Algorithms for NLP

Speech Inference

Taylor Berg-Kirkpatrick – CMU

Slides: Dan Klein – UC Berkeley

Project Announcements

- Due date postponed: now due Tuesday 9/27 at 11:59pm
- Will be using blackboard 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);`

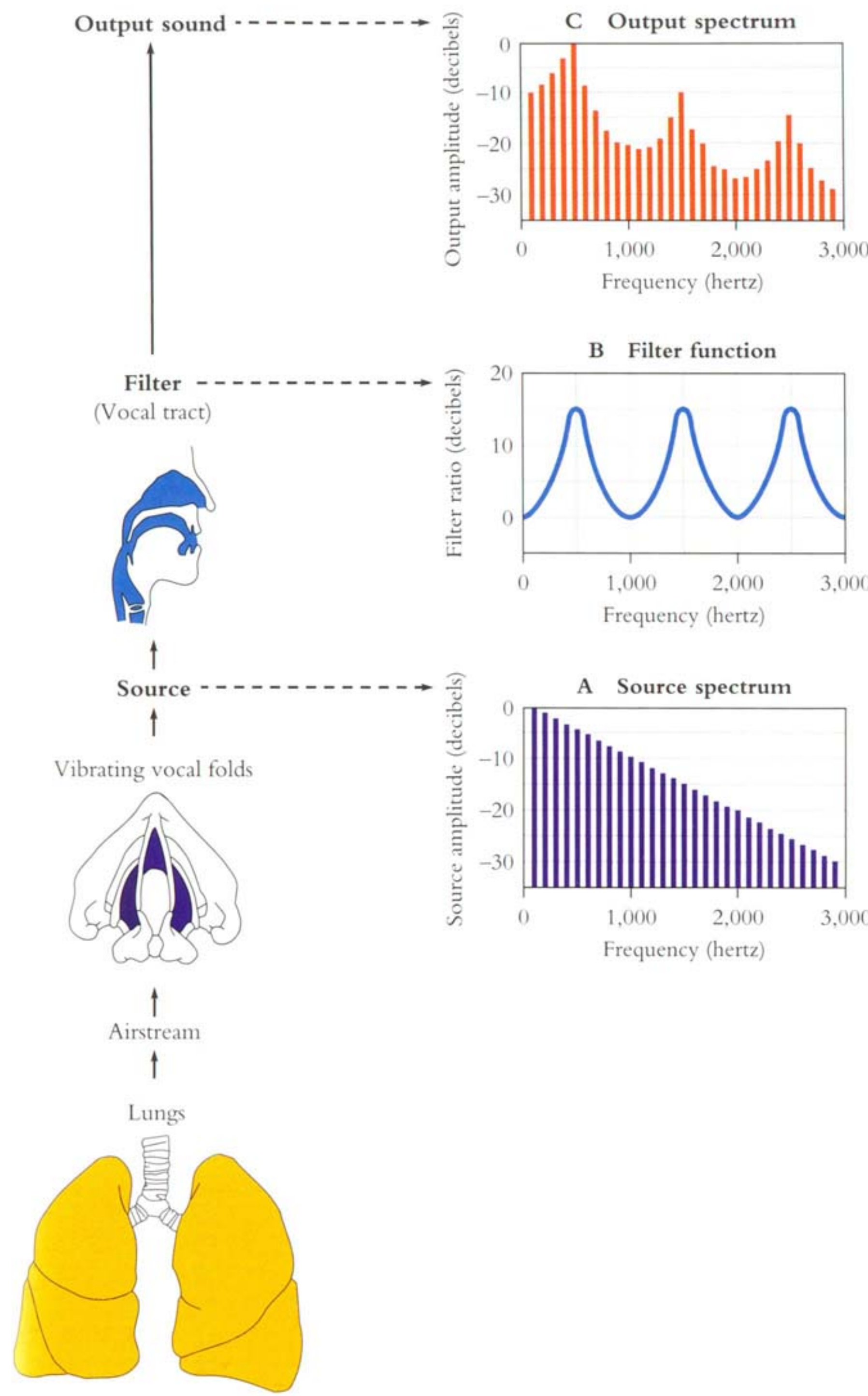
Project Grading

- 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

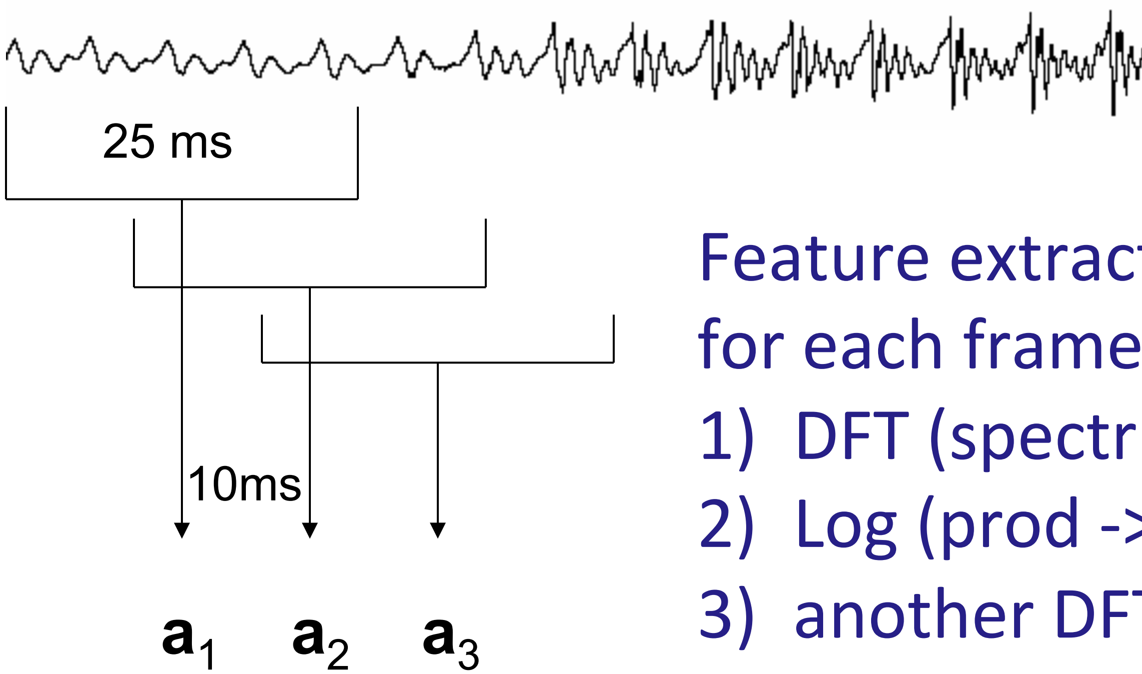
Why these Peaks?

- 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



Feature Extraction

- A frame (25 ms wide) extracted every 10 ms

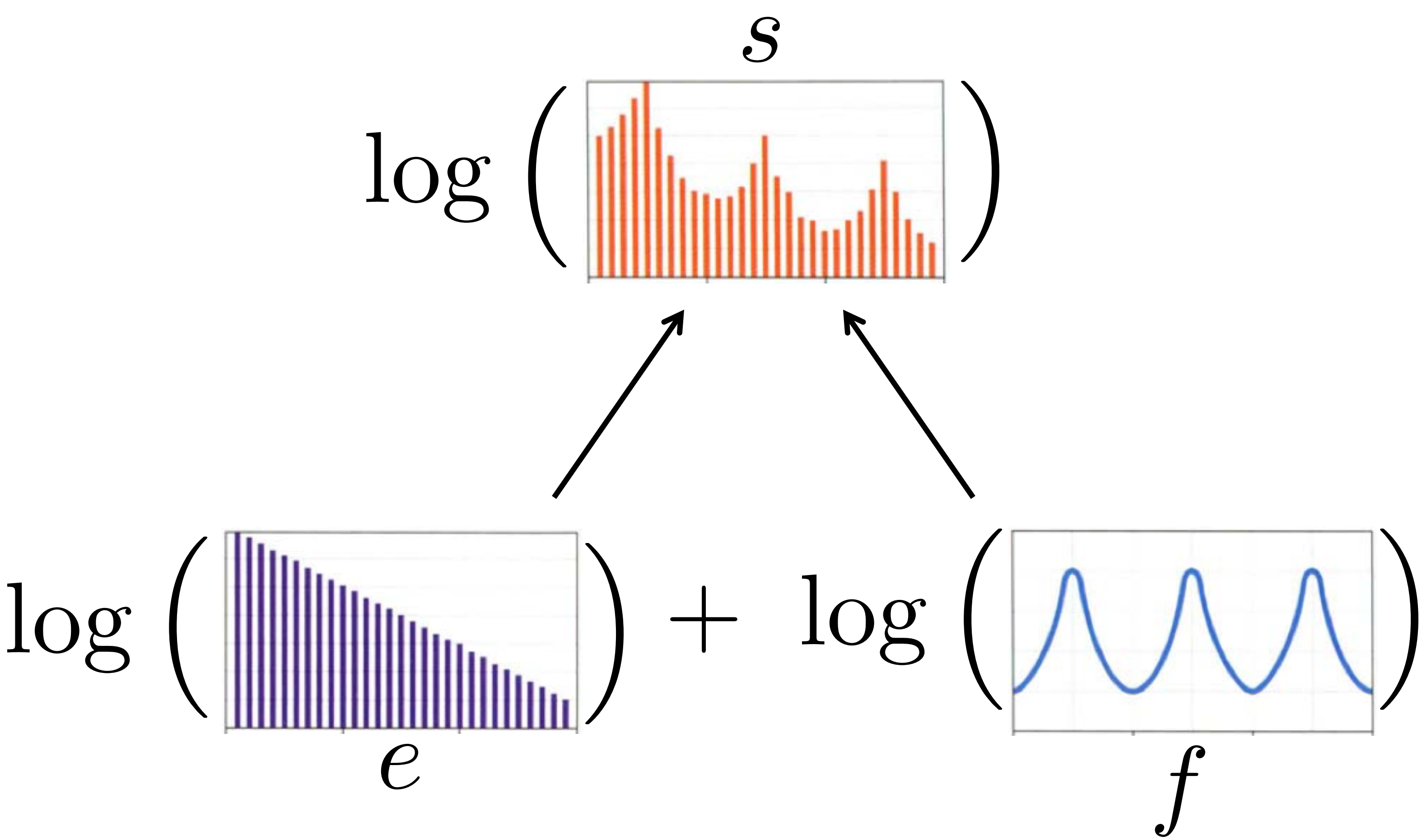


Feature extraction for each frame:

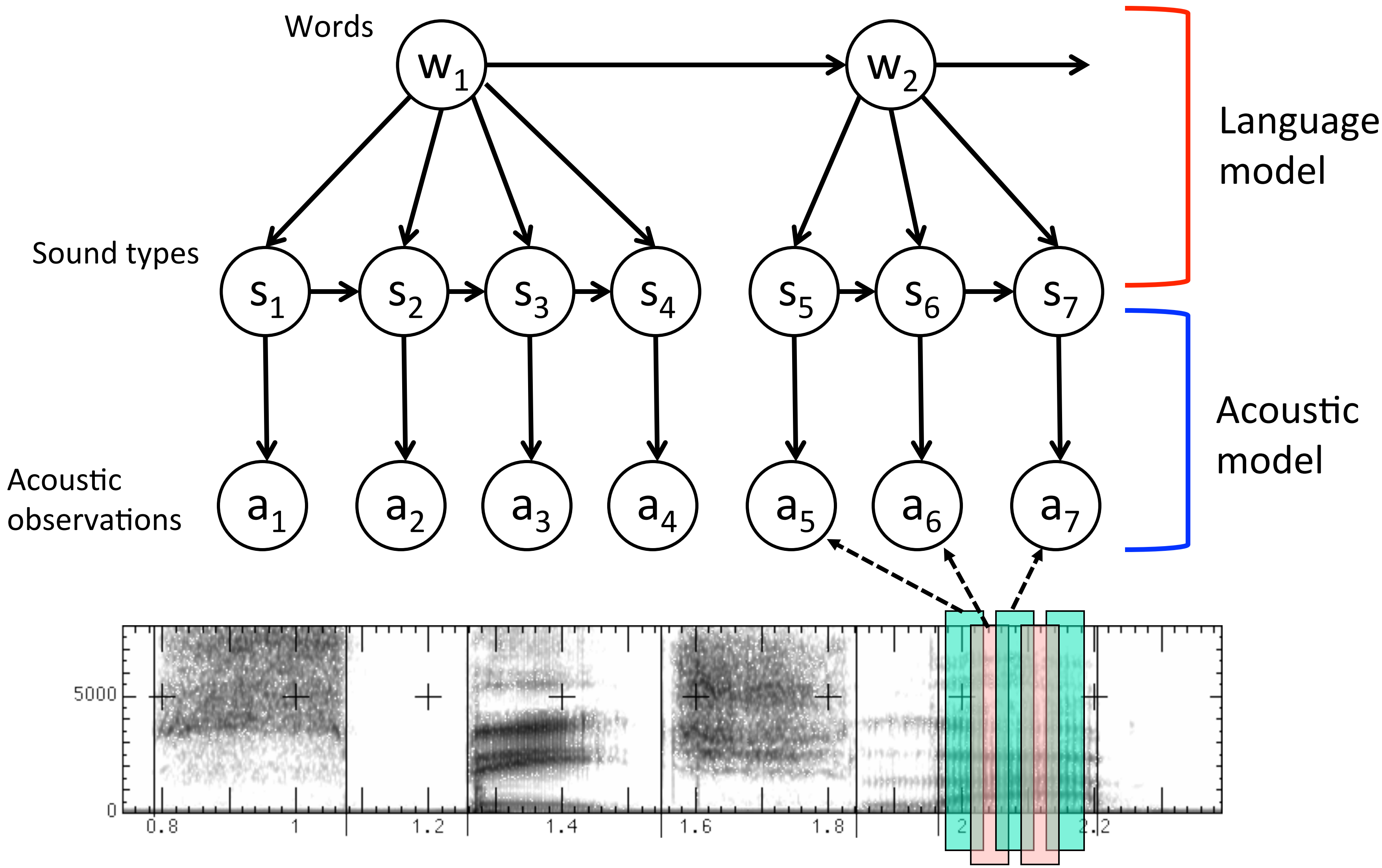
1) DFT (spectrum)
2) Log (prod -> sum)
3) another DFT (lowpass)

Figure: Simon Arnfield

Deconvolution / Liftering

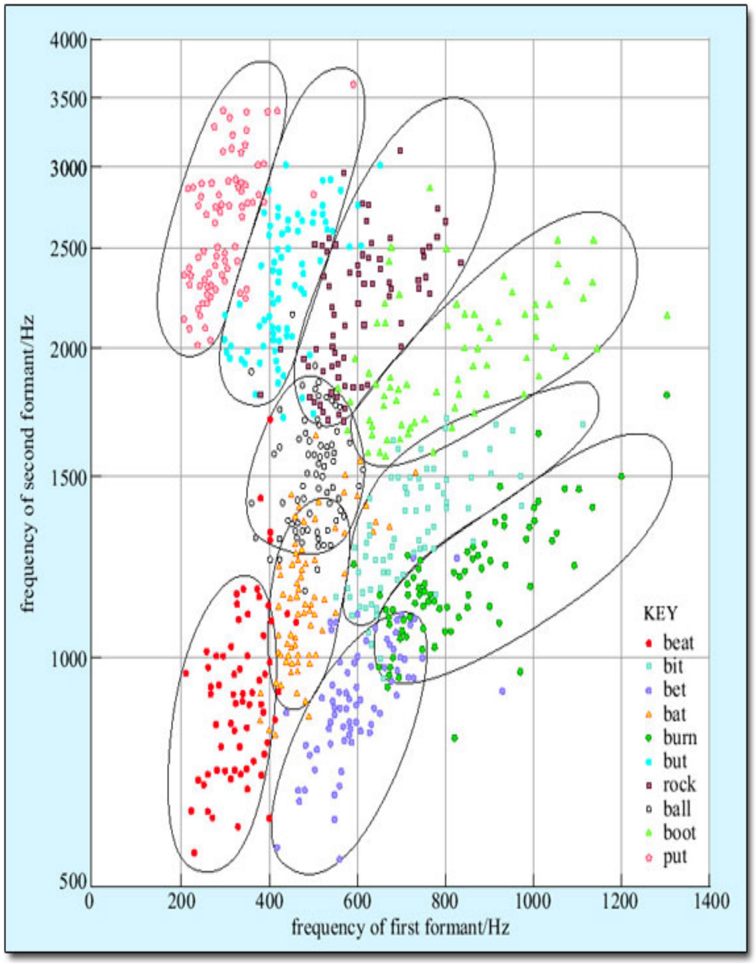


Speech Model



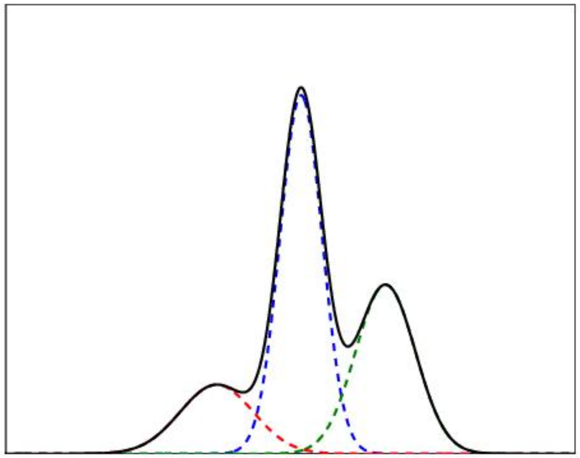
HMMs for Continuous Observations

- 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 progressive
  - Context independent subphone (~3 per phone)
  - Context dependent phone (triphones)
  - State tying of CD phone

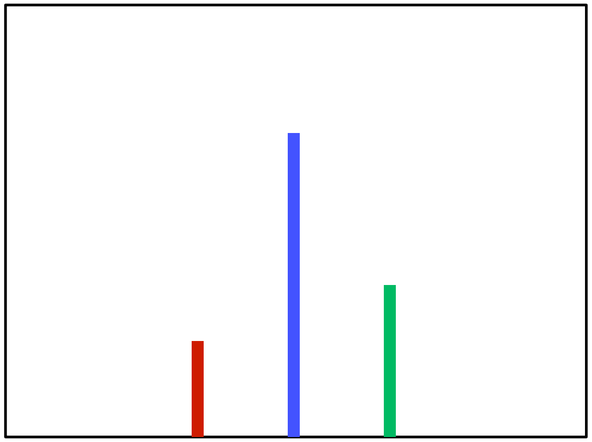


GMMs

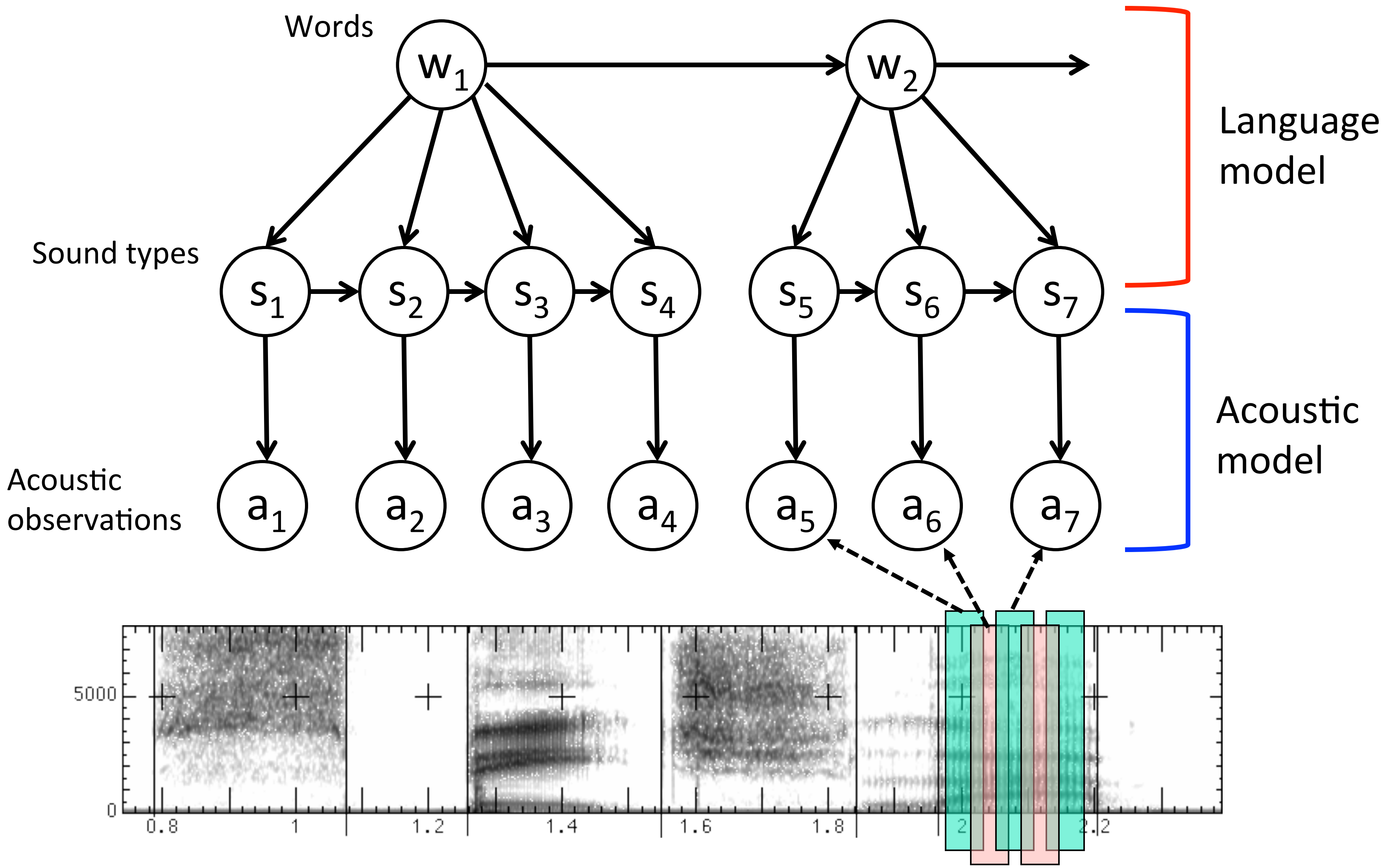
- Summary: each state has an emission distribution P(x|s) (likelihood function) parameterized by:
  - M mixture weights
  - M mean vectors of dimensionality D
  - Either M covariance matrices of DxD or M Dx1 diagonal variance vectors



- 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, cf next week)



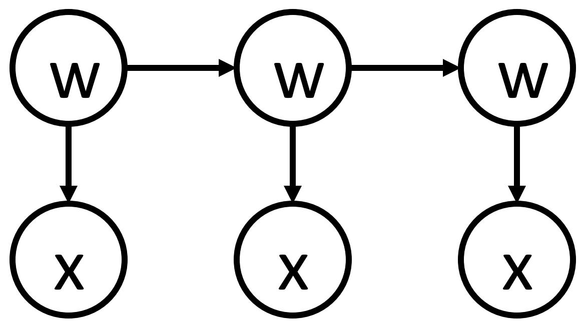
Speech Model



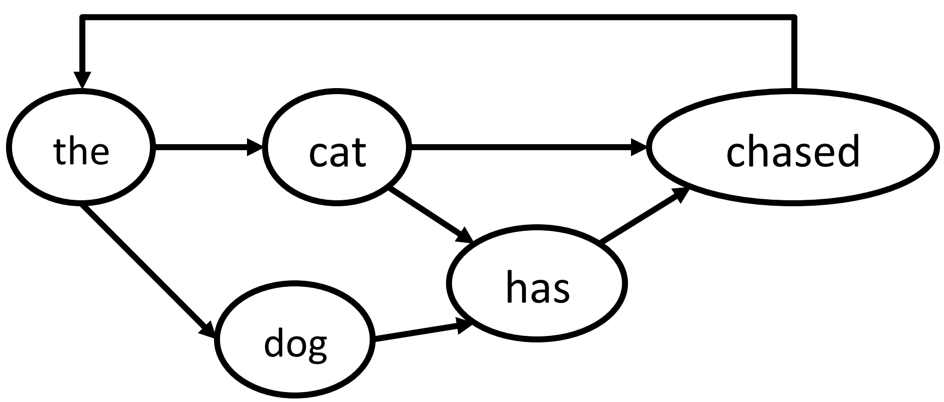
State Model

State Transition Diagrams

- Bayes Net: HMM as a Graphical Model



- State Transition Diagram: Markov Model as a Weighted FSA



ASR Lexicon

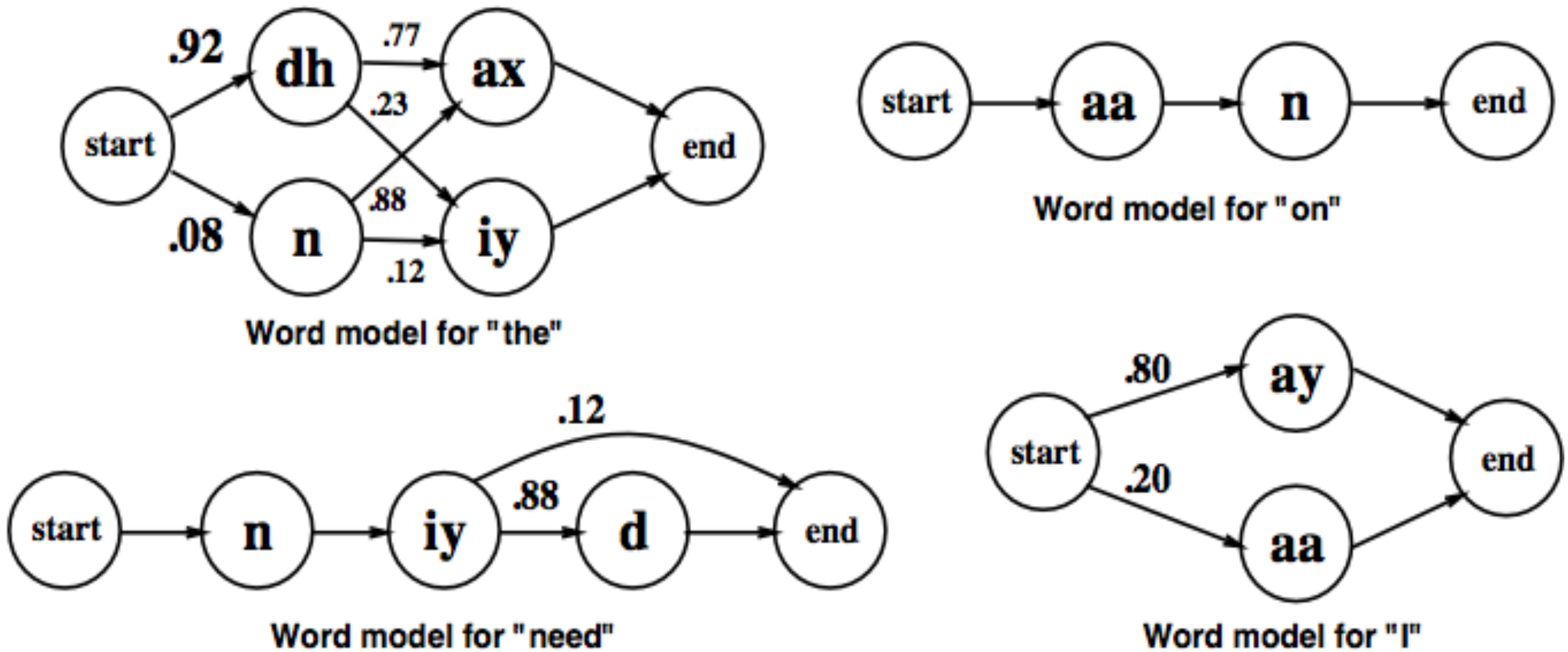


Figure: J & M

Lexical State Structure

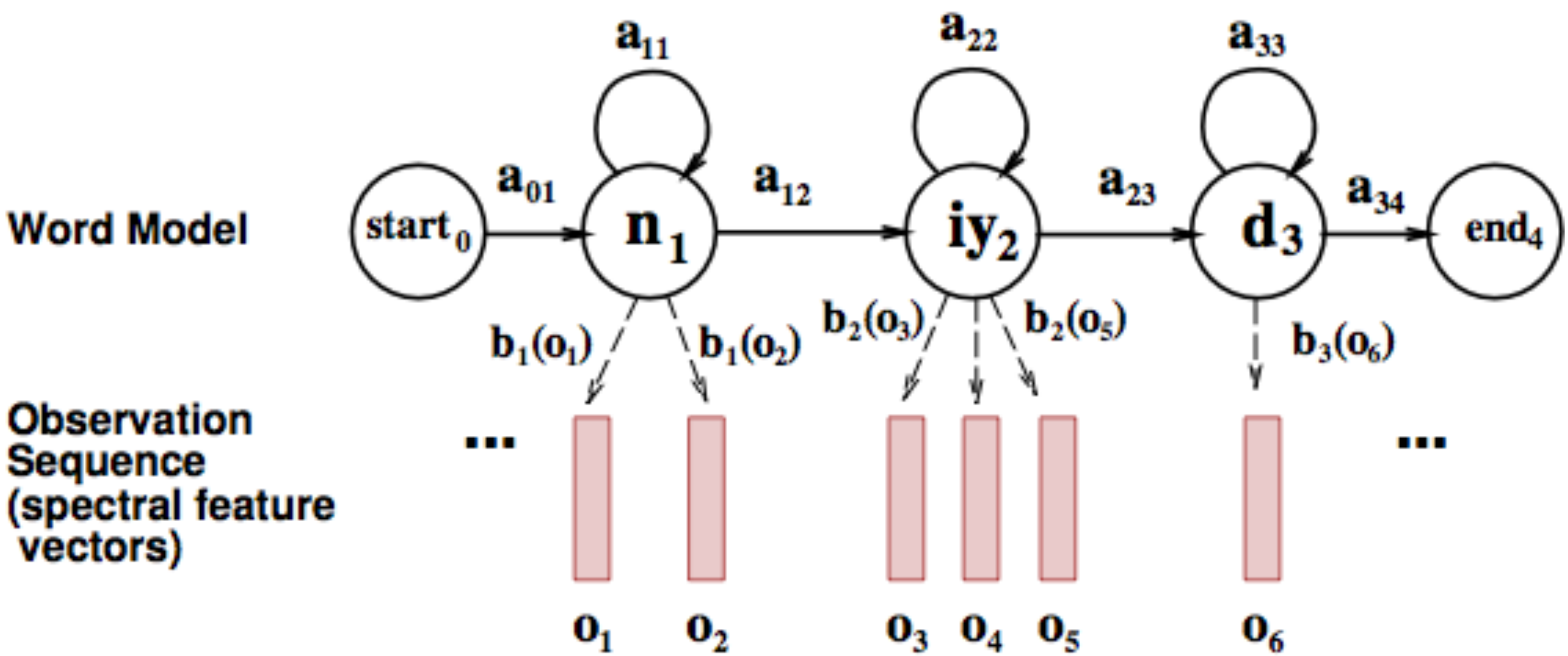


Figure: J & M

Adding an LM

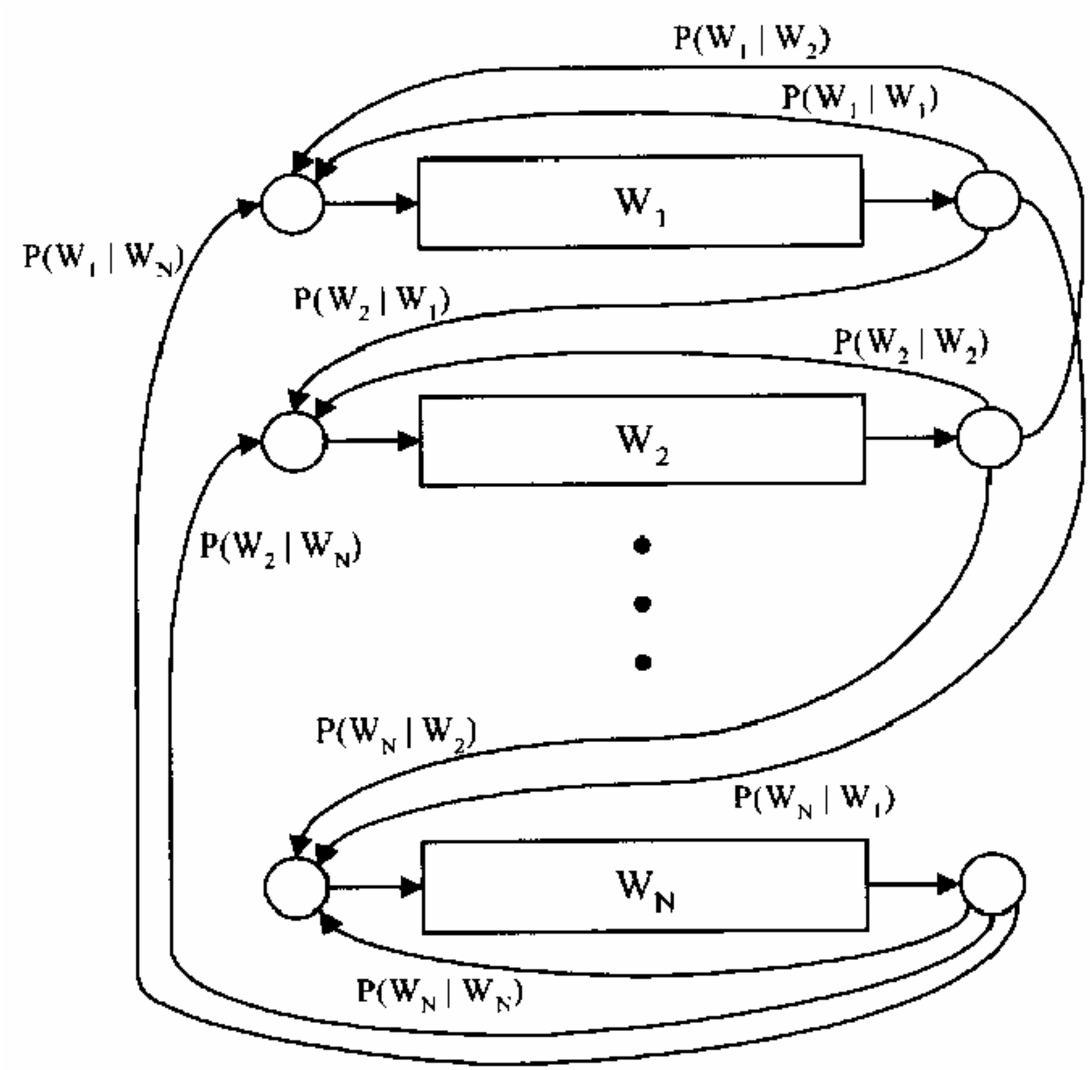


Figure from Huang et al page 618

State Space

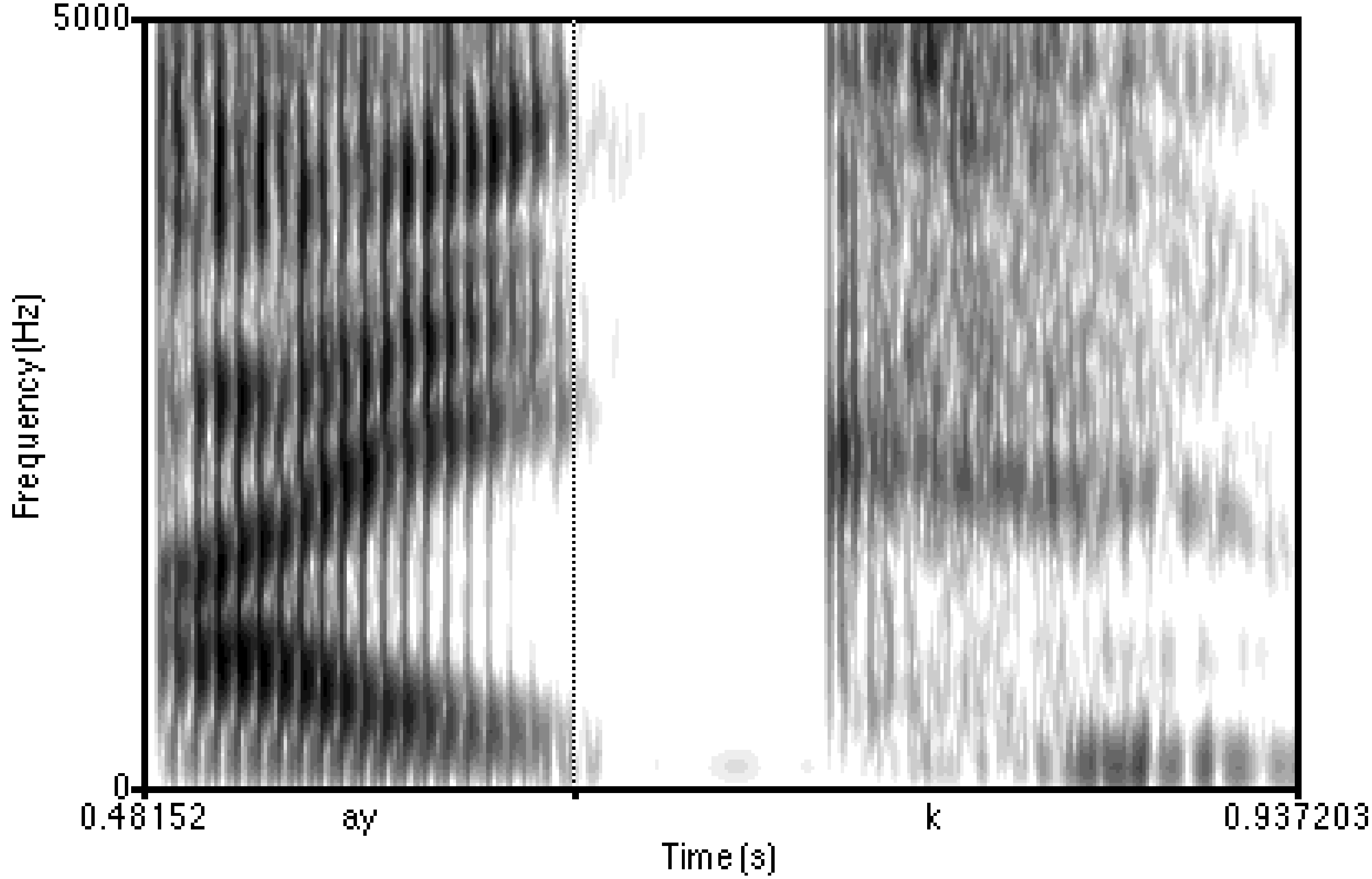
- State space must include
  - Current word (|V| on order of 20K+)
  - Index within current word (|L| on order of 5)
  - E.g. (lec[t]ure) (though not in orthography!)

- 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

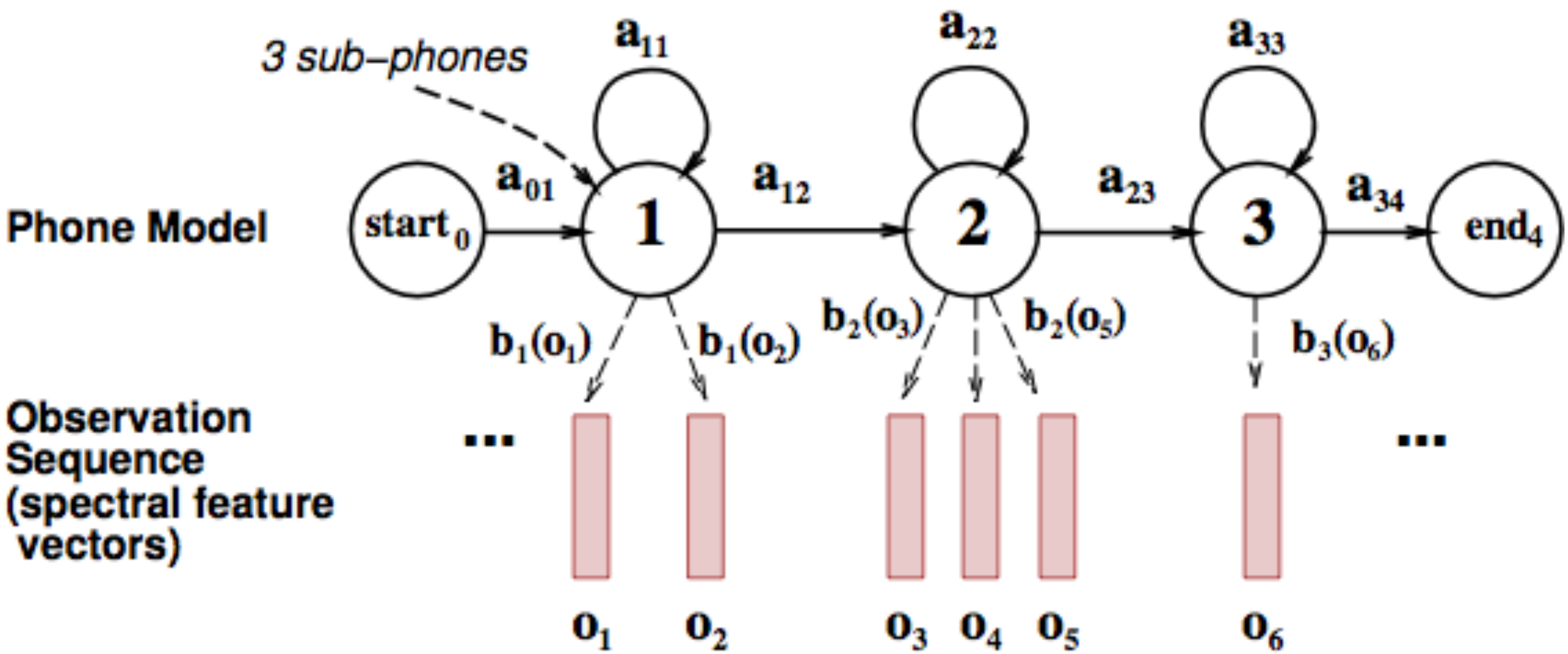


Figure: J & M

A Word with Subphones

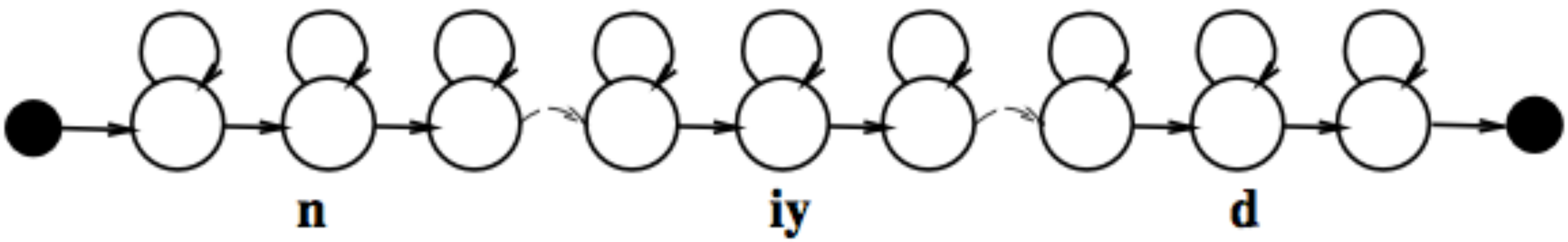
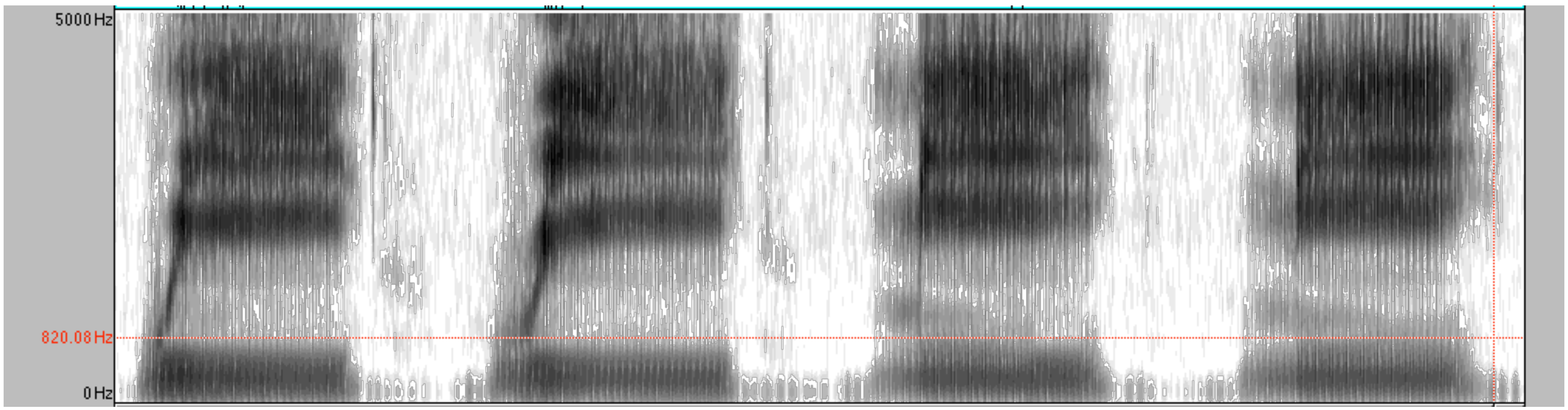


Figure: J & M

Modeling phonetic context



w iy   r iy   m iy   n iy

“Need” with triphone models

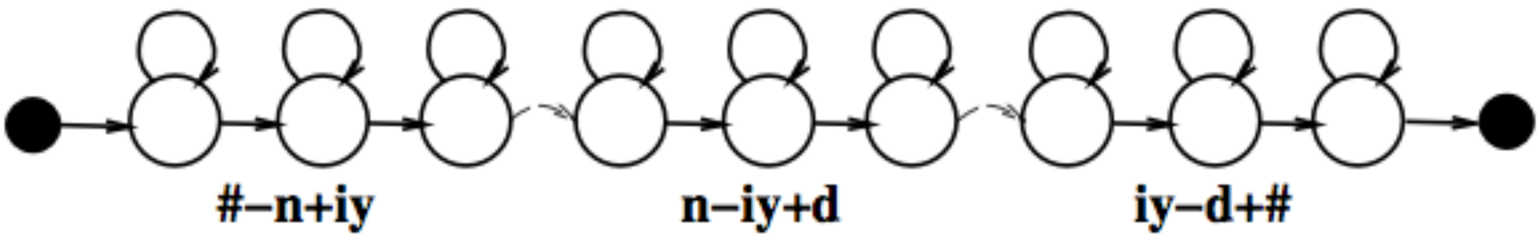


Figure: J & M

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

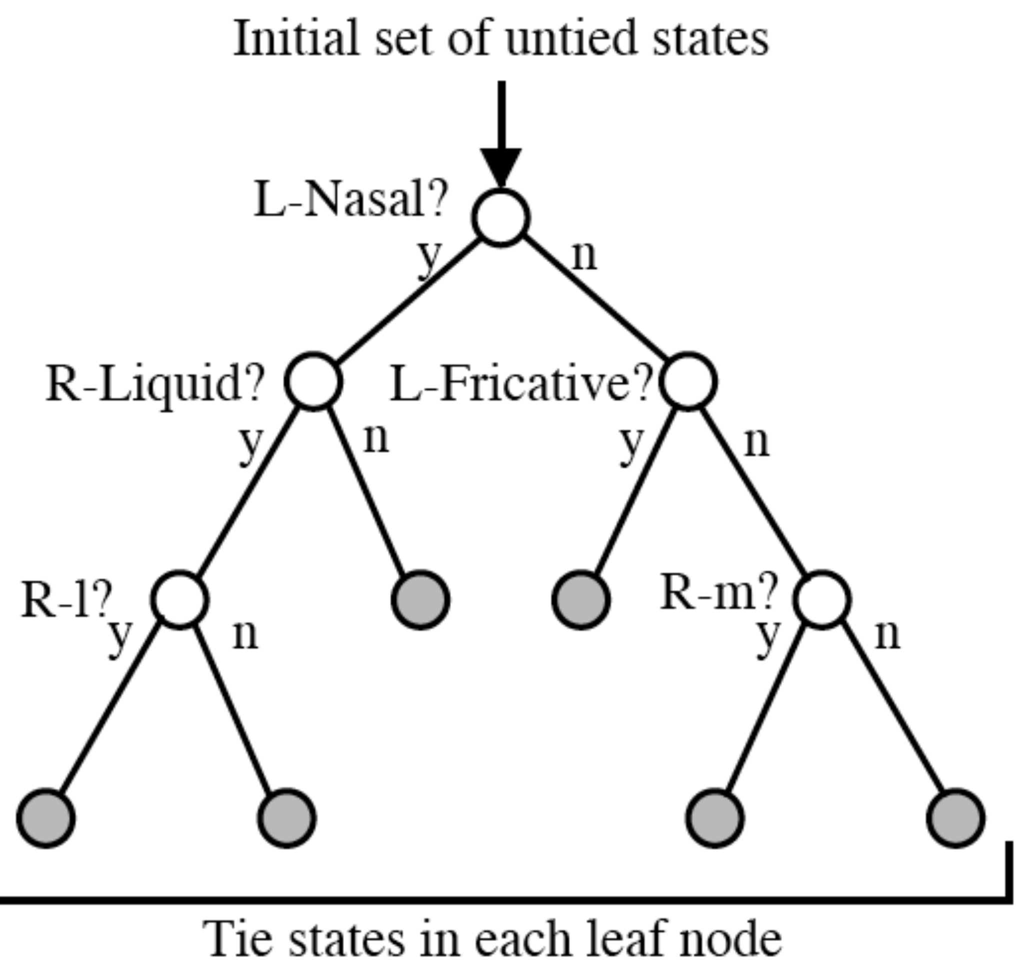


Figure: J & M

FSA for Lexicon + Bigram LM

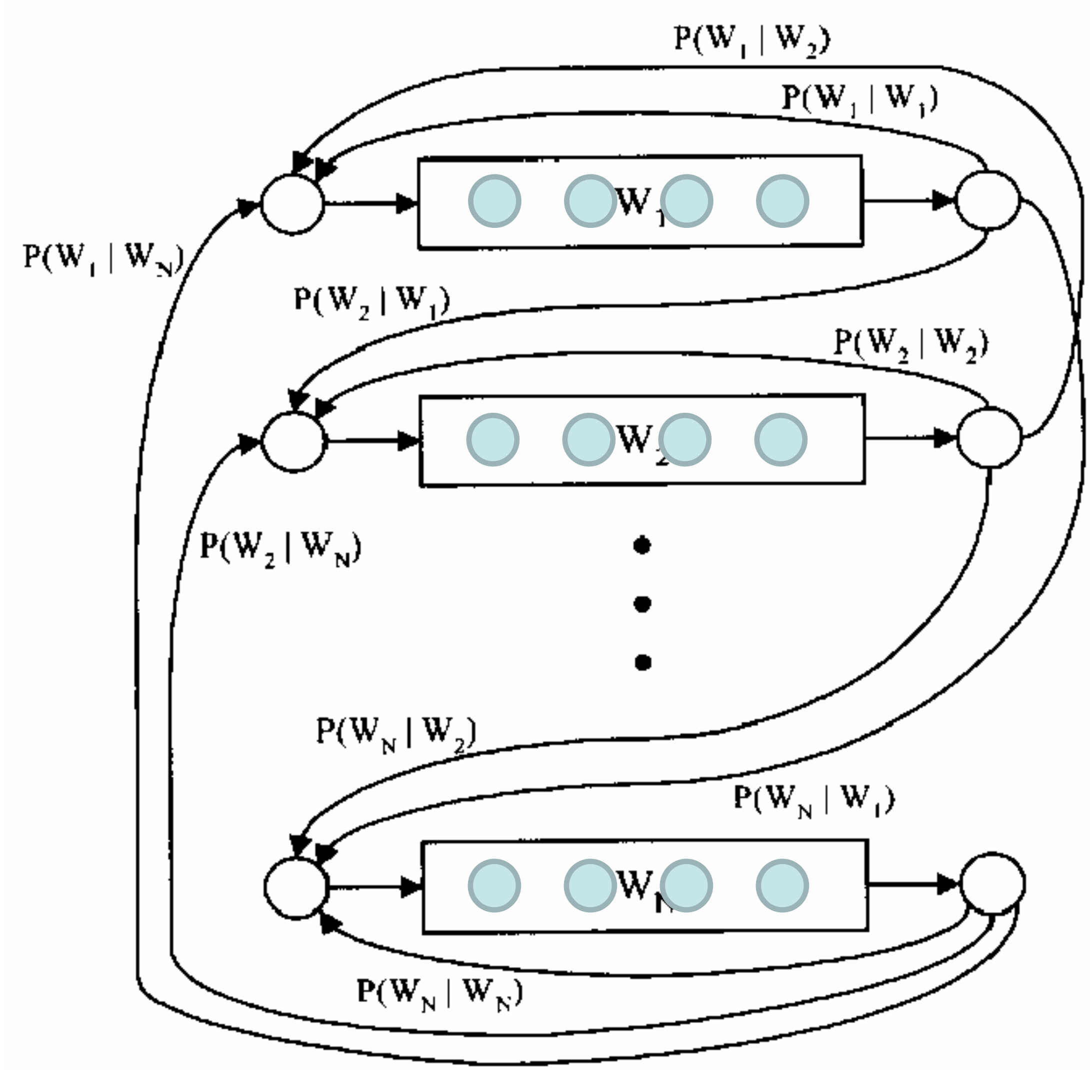


Figure from Huang et al page 618

State Space

- Full state space

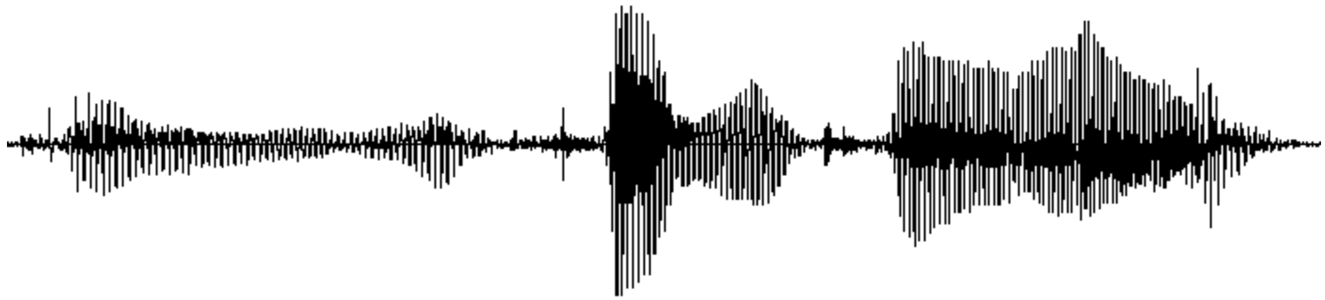
(LM context, lexicon index, subphone)

- Details:
  - LM context is the past n-1 words
  - Lexicon index is a phone position within a word (or a trie of the lexicon)
  - Subphone is begin, middle, or end
  - E.g. (after the, lec[t-mid]ure)

- Acoustic model depends on clustered phone context
  - But this doesn't grow the state space

Decoding

Inference Tasks



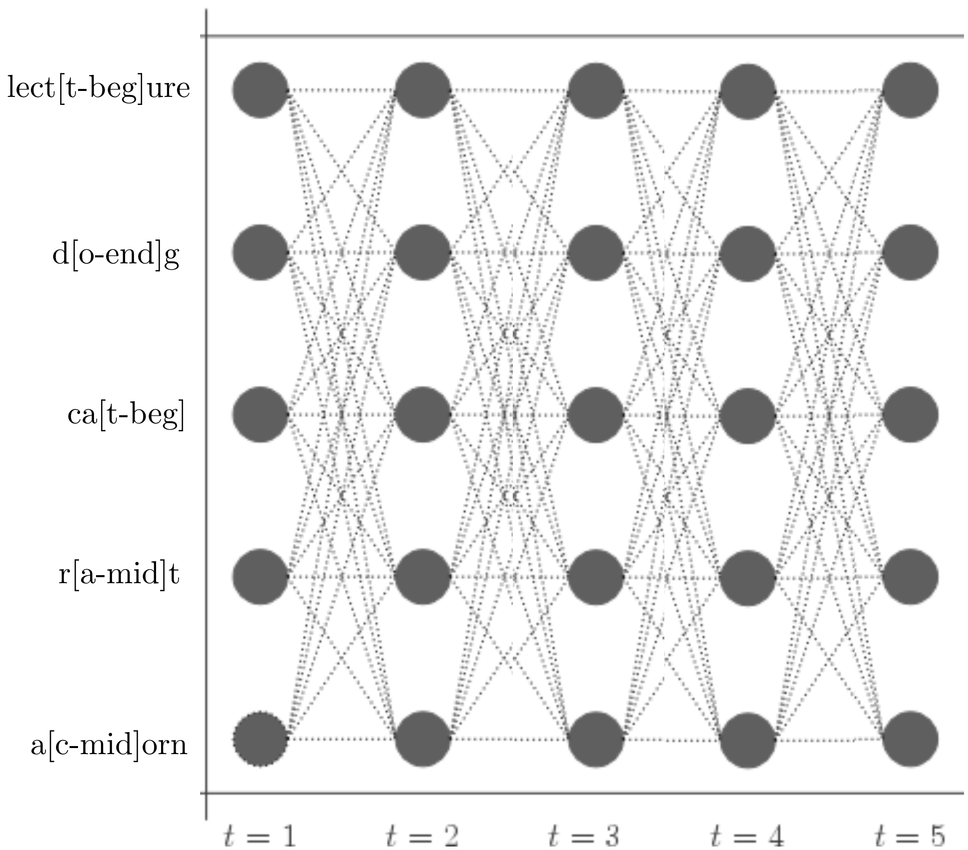
Most likely word sequence:

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

State Trellis



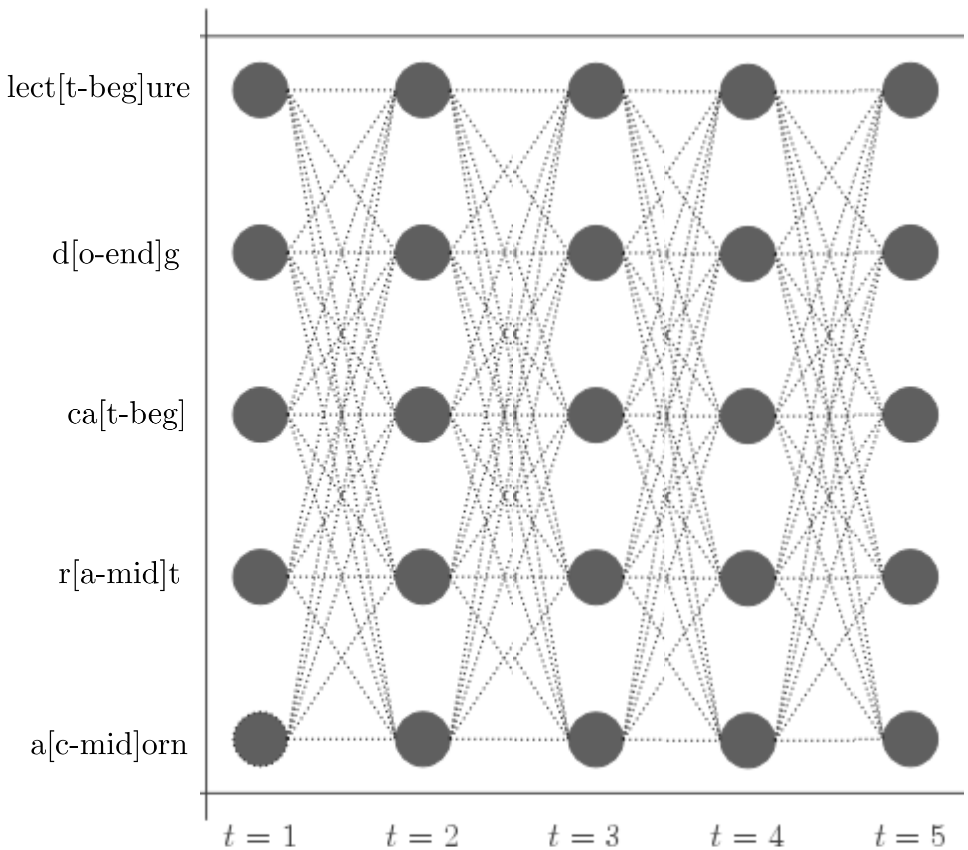
$$\phi_t(s_{t-1}, s_t) = P(a_t|s_t)P(s_t|s_{t-1})$$

$$P(a,s) = \prod_t P(a_t|s_t)P(s_t|s_{t-1})$$

$$= \prod_t \phi_t(s_{i-1}, s_i)$$

Figure: Enrique Benimeli

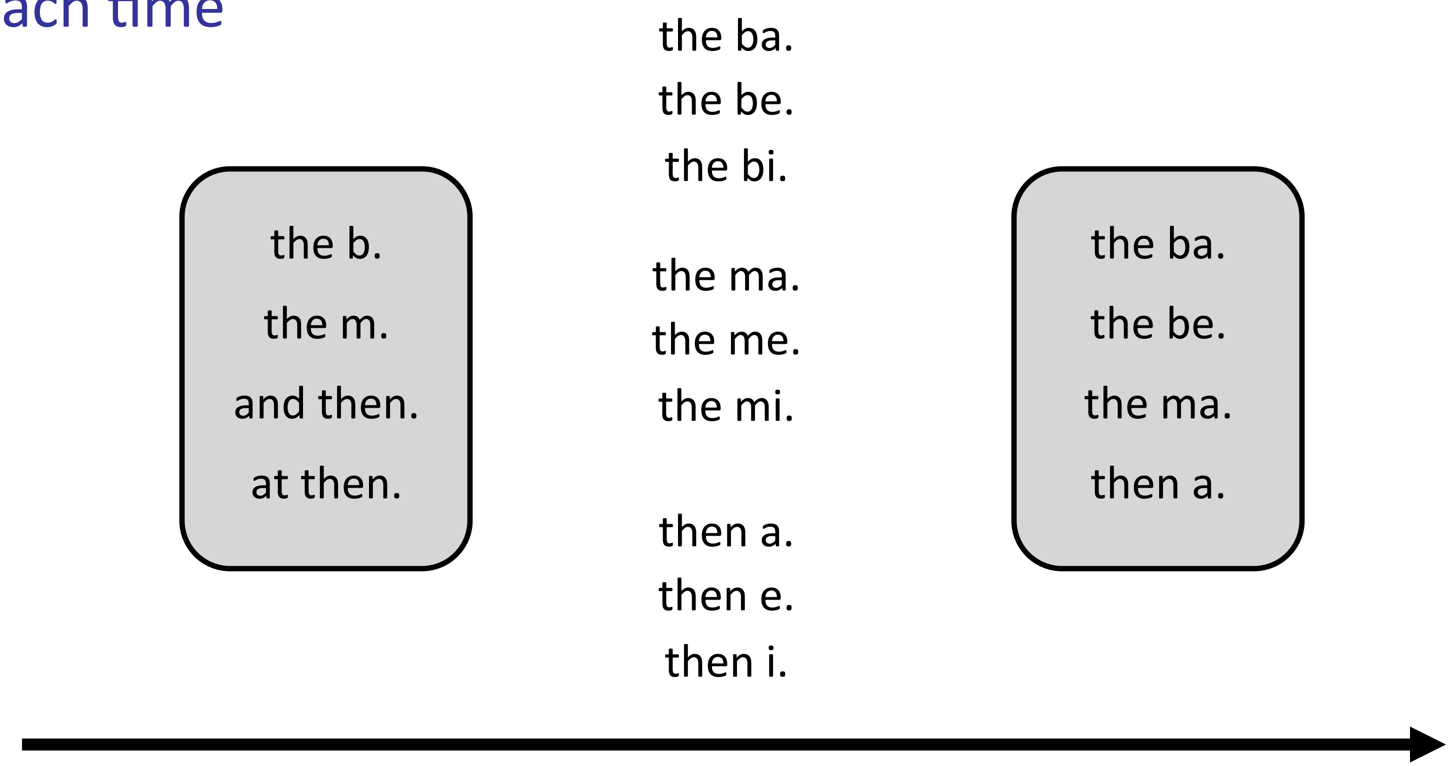
Naïve Viterbi



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

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



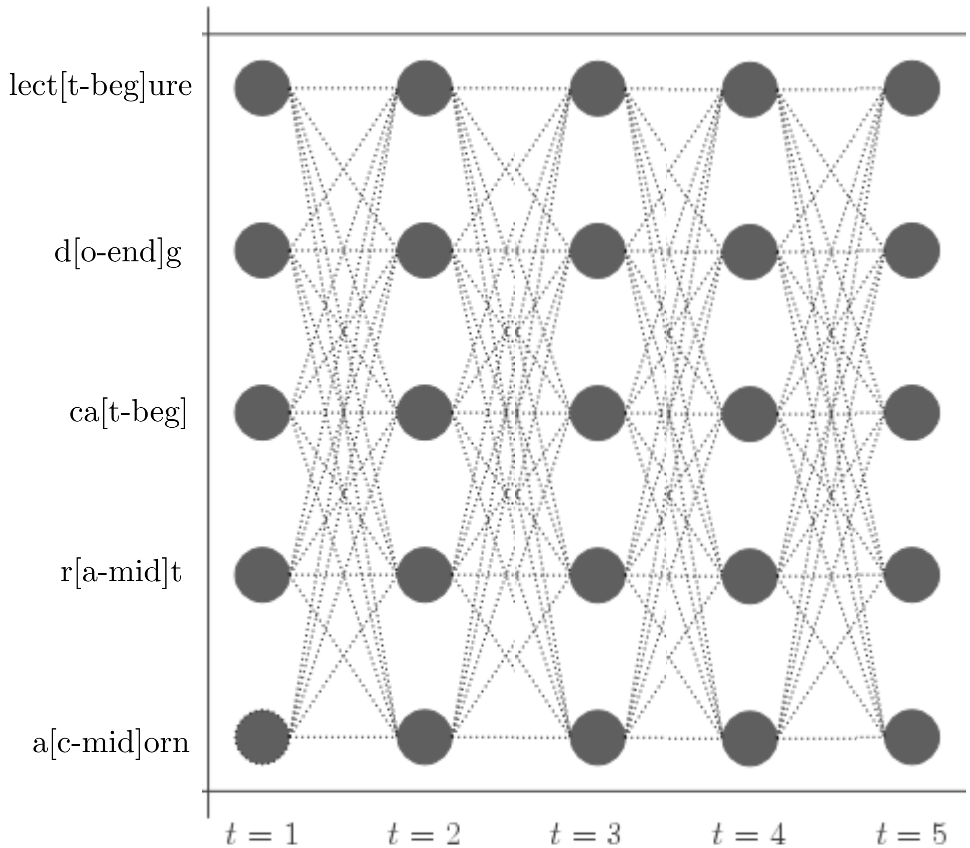
- Important: still dynamic programming; collapse equiv states

Beam Search

- At each time step
  - Start: Beam (collection) v_t of hypotheses s at time t
  - For each s in v_t
    - Compute all extensions s' at time t+1
    - Score s' from s
    - Put s' in v_{t+1} replacing existing s' if better
  - Advance to t+1

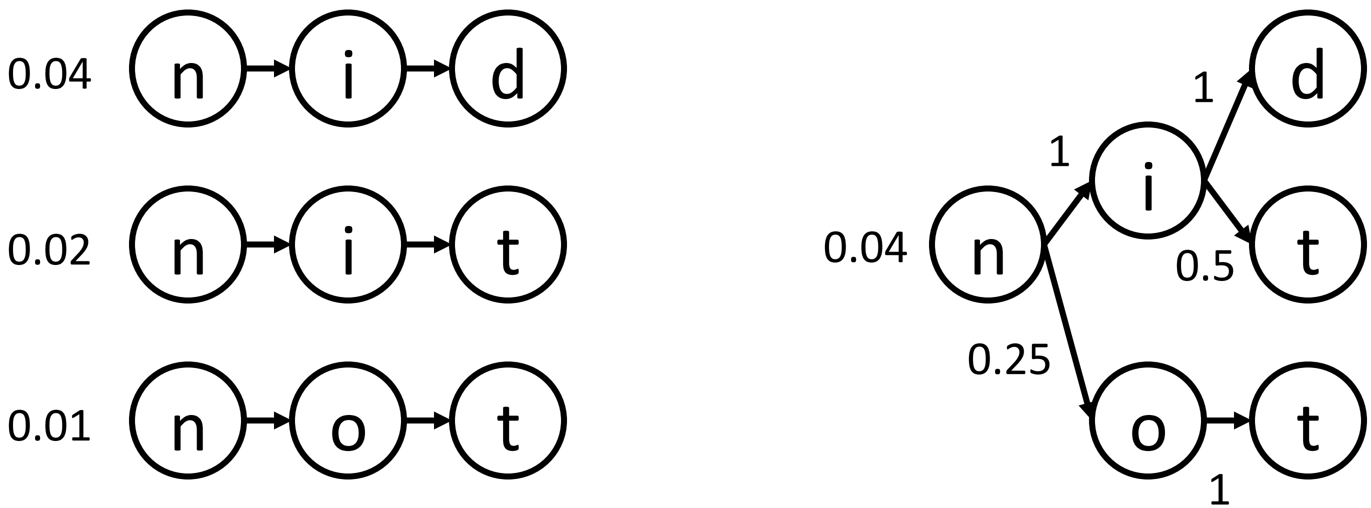
- Beams are priority queues of fixed size* k (e.g. 30) and retain only the top k hypotheses

Beam Search



Prefix Trie Encodings

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

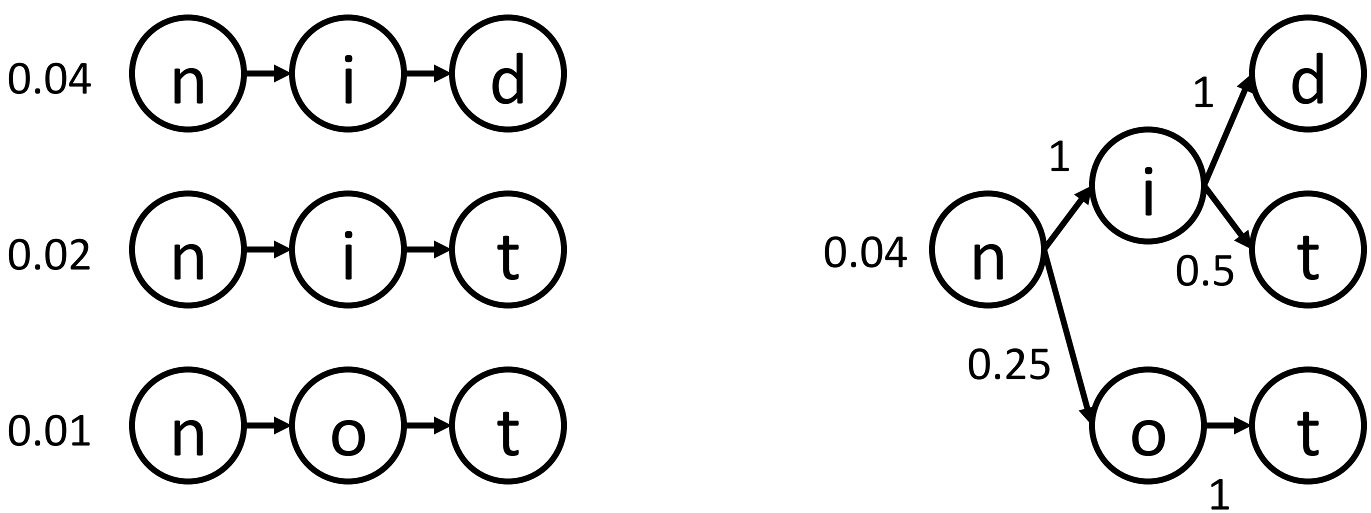


- A specific instance of minimizing weighted FSAs [Mohri, 94]

Example: Aubert, 02

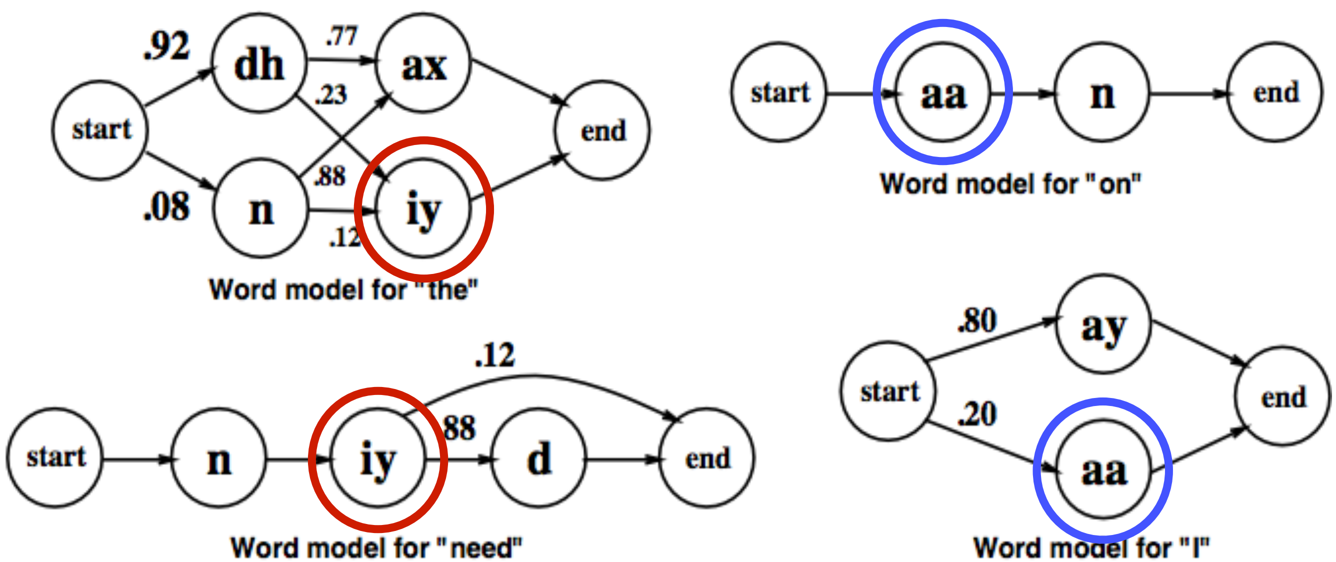
LM Score Integration

- Imagine you have a unigram language model
- When does a hypothesis get "charged" for cost of a word?
  - In naïve lexicon FSA, can charge when word is begun
  - In naïve prefix trie, don't know word until the end
  - … but you can charge partially as you complete it



Emission Caching

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



LM Reweighting

- Noisy channel suggests

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

- In practice, want to boost LM

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

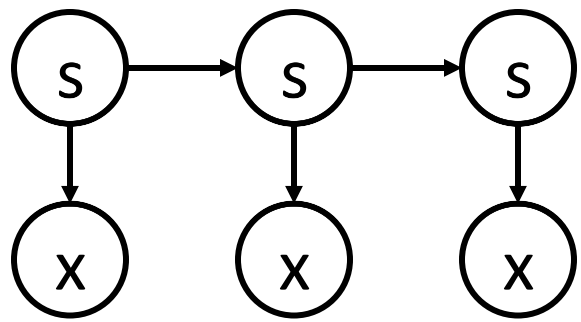
- Also, good to have a “word bonus” to offset LM costs

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

- The needs for these tweaks are both consequences of broken independence assumptions in the model, so won’t easily get fixed within the probabilistic framework

Training

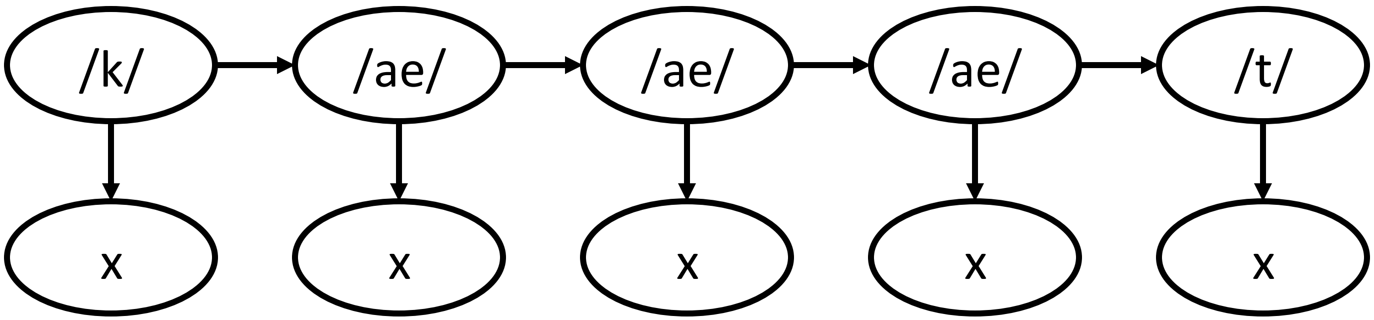
What Needs to be Learned?



- 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

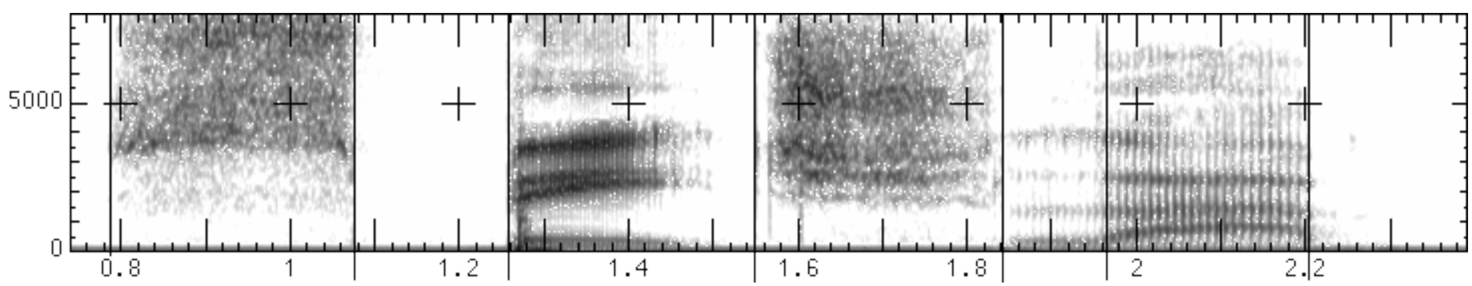
Forced Alignment

- 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"

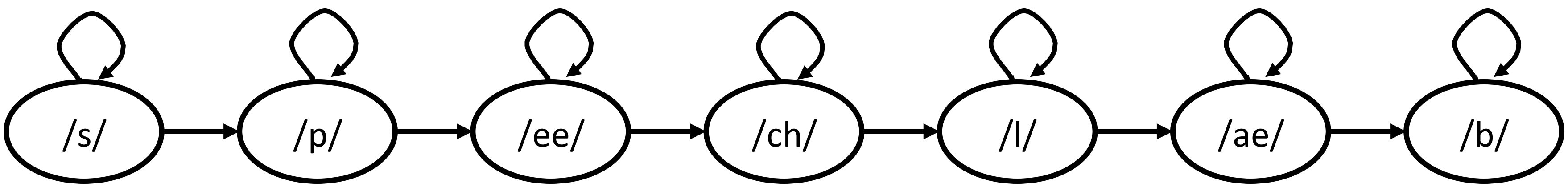
ssssssssppppeeeeeeetshshshshllllaeaeaebbbbb



- 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

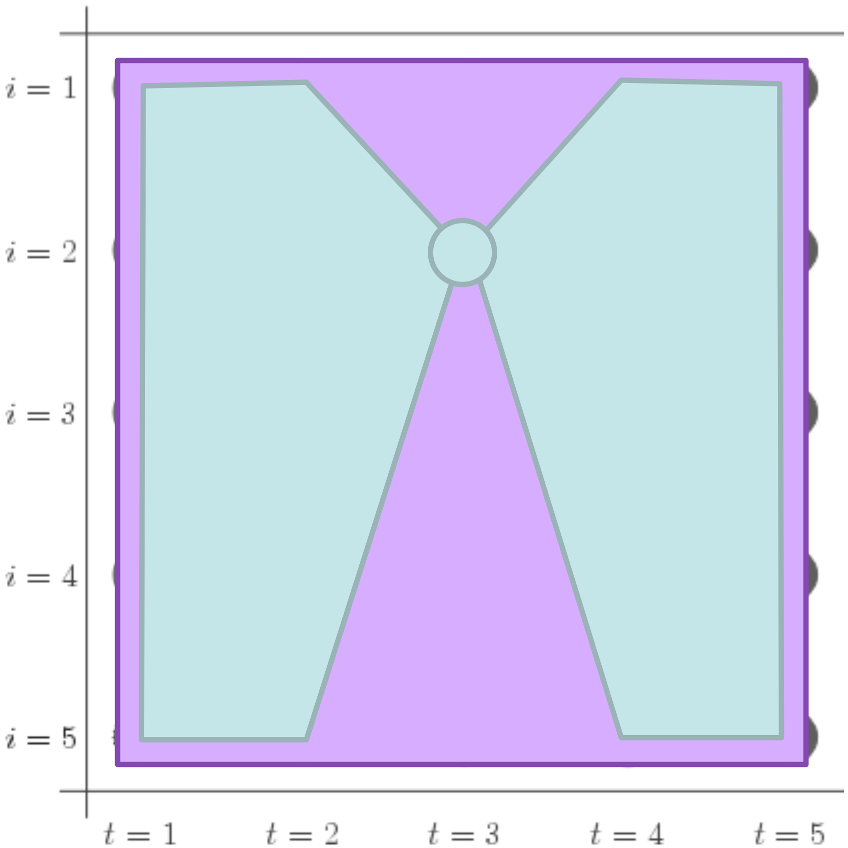
- Expectation Maximization (Hard EM for now)
  - Alternating optimization
  - Impute completions for unlabeled variables (here, the states at each time step)
  - Re-estimate model parameters (here, Gaussian means, variances, mixture ids)
  - Repeat
  - One of the earliest uses of EM!

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: $P(s_t \mid x)$

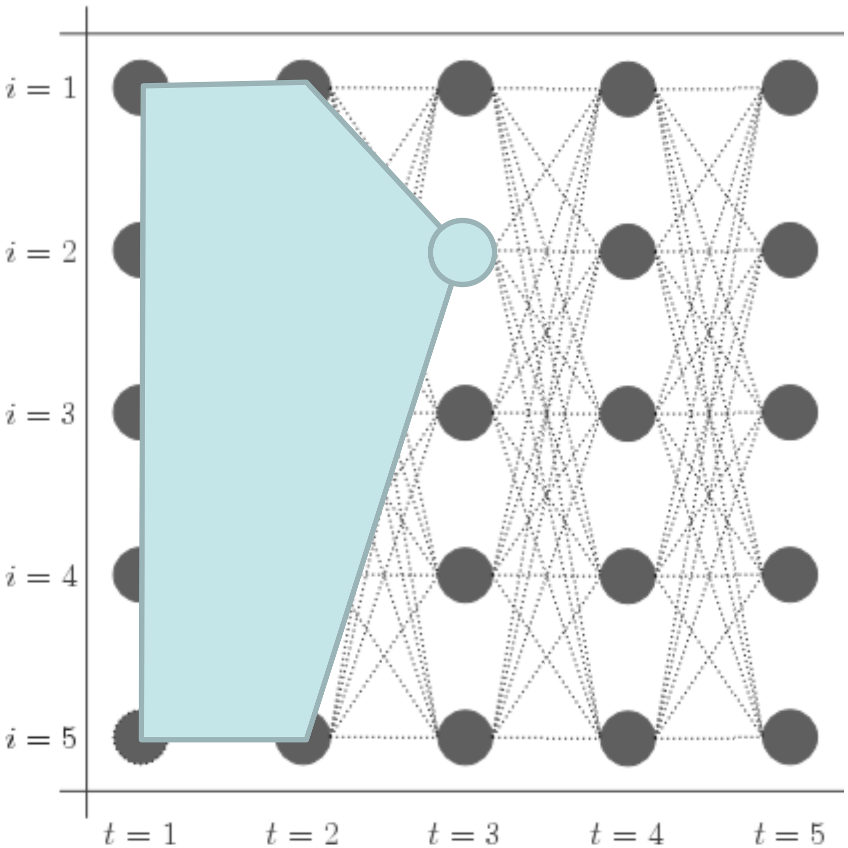
Computing Marginals



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

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

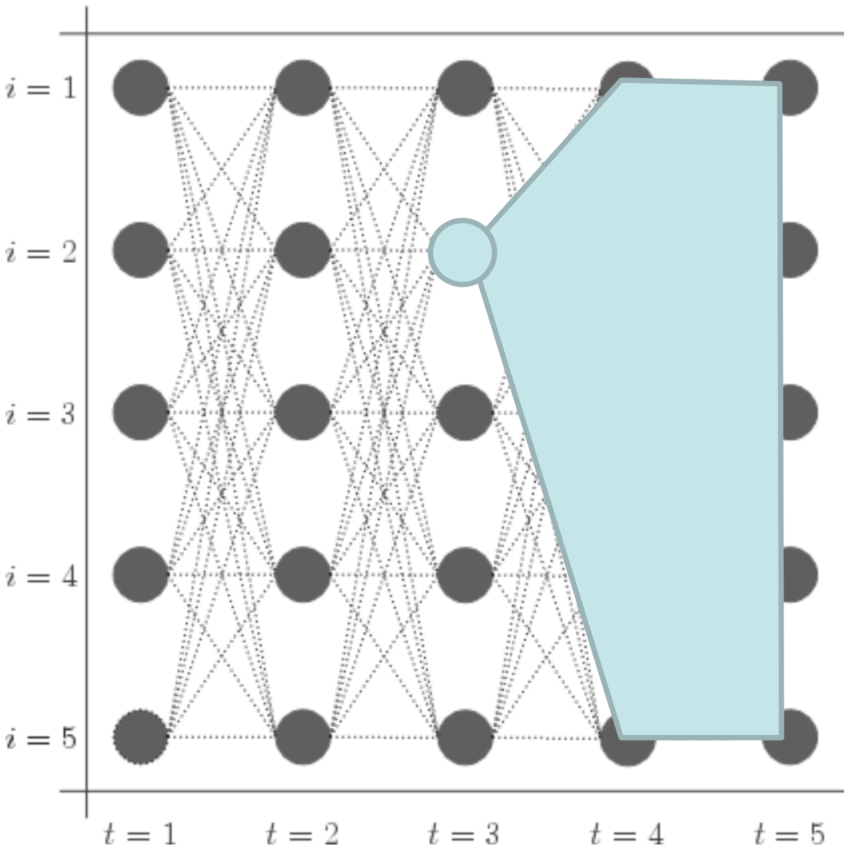
Forward Scores



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

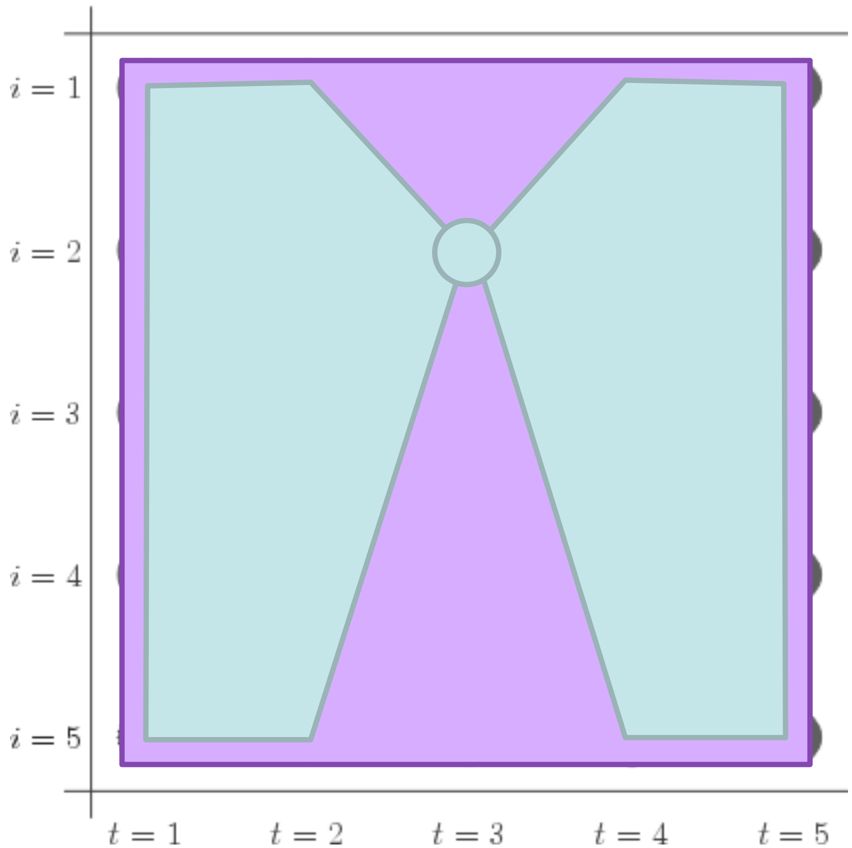
$$\alpha_t(s_t) = \sum_{s_{t-1}} \alpha_{t-1}(s_{t-1})\phi_t(s_{t-1}, s_t)$$

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