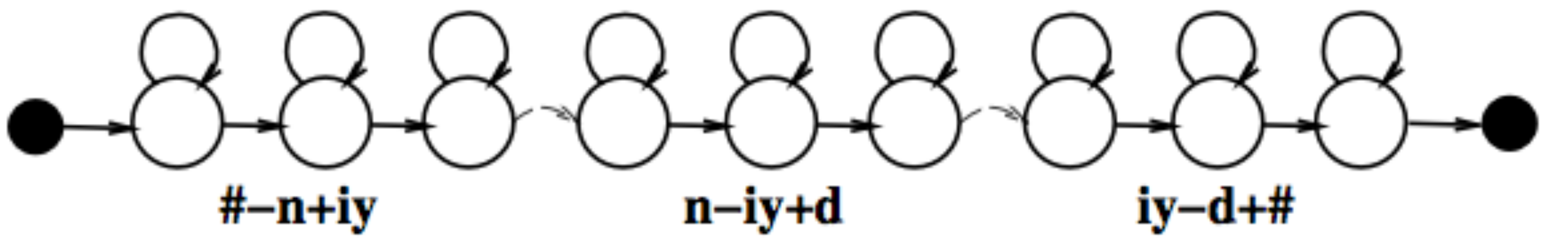


Figure: J & M

Lots of Triphones

- Possible triphones: 50x50x50=125,000
- How many triphone types actually occur?
- 20K word WSJ Task (from Bryan Pellom)
  - Word internal models: need 14,300 triphones
  - Cross word models: need 54,400 triphones
- Need to generalize models, tie triphones

State Tying / Clustering

- [Young, Odell, Woodland 1994]
- How do we decide which triphones to cluster together?
- Use phonetic features (or 'broad phonetic classes')
  - Stop
  - Nasal
  - Fricative
  - Sibilant
  - Vowel
  - lateral

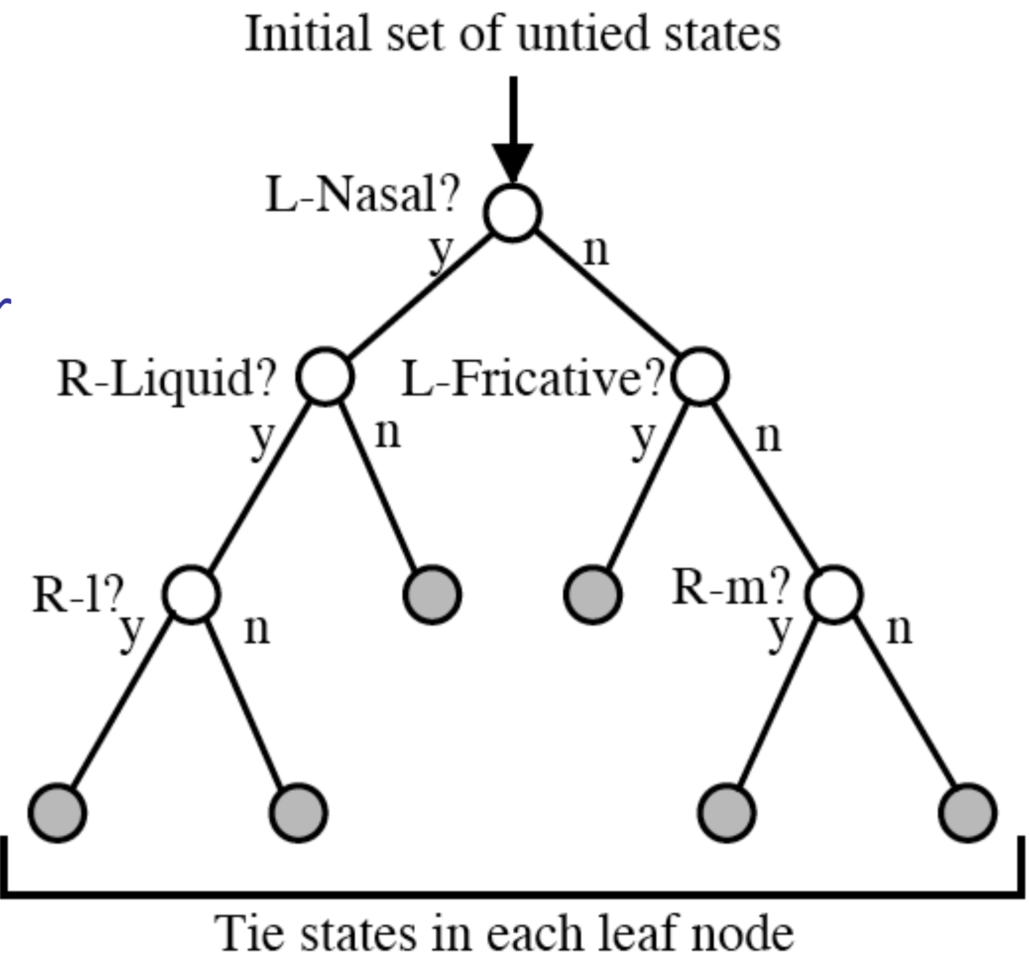


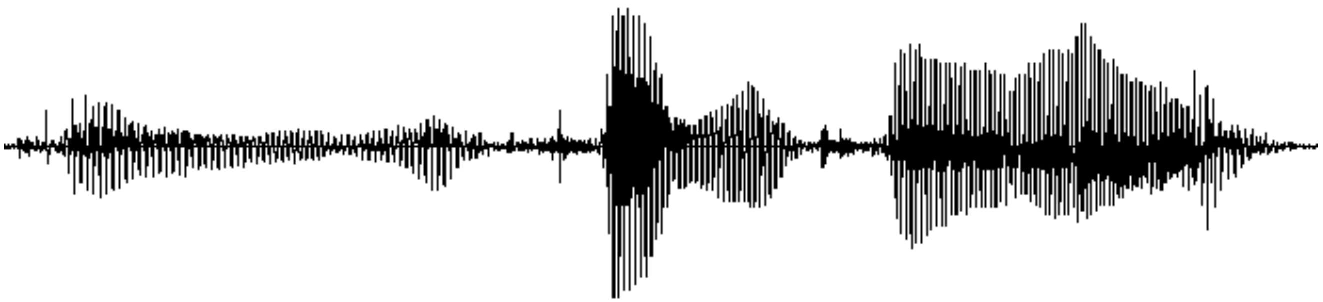
Figure: J & M

State Space

- State space now includes
  - Current word: |W| is order 20K
  - Index in current word: |L| is order 5
  - Subphone position: 3
  - E.g. (lec[t-mid]ure)
- Acoustic model depends on clustered phone context
  - But this doesn't grow the state space
- But, adding the LM context for trigram+ does
  - (after the, lec[t-mid]ure)
  - This is a real problem for decoding

Decoding

Inference Tasks



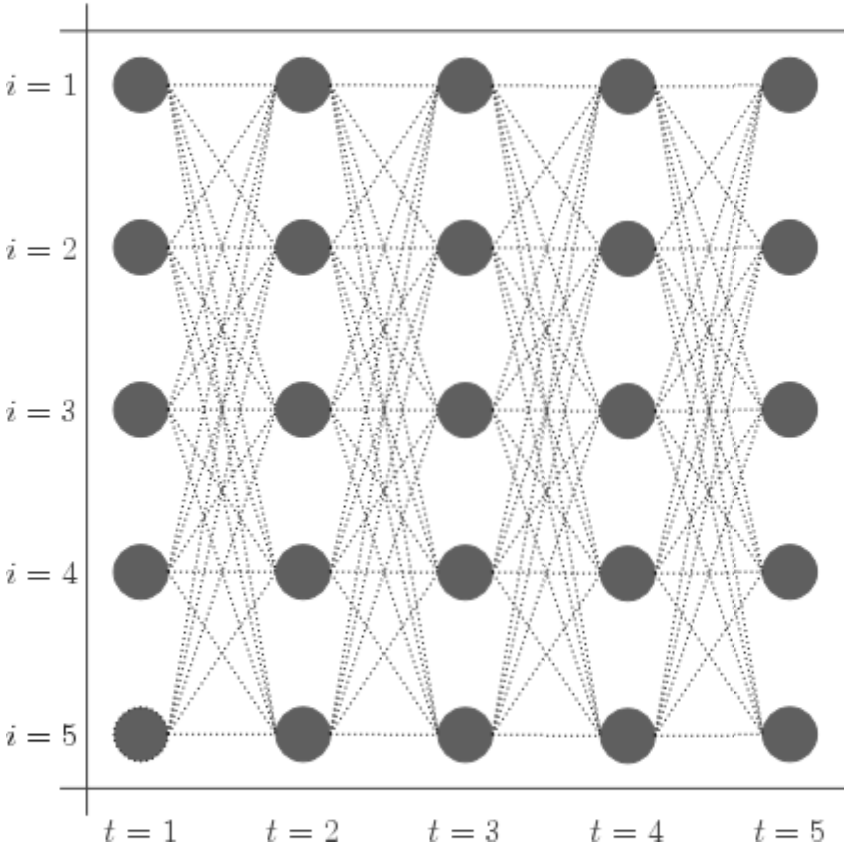
Most likely word sequence:

d - ae - d

Most likely state sequence:

d_1-d_6-d_6-d_4-ae_5-ae_2-ae_3-ae_0-d_2-d_2-d_3-d_7-d_5

Viterbi Decoding



$$\phi_t(s_t, s_{t-1}) = P(x_t|s_t)P(s_t|s_{t-1})$$

$$v_t(s_t) = \max_{s_{t-1}} \phi_t(s_t, s_{t-1}) v_{t-1}(s_{t-1})$$

Figure: Enrique Benimeli

Viterbi Decoding

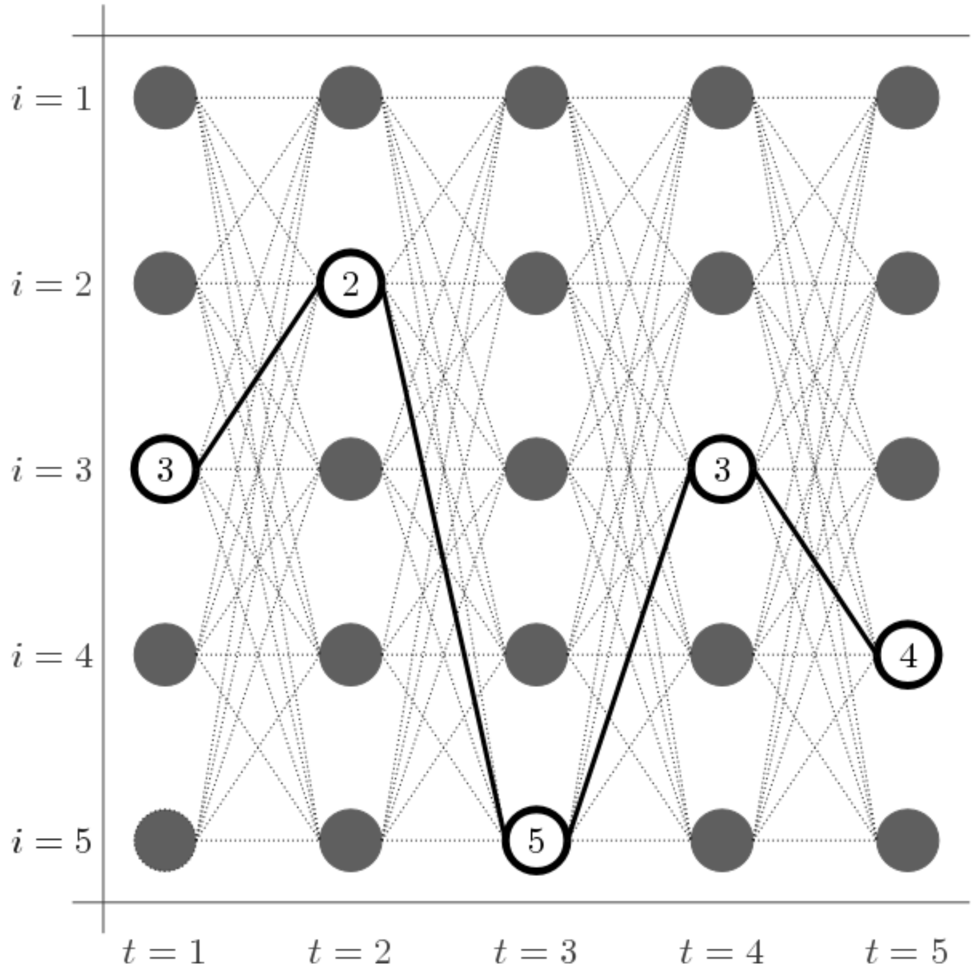
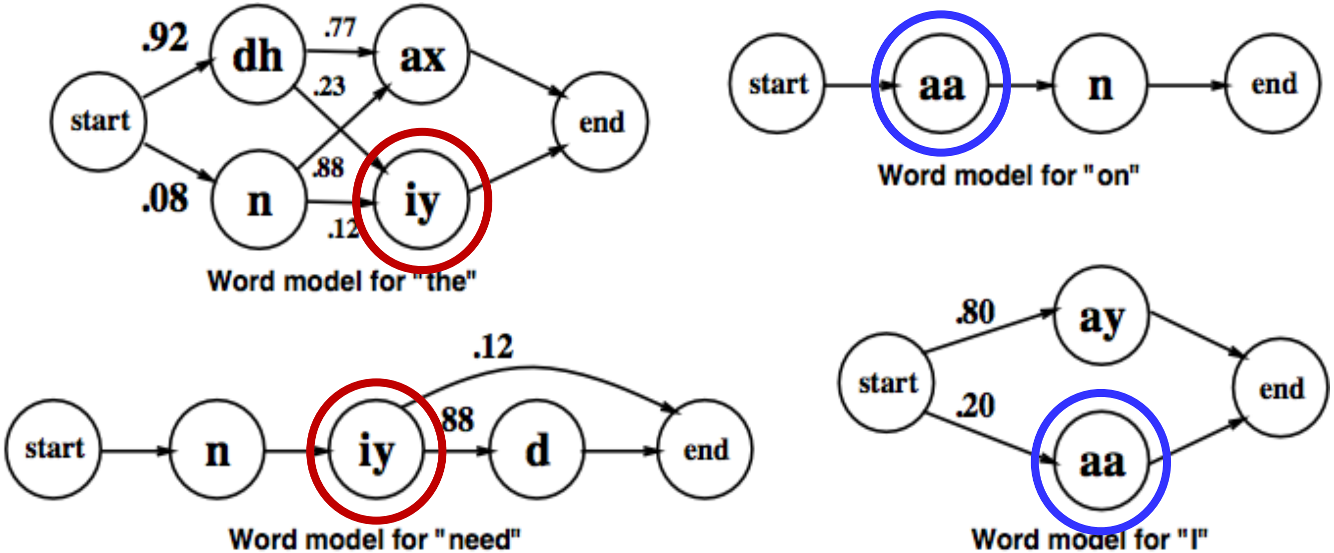


Figure: Enrique Benimeli

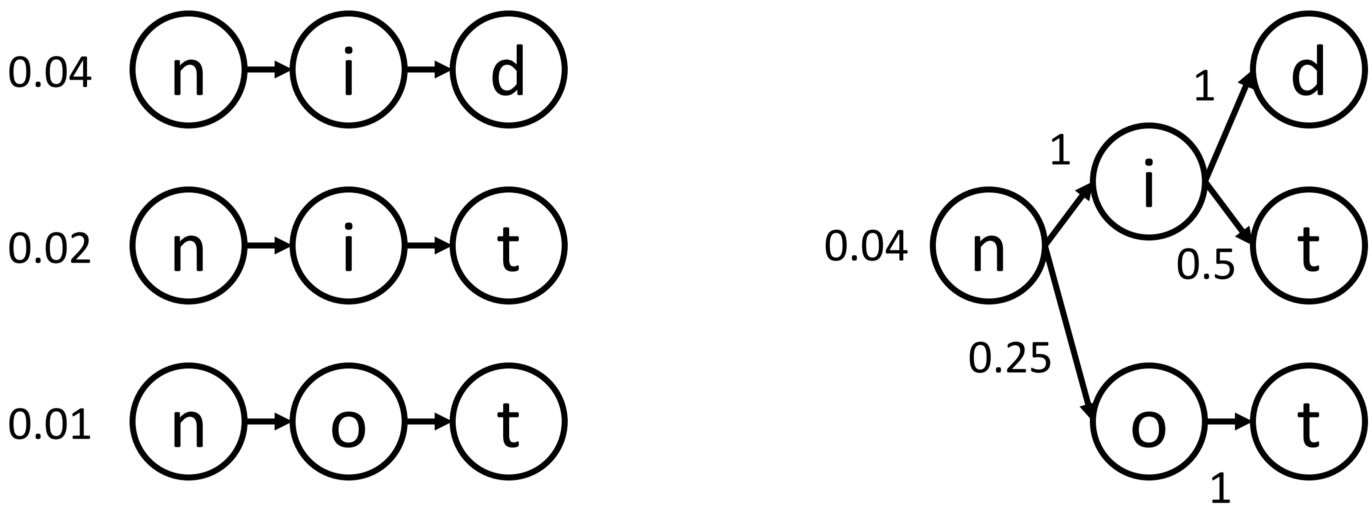
Emission Caching

- Problem: scoring all the P(x|s) values is too slow
- Idea: many states share tied emission models, so cache them



Prefix Trie Encodings

- Problem: many partial-word states are indistinguishable
- Solution: encode word production as a prefix trie (with pushed weights)



- A specific instance of minimizing weighted FSAs [Mohri, 94]

Figure: Aubert, 02

Beam Search

- Problem: trellis is too big to compute v(s) vectors
- Idea: most states are terrible, keep v(s) only for top states at each time

the b.
the m.
and then.
at then.

the ba.
the be.
the bi.

the ma.
the me.
the mi.

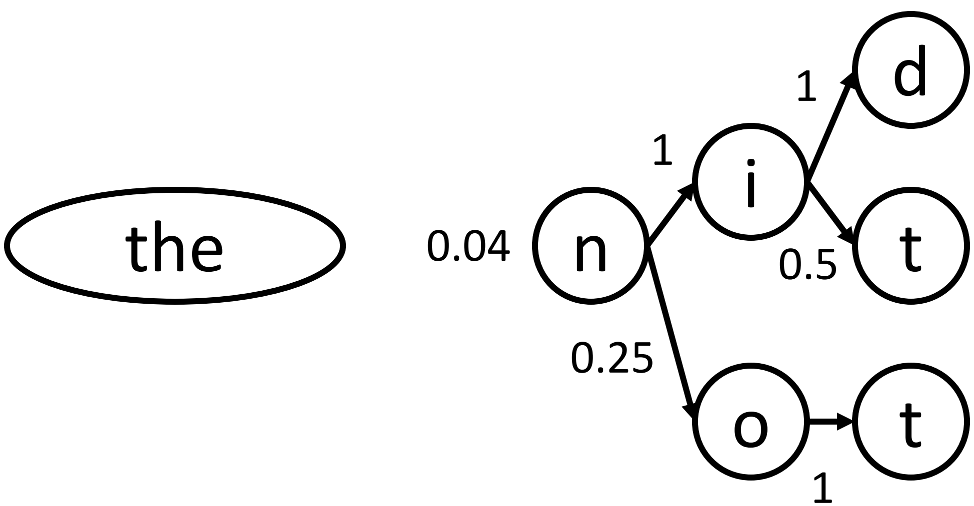
then a.
then e.
then i.

the ba.
the be.
the ma.
then a.

- Important: still dynamic programming; collapse equiv states

LM Factoring

- Problem: Higher-order n-grams explode the state space
- (One) Solution:
  - Factor state space into (word index, lm history)
  - Score unigram prefix costs while inside a word
  - Subtract unigram cost and add trigram cost once word is complete



LM Reweighting

- Noisy channel suggests

$$P(x|w)P(w)$$

- In practice, want to boost LM

$$P(x|w)P(w)^{\alpha}$$

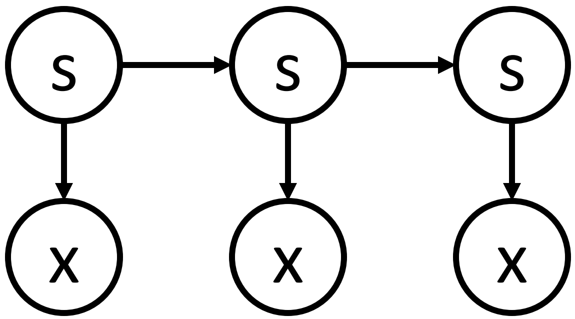
- Also, good to have a “word bonus” to offset LM costs

$$P(x|w)P(w)^{\alpha}|w|^{\beta}$$

- These are both consequences of broken independence assumptions in the model

Training

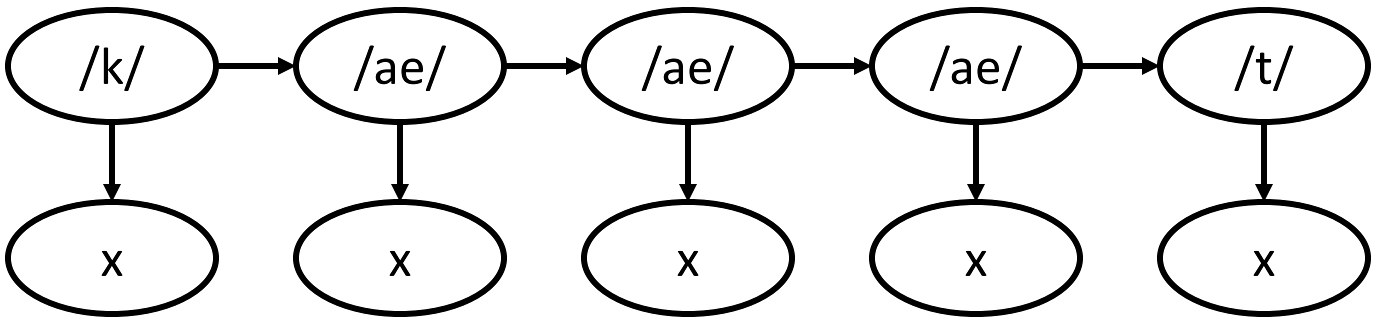
What Needs to be Learned?



- Emissions: P(x | phone class)
  - X is MFCC-valued
- Transitions: P(state | prev state)
  - If between words, this is P(word | history)
  - If inside words, this is P(advance | phone class)
  - (Really a hierarchical model)

Estimation from Aligned Data

- What if each time step was labeled with its (context-dependent sub) phone?



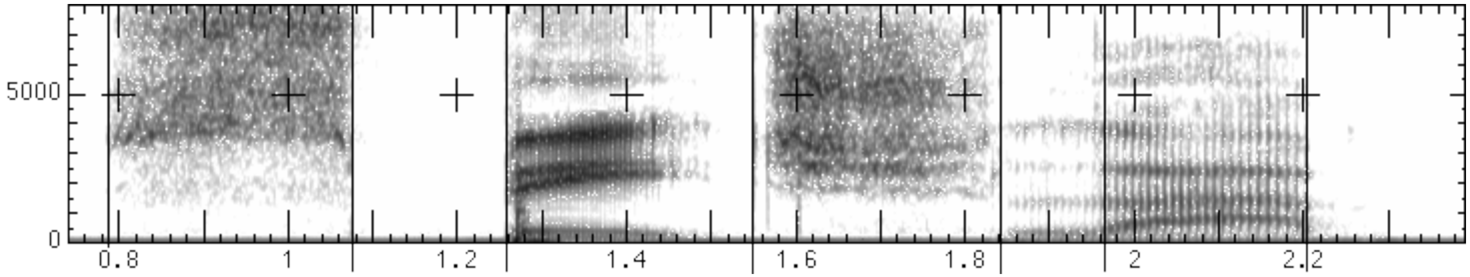
- Can estimate P(x|/ae/) as empirical mean and (co-)variance of x's with label /ae/
- Problem: Don't know alignment at the frame and phone level

Forced Alignment

- What if the acoustic model P(x|phone) was known?
  - … and also the correct sequences of words / phones
- Can predict the best alignment of frames to phones

"speech lab"

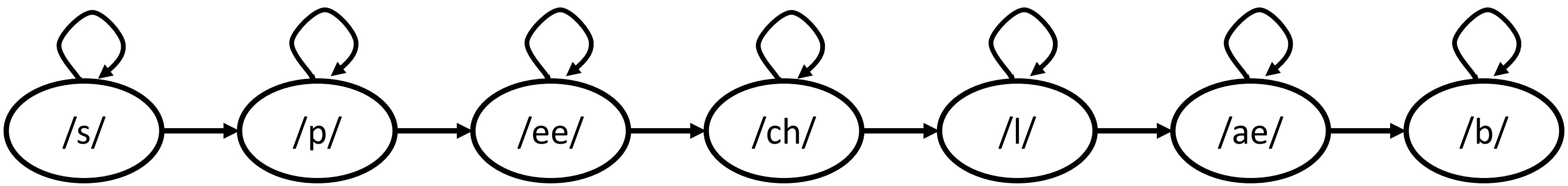
sssssssspppeeeeeeetshshshshllllaeaeaebbbbb



- Called "forced alignment"

Forced Alignment

- Create a new state space that forces the hidden variables to transition through phones in the (known) order



- Still have uncertainty about durations

- In this HMM, all the parameters are known
  - Transitions determined by known utterance
  - Emissions assumed to be known
  - Minor detail: self-loop probabilities

- Just run Viterbi (or approximations) to get the best alignment

EM for Alignment

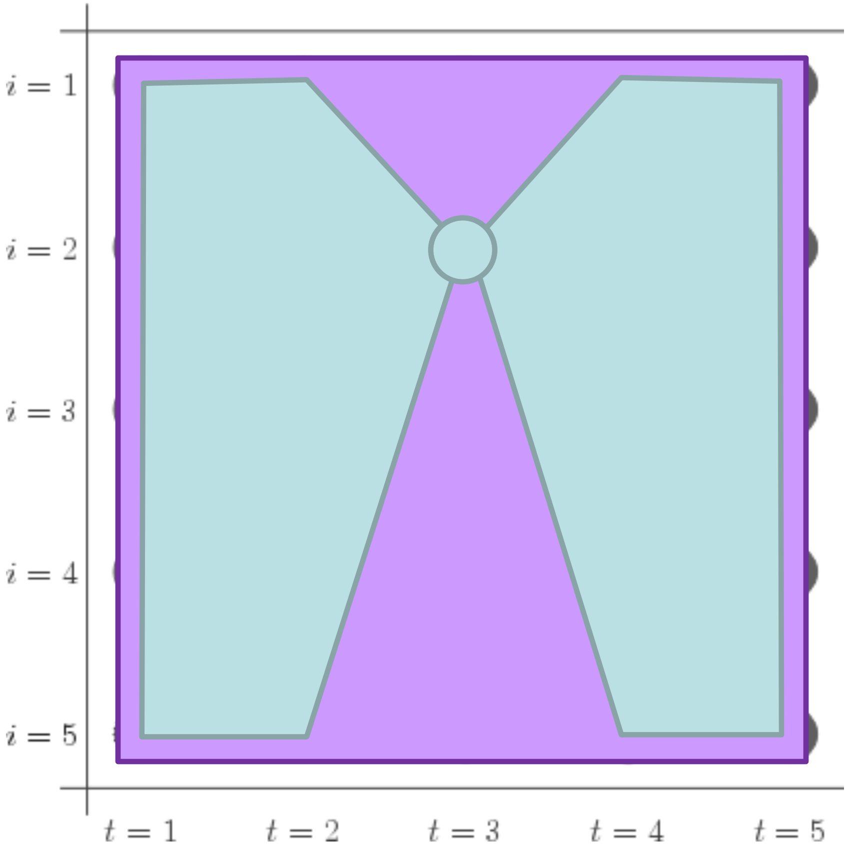
- Input: acoustic sequences with word-level transcriptions
- We don’t know either the emission model or the frame alignments
- Expectation Maximization (Hard EM for now)
  - Alternating optimization
  - Impute completions for unlabeled variables (here, the states at each time step)
  - Re-estimate model parameters (here, Gaussian means, variances, mixture ids)
  - Repeat
  - One of the earliest uses of EM!

Soft EM

- Hard EM uses the best single completion
  - Here, single best alignment
  - Not always representative
  - Certainly bad when your parameters are initialized and the alignments are all tied
  - Uses the count of various configurations (e.g. how many tokens of /ae/ have self-loops)

- What we'd really like is to know the fraction of paths that include a given completion
  - E.g. 0.32 of the paths align this frame to /p/, 0.21 align it to /ee/, etc.
  - Formally want to know the expected count of configurations
  - Key quantity: $P(s_t \mid x)$

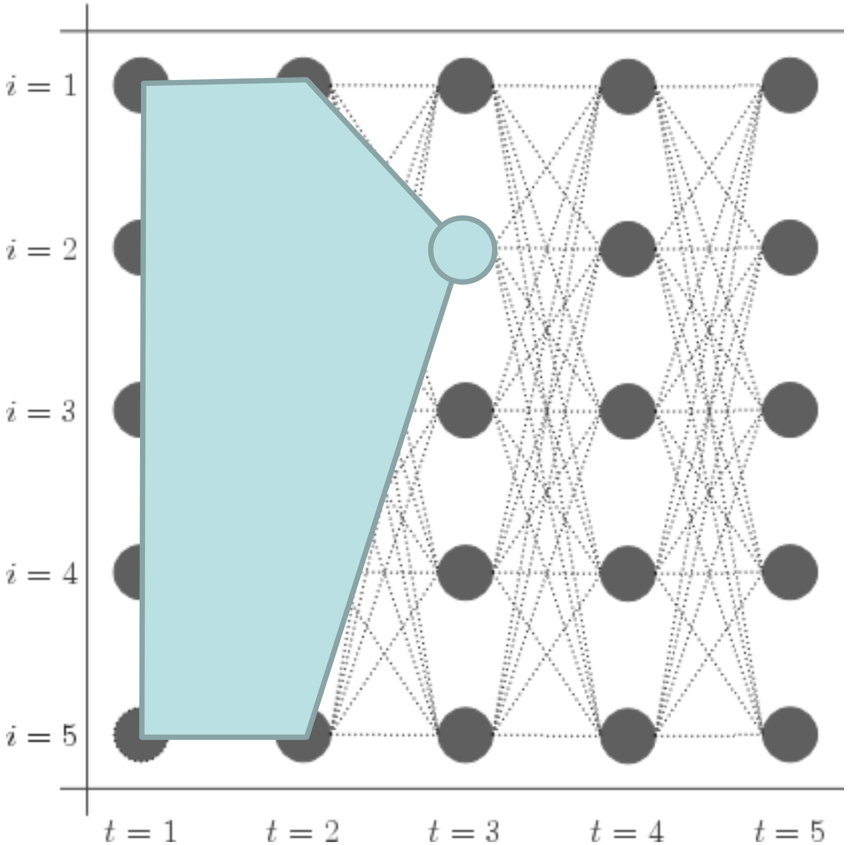
Computing Marginals



$$P(s_t|x) = \frac{P(s_t, x)}{P(x)}$$

$$= \frac{\text{sum of all paths through s at t}}{\text{sum of all paths}}$$

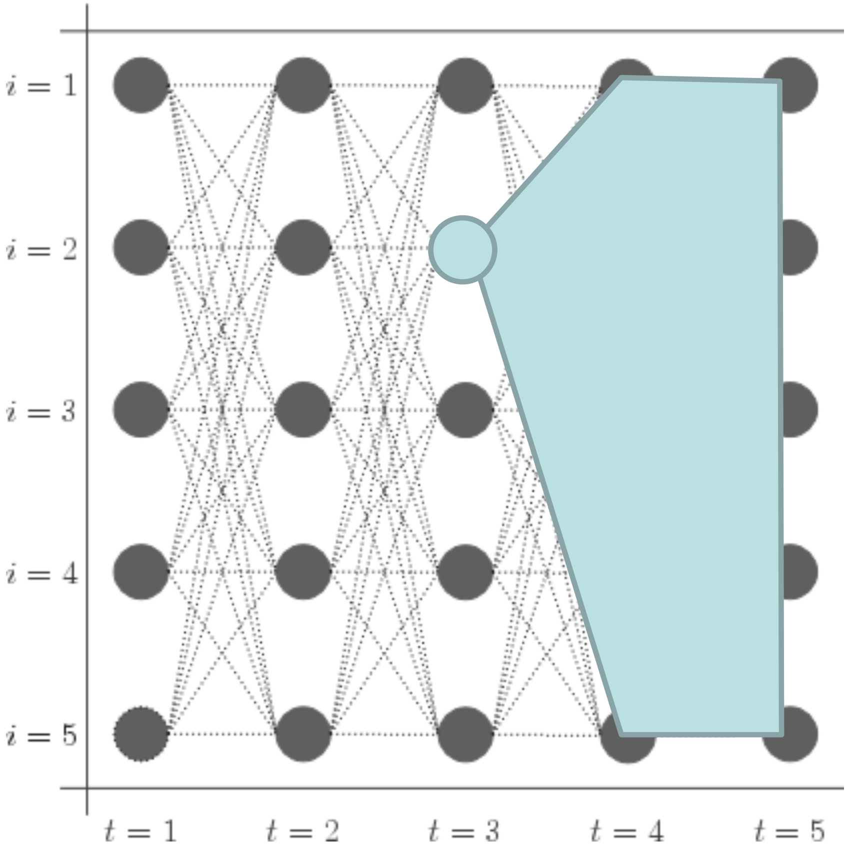
Forward Scores



$$v_t(s_t) = \max_{s_{t-1}} v_{t-1}(s_{t-1})\phi_t(s_{t-1}, s_t)$$

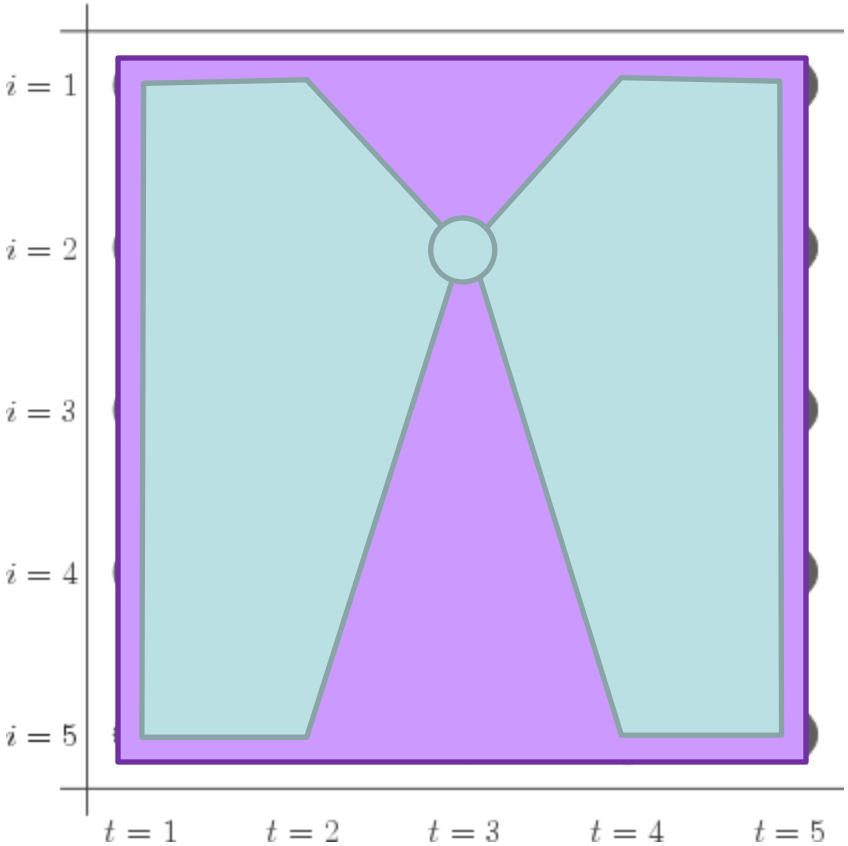
$$\alpha_t(s_t) = \sum_{s_{t-1}} \alpha_{t-1}(s_{t-1})\phi_t(s_{t-1}, s_t)$$

Backward Scores



$$\beta_t(s_t) = \sum_{s_{t+1}} \beta_{t+1}(s_{t+1})\phi_t(s_t, s_{t+1})$$

Total Scores



$$P(s_t, x) = \alpha_t(s_t)\beta_t(s_t)$$

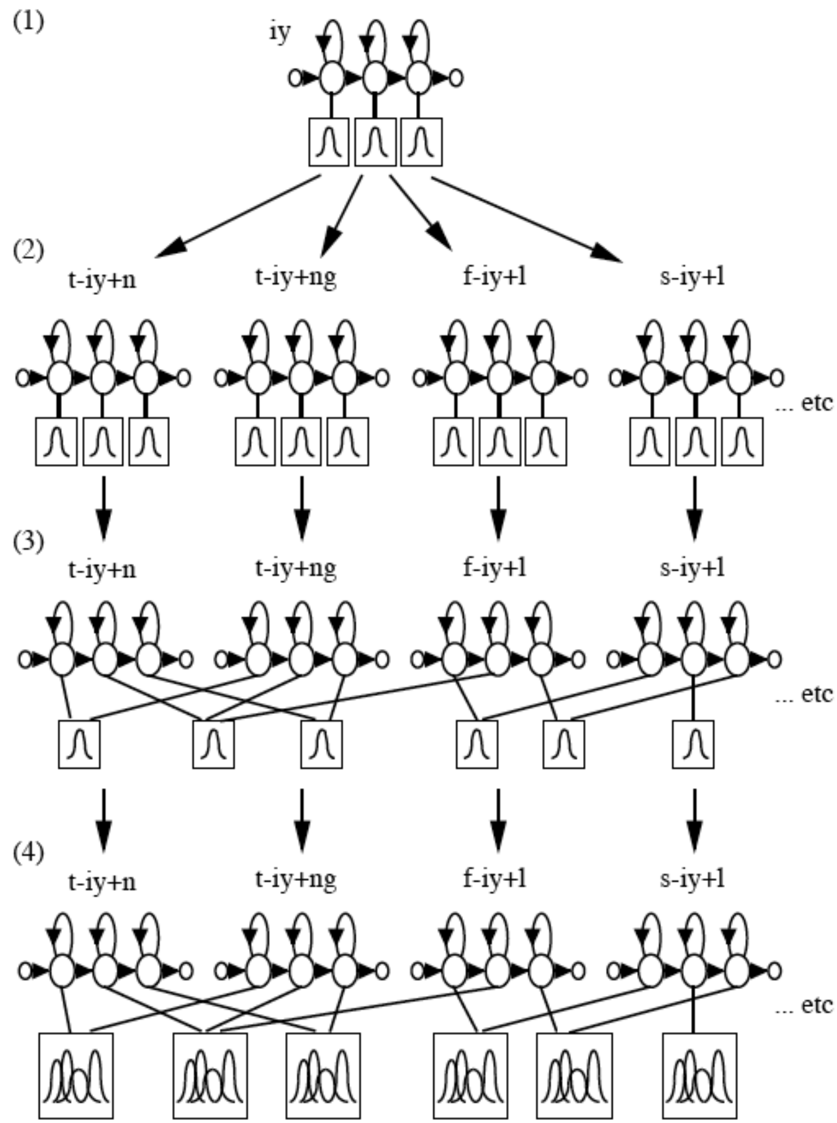
$$\begin{aligned} P(x) &= \sum_{s_t} \alpha_t(s_t)\beta_t(s_t) \\ &= \alpha_T(\text{stop}) \\ &= \beta_0(\text{start}) \end{aligned}$$

Fractional Counts

- Computing fractional (expected) counts
  - Compute forward / backward probabilities
  - For each position, compute marginal posteriors
  - Accumulate expectations
  - Re-estimate parameters (e.g. means, variances, self-loop probabilities) from ratios of these expected counts

Staged Training and State Tying

- Creating CD phones:
  - Start with monophone, do EM training
  - Clone Gaussians into triphones
  - Build decision tree and cluster Gaussians
  - Clone and train mixtures (GMMs)
- General idea:
  - Introduce complexity gradually
  - Interleave constraint with flexibility



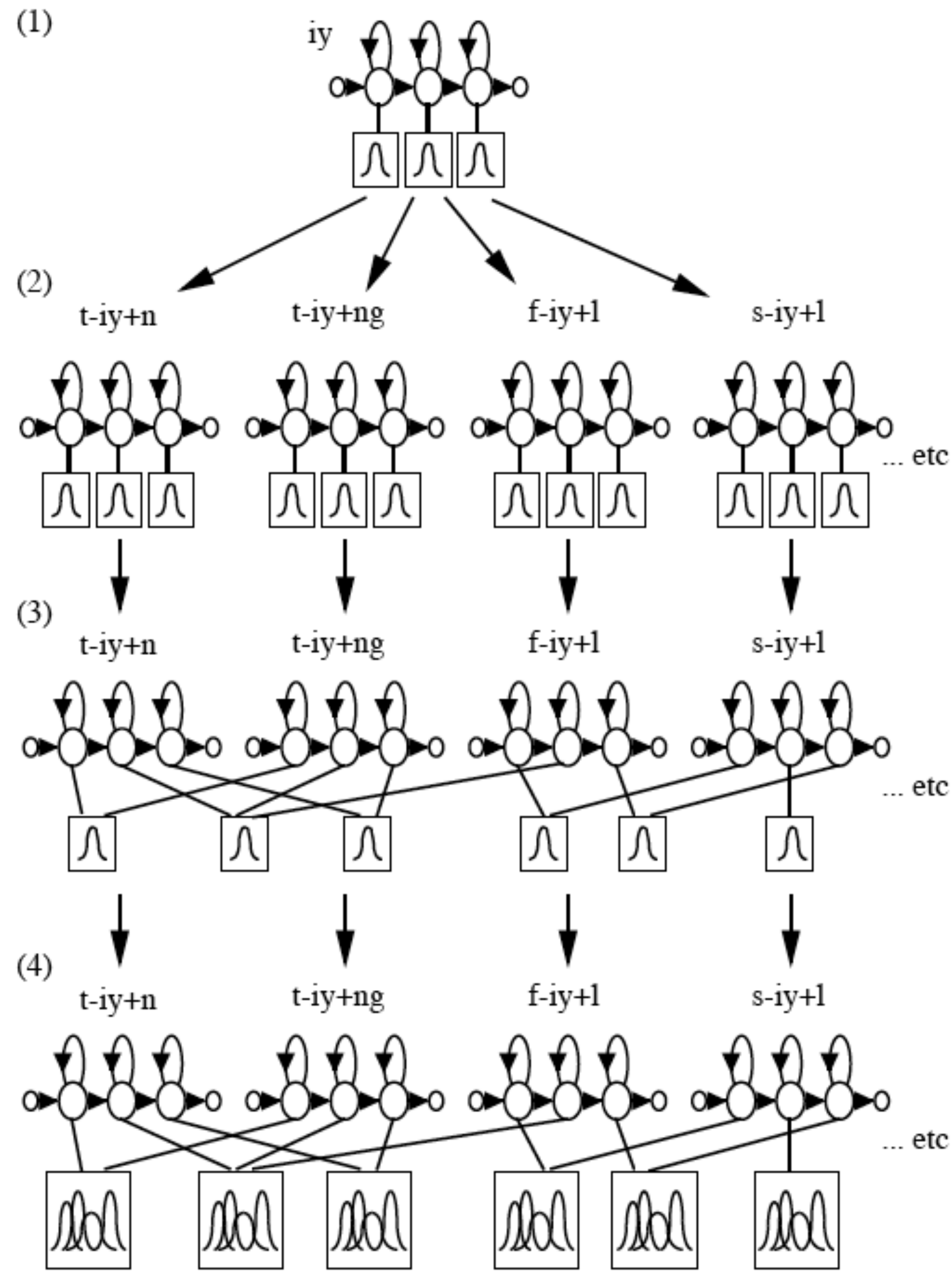
Training Mixture Models

- Input: wav files with unaligned transcriptions
- Forced alignment
  - Computing the "Viterbi path" over the training data (where the transcription is known) is called "forced alignment"
  - We know which word string to assign to each observation sequence.
  - We just don't know the state sequence.
  - So we constrain the path to go through the correct words (by using a special example-specific language model)
  - And otherwise run the Viterbi algorithm
- Result: aligned state sequence

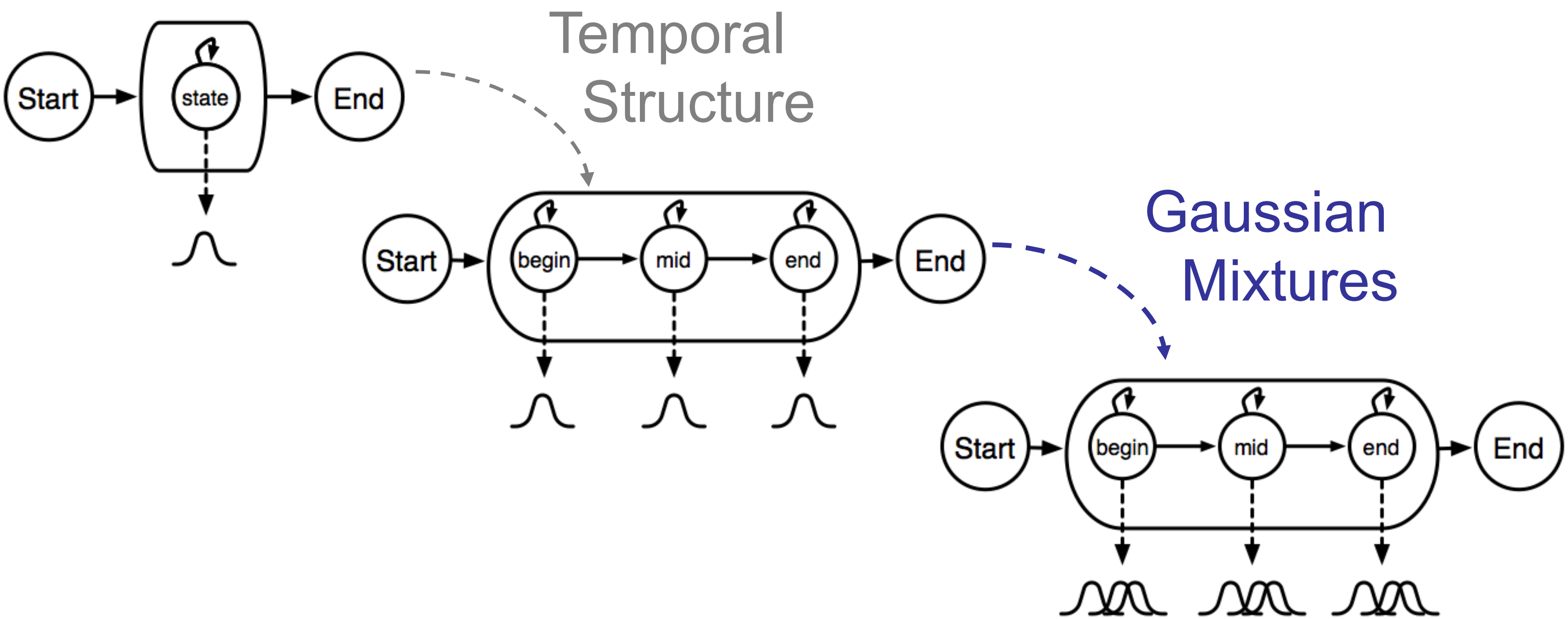
State Tying

- Creating CD phones:
  - Start with monophone, do EM training
  - Clone Gaussians into triphones
  - Build decision tree and cluster Gaussians
  - Clone and train mixtures (GMMs)

- General idea:
  - Introduce complexity gradually
  - Interleave constraint with flexibility



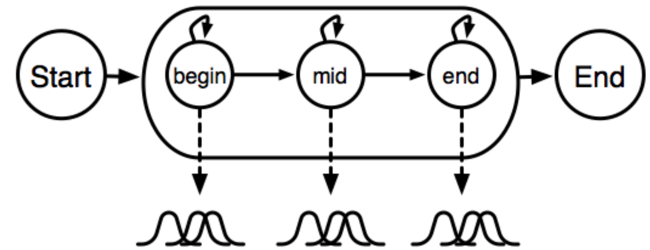
Standard subphone/mixture HMM



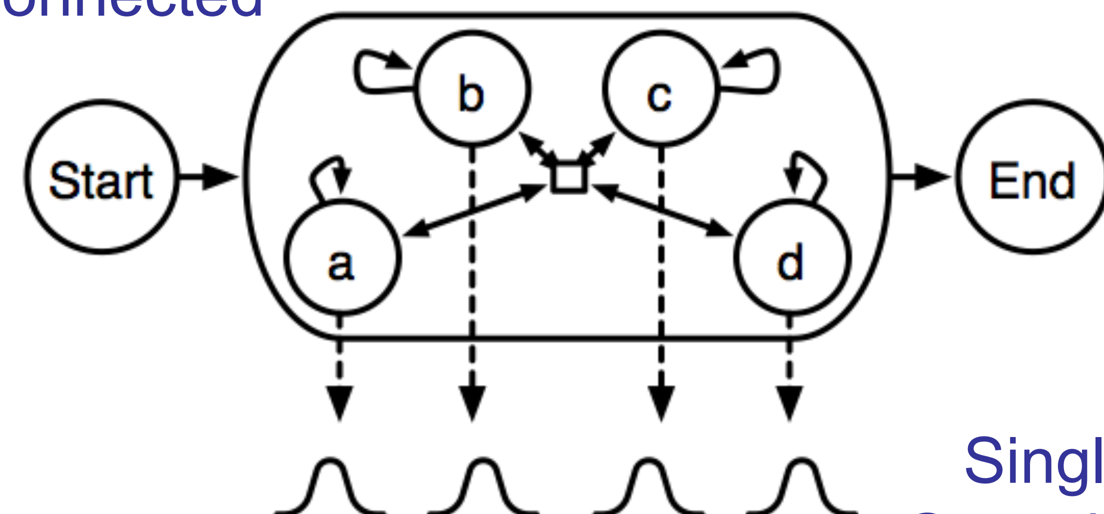
Model	Error rate
HMM Baseline	**25.1%**

An Induced Model

Standard Model



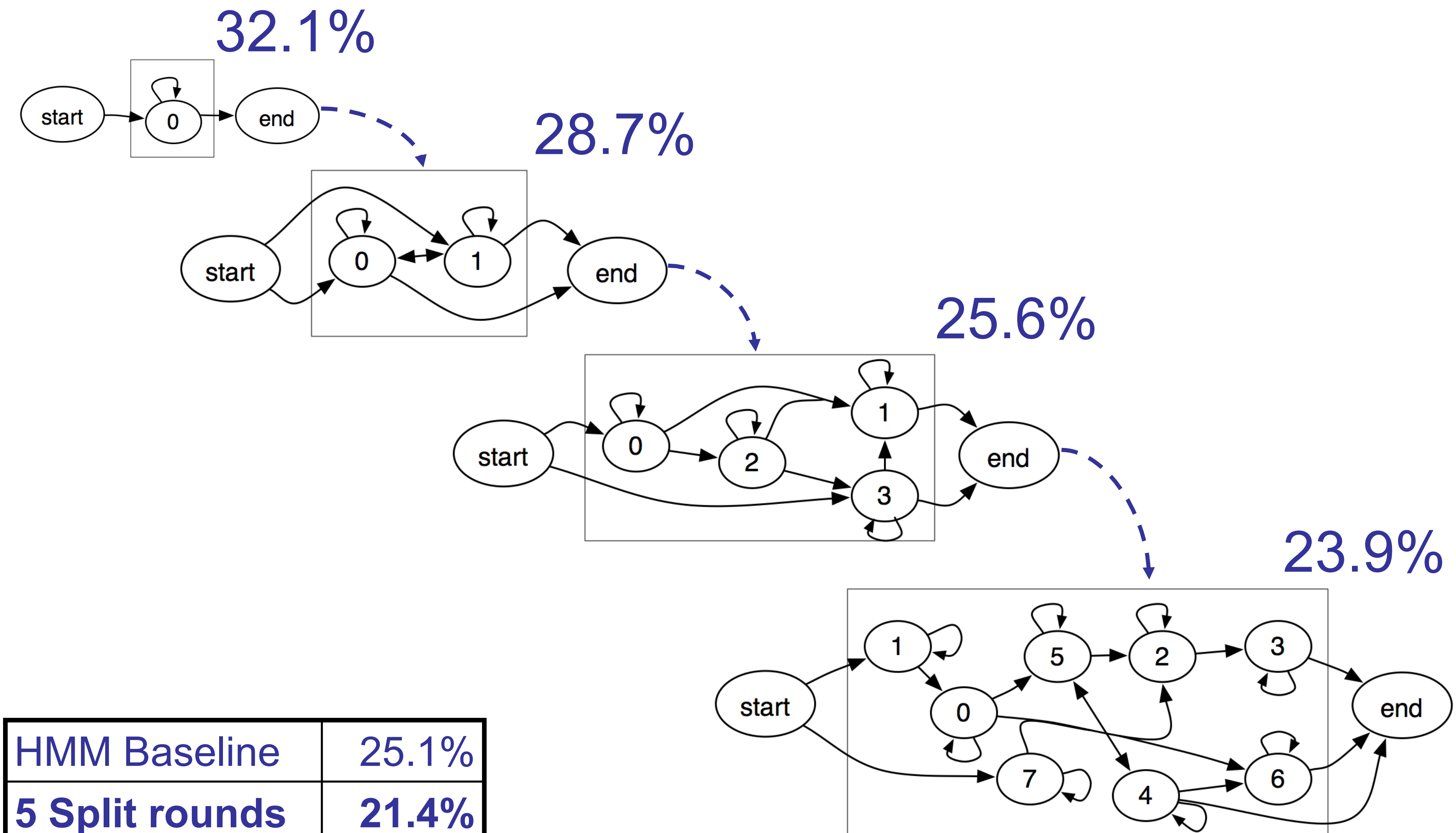
Fully Connected



Single Gaussians

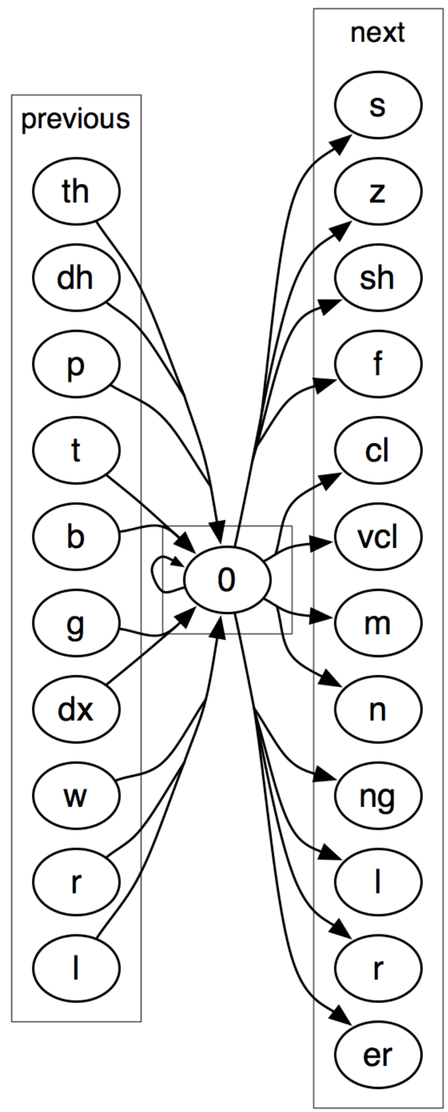
[Petrov, Pauls, and Klein, 07]

Hierarchical Split Training with EM

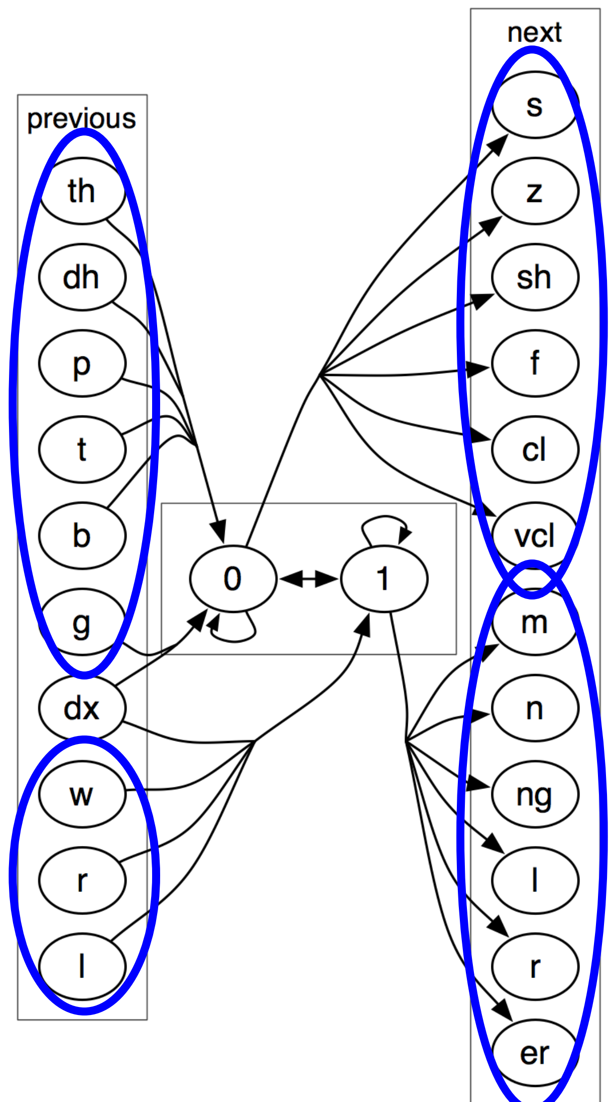


HMM Baseline	25.1%
5 Split rounds	**21.4%**

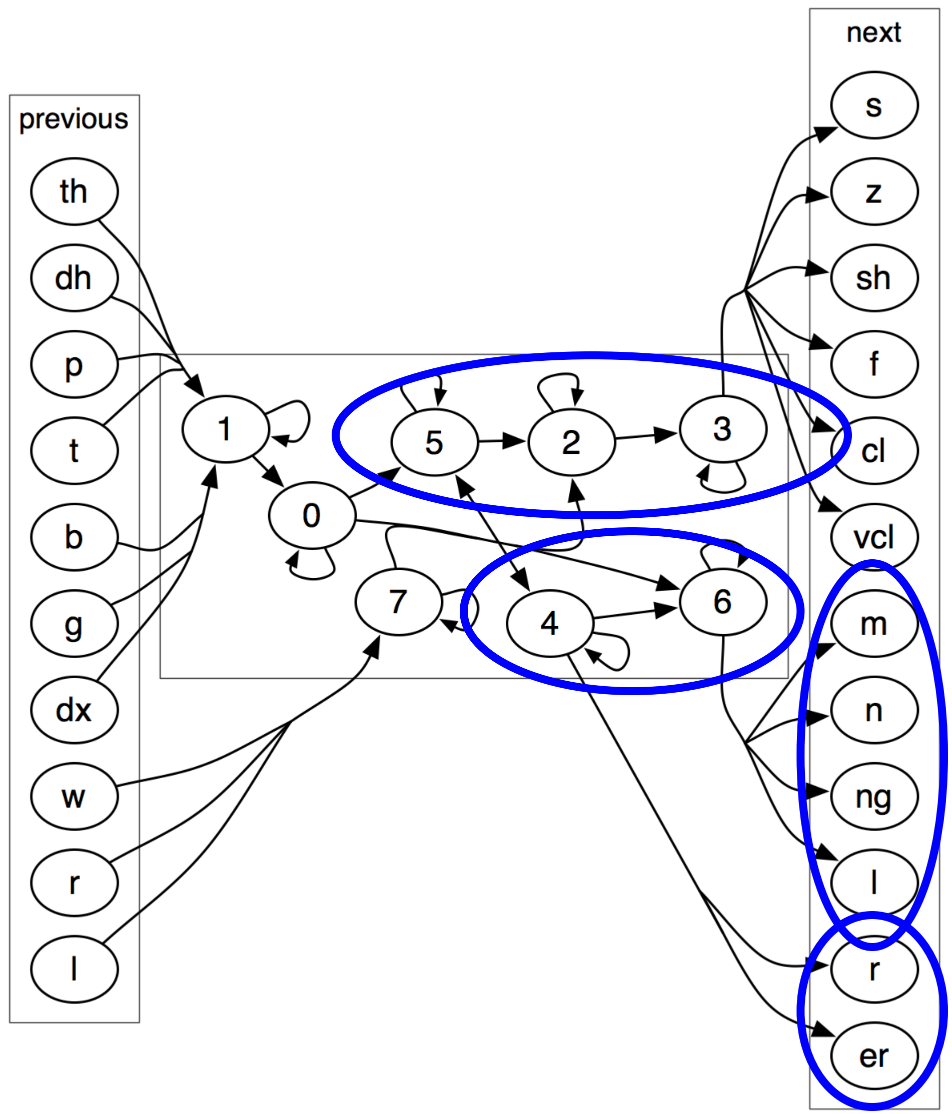
Refinement of the /ih/-phone



Refinement of the /ih/-phone



Refinement of the /ih/-phone



HMM states per phone