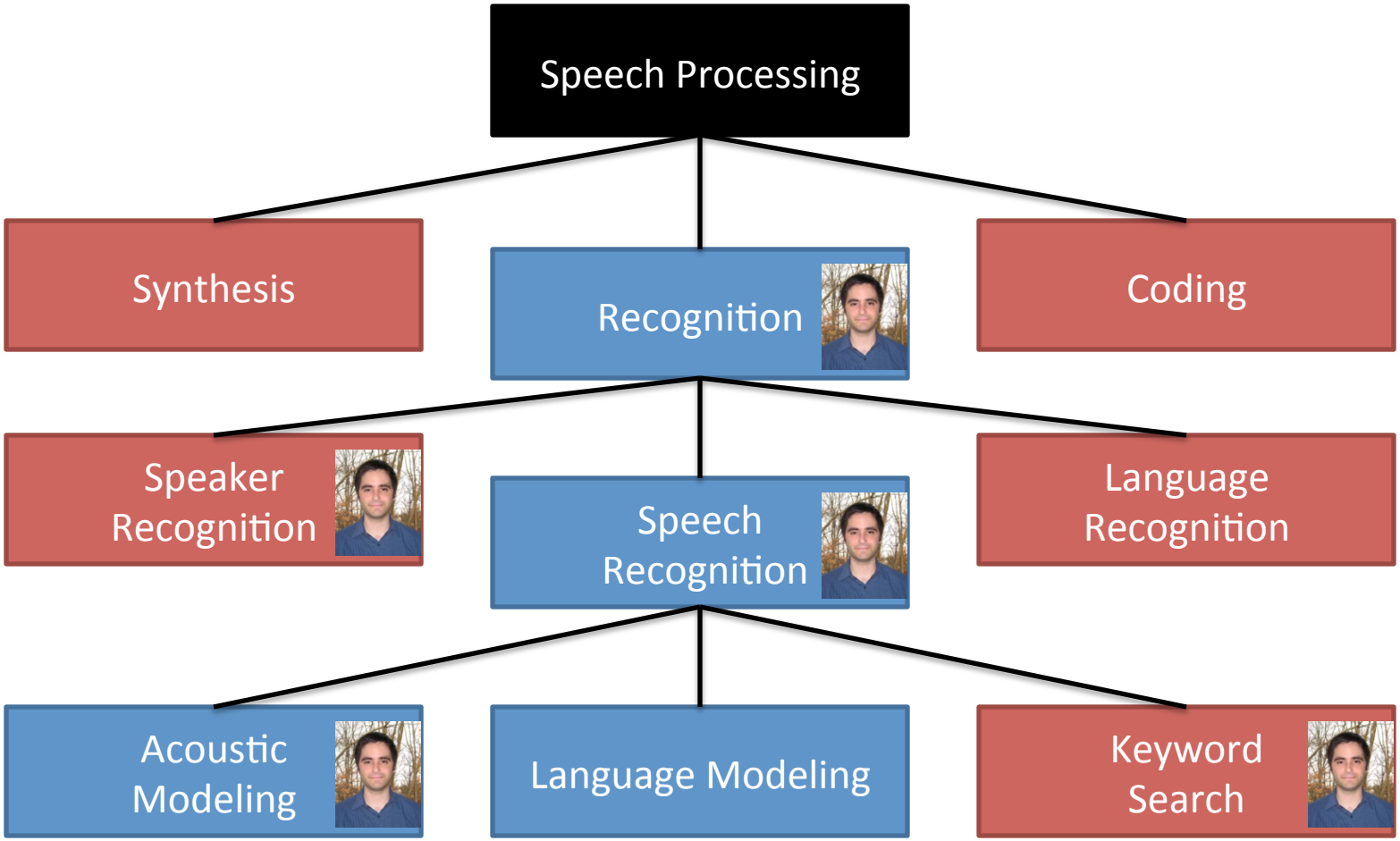
An Overview of Speech Technologies

Aren Jansen

JOHNS HOPKINS
UNIVERSITY

Thanks to Brian Kingsbury (IBM) and Hynek Hermansky (JHU)
for some of the materials contained in this lecture.

Core Automatic Speech Technologies



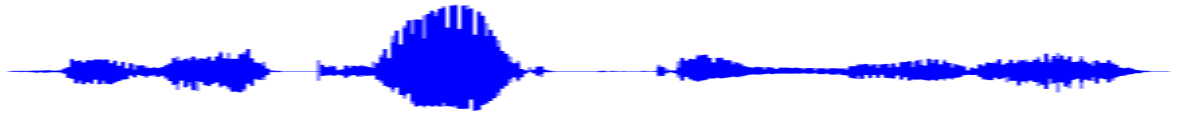
From i.i.d. Samples to i.i.d. Time Series

- Most of what you covered so far:
  - **Given:** Z = {(x_i,y_i)} pairs (for supervised case)
  - **Learn:** f(x) → y

- Speech recognition: vector time series, categorical labels not at the sample level:
  - **Given:** $X_i = x_1\ x_2 \ldots x_T$ where $x_t \in R^d$ and $Y_i = y_1 y_2 \ldots y_n$
  - **Notice:** n != T
  - **Learn:** f(X) → Y

Speech is Rich with Structure

Observed:



Acoustic:

Acoustic-Phonetic: voiced unvoiced voiced unvoiced voiced unvoiced voiced unvoiced

Phonetic: en s ai k l ow p iy d iy aa s

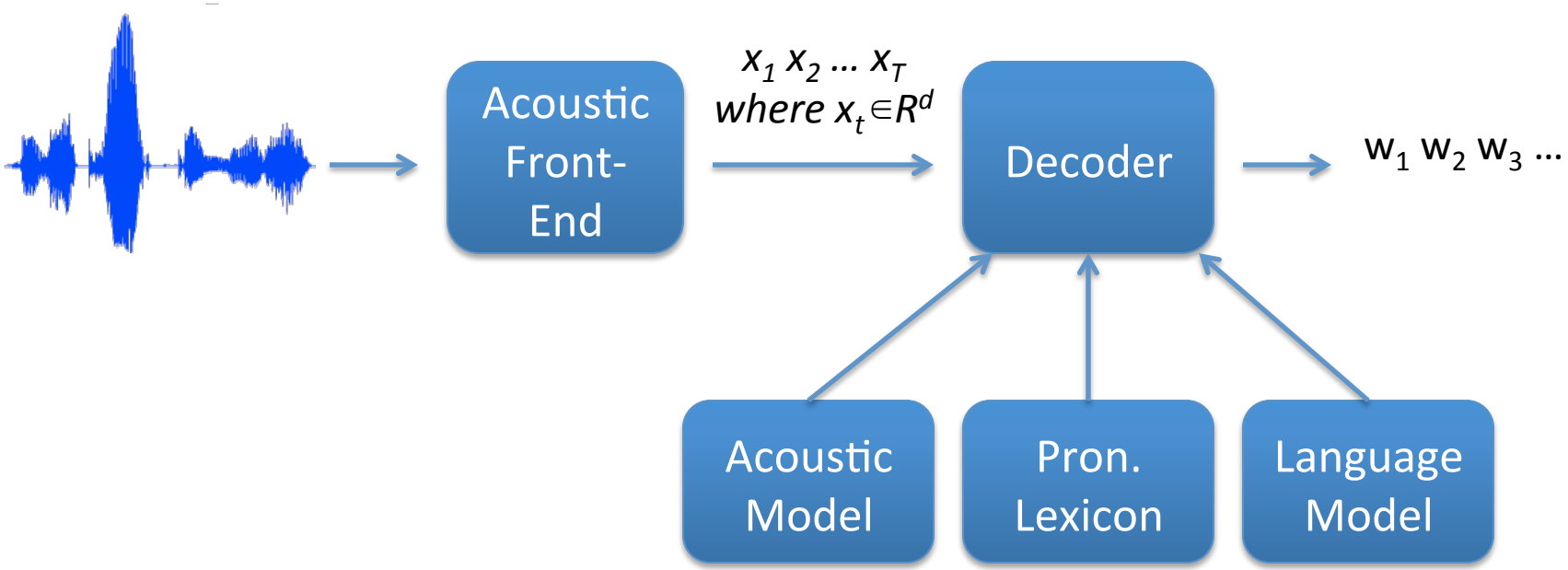
Lexical: encyclopedias

Grammatical: (he sold, NP, to her)

Semantic: {book, reference, knowledge, wikipedia}

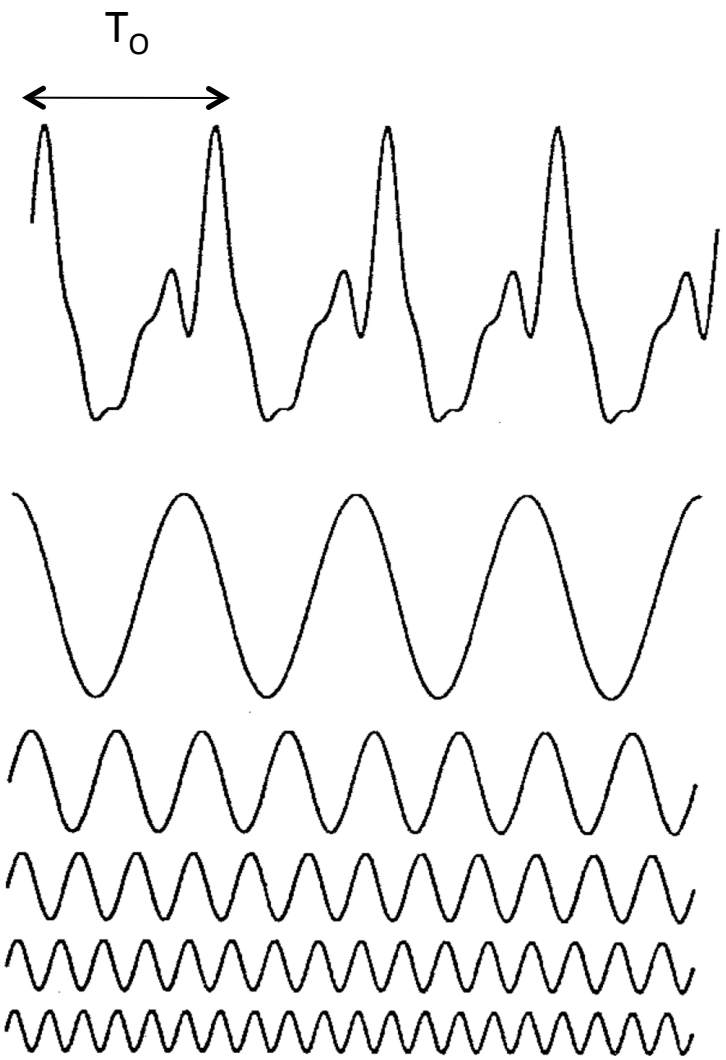
Speech recognition requires modeling of all levels of hierarchy

Automatic Speech Recognition Pipeline



All Speech Processing Begins Here

(in one form or another)

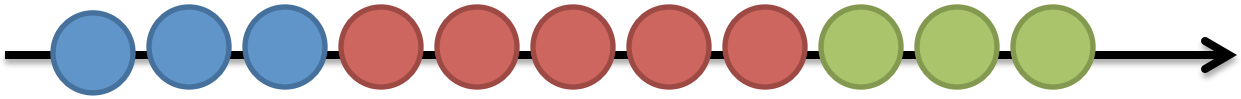


- **Fourier Analysis**

  decompose the signal into a sum of sinusoids across the whole range of frequency

Beads on a String

Speech is a quasi-stationary signal

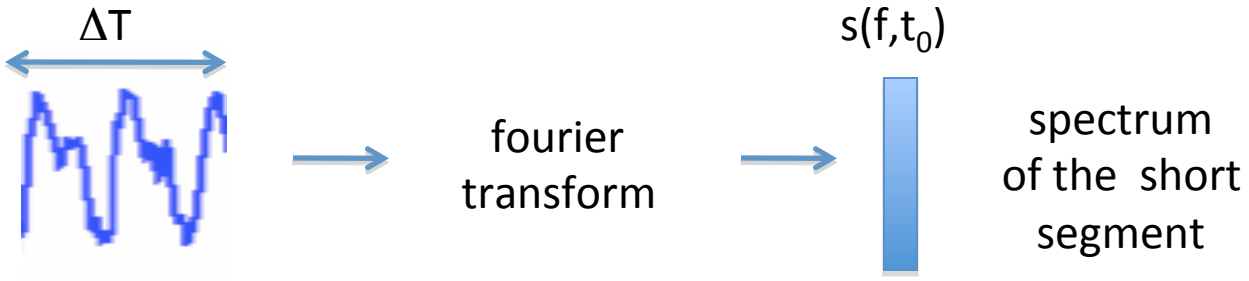


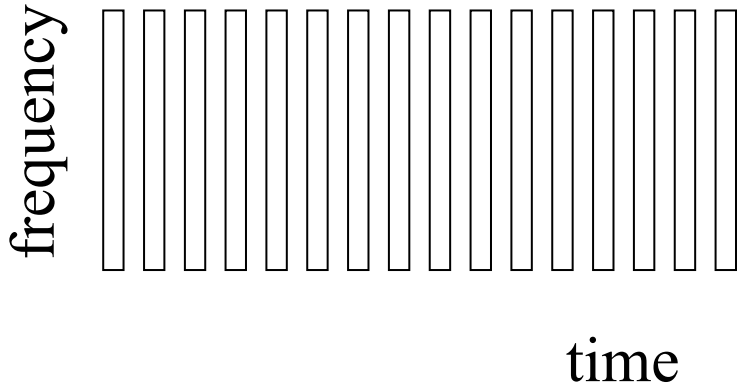
Time →

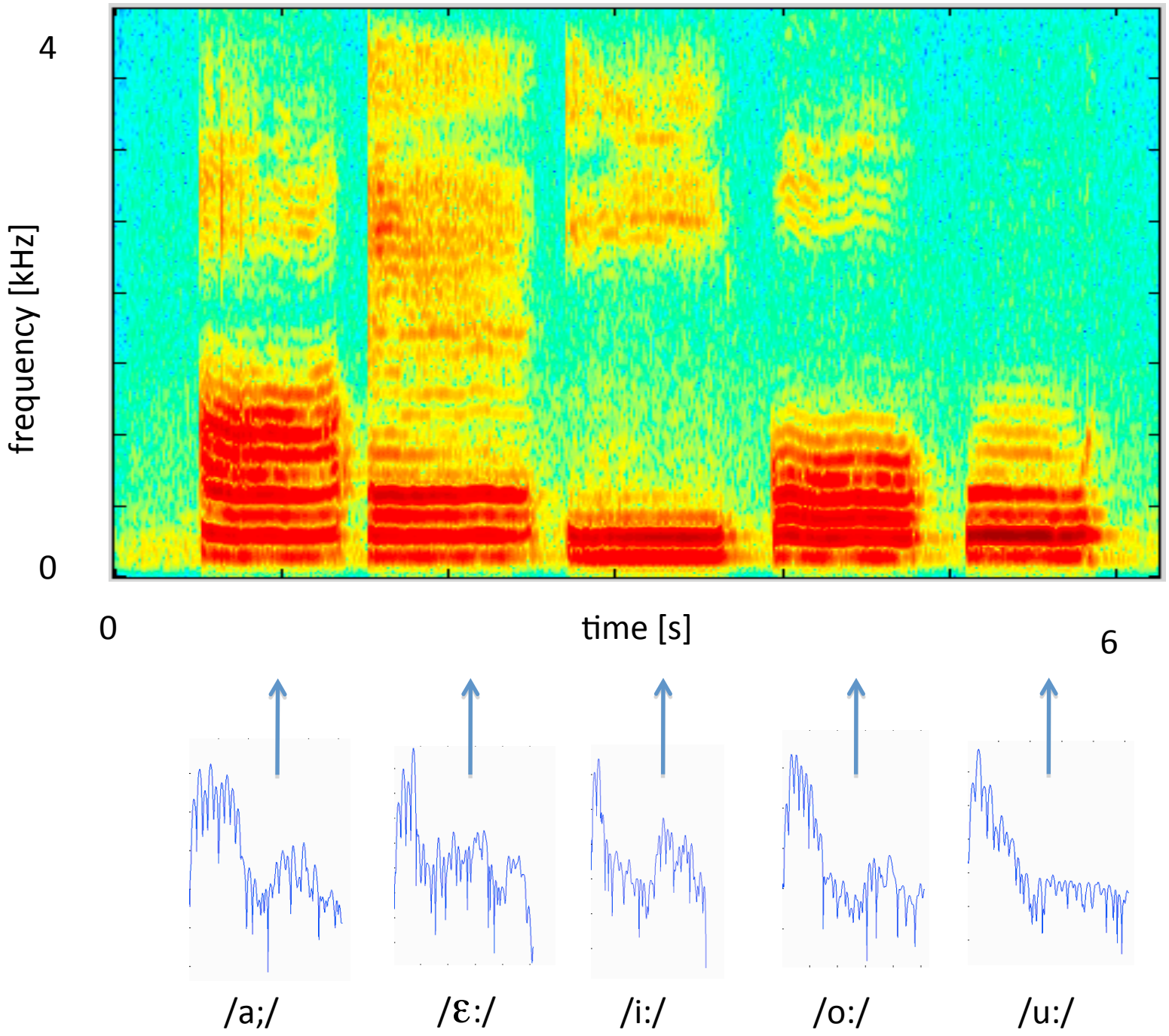
Short-time Analysis for Quasi-stationary Signals

t_0

time

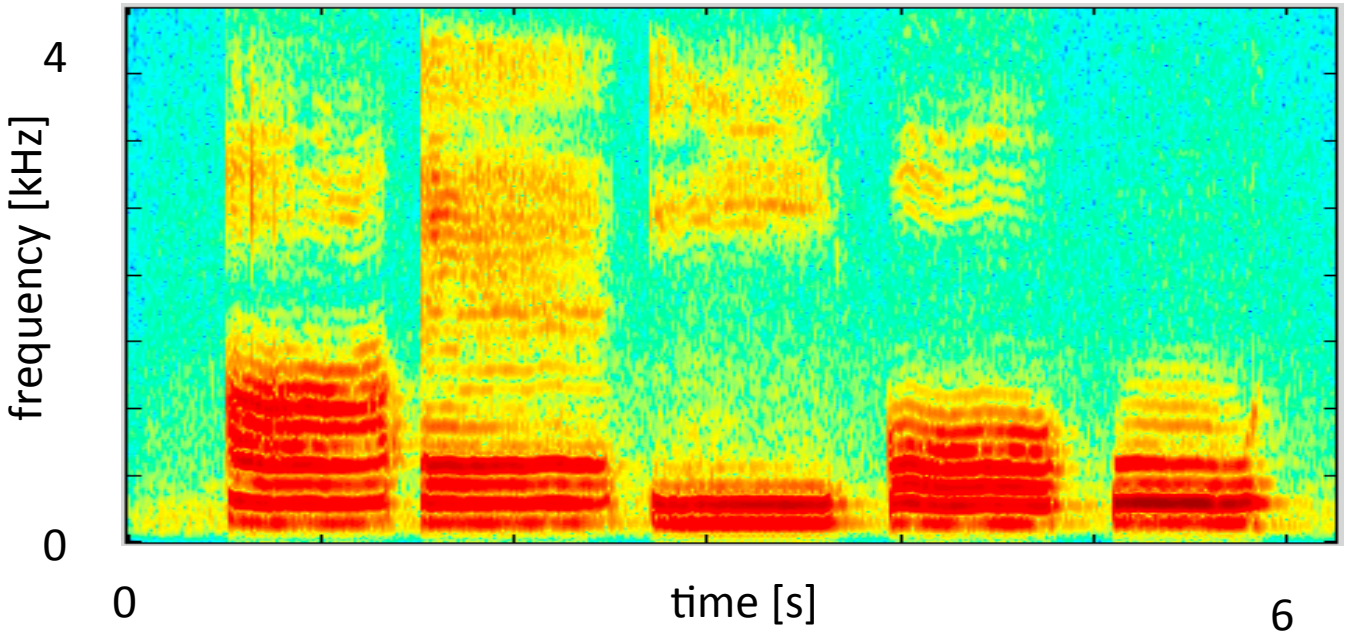





4
frequency [kHz]
0
0
time [s]
6
/a;/
/ɛ:/
/i:/
/o:/
/u:/

Removing Speaker Characteristics

- All speech recognition front-ends attempt to remove speaker dependent factors (so do speaker recognizers!)
- Typically accomplished using spectral smoothing of various types



Acoustic Front-ends

- The standards:
  - Mel Frequency Cepstral Coefficients (MFCCs)
    - Mel Scale and discrete cosine transform
  - Perceptual Linear Prediction (PLPs)
    - Bark Scale and linear prediction (and typically DCT)
- Data driven:
  - Neural Network Posteriorgrams
    - Use phonetically transcribed training data to train ANNs
- Recent trends:
  - Deep belief network pre-training
  - Spectro-temporal receptive fields (2D Gabors)

Standard Front-end Tricks

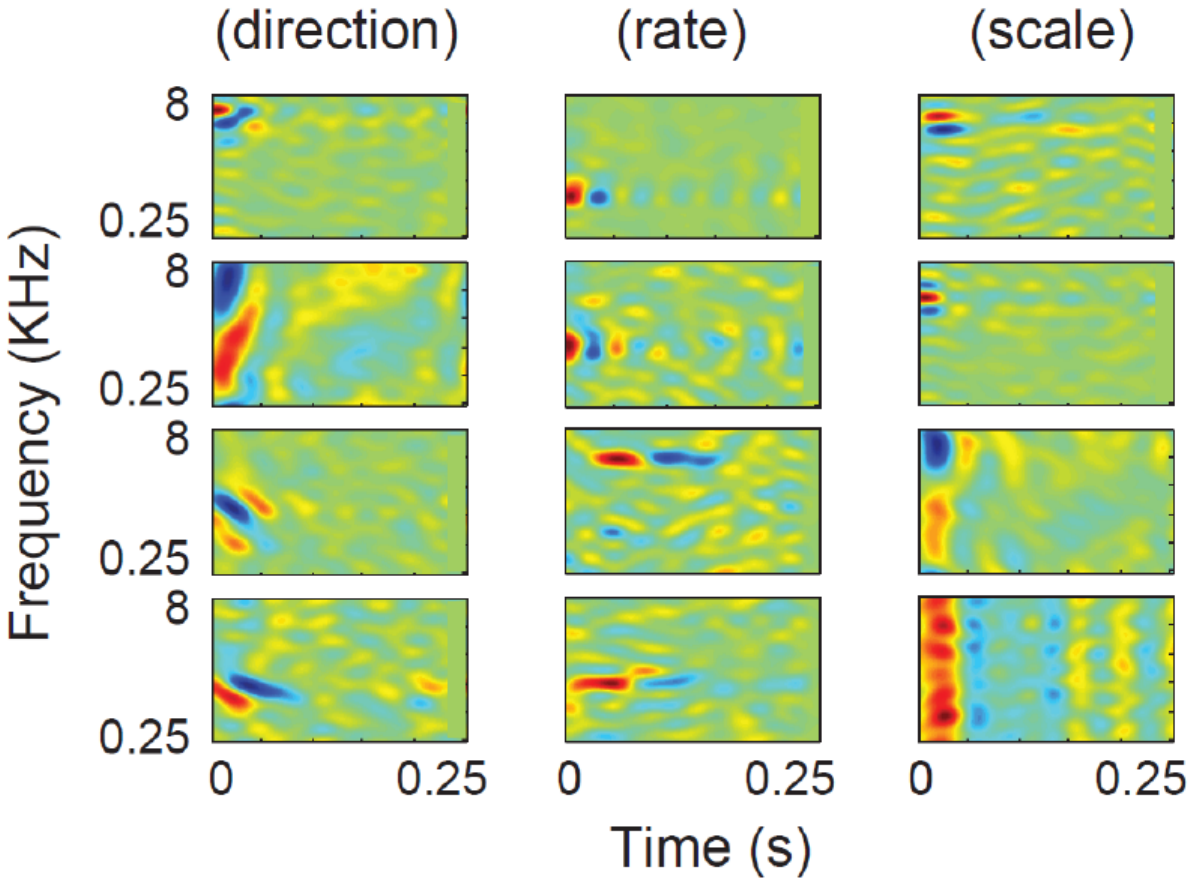
- Velocity and Acceleration features
  - Interested in changes (edges)
  - Instantaneous is noisy, so we average (slope of line fit to several points in trajectory)

- Temporal Context (+ LDA or PCA)
  - Form supervectors from several neighboring observation
  - Learn to reduce dimension with or without labeled data

- Cepstral Mean Subtraction
  - Compensation for convolutional noise (e.g. channel/ microphone variation)
  - $s' = s * n \rightarrow S'(f) = S(f)\,N(f) \rightarrow \langle \log S'(f) \rangle = \langle \log(S(f)) \rangle + \log(N(f))$

A Biologically Inspired Alternative

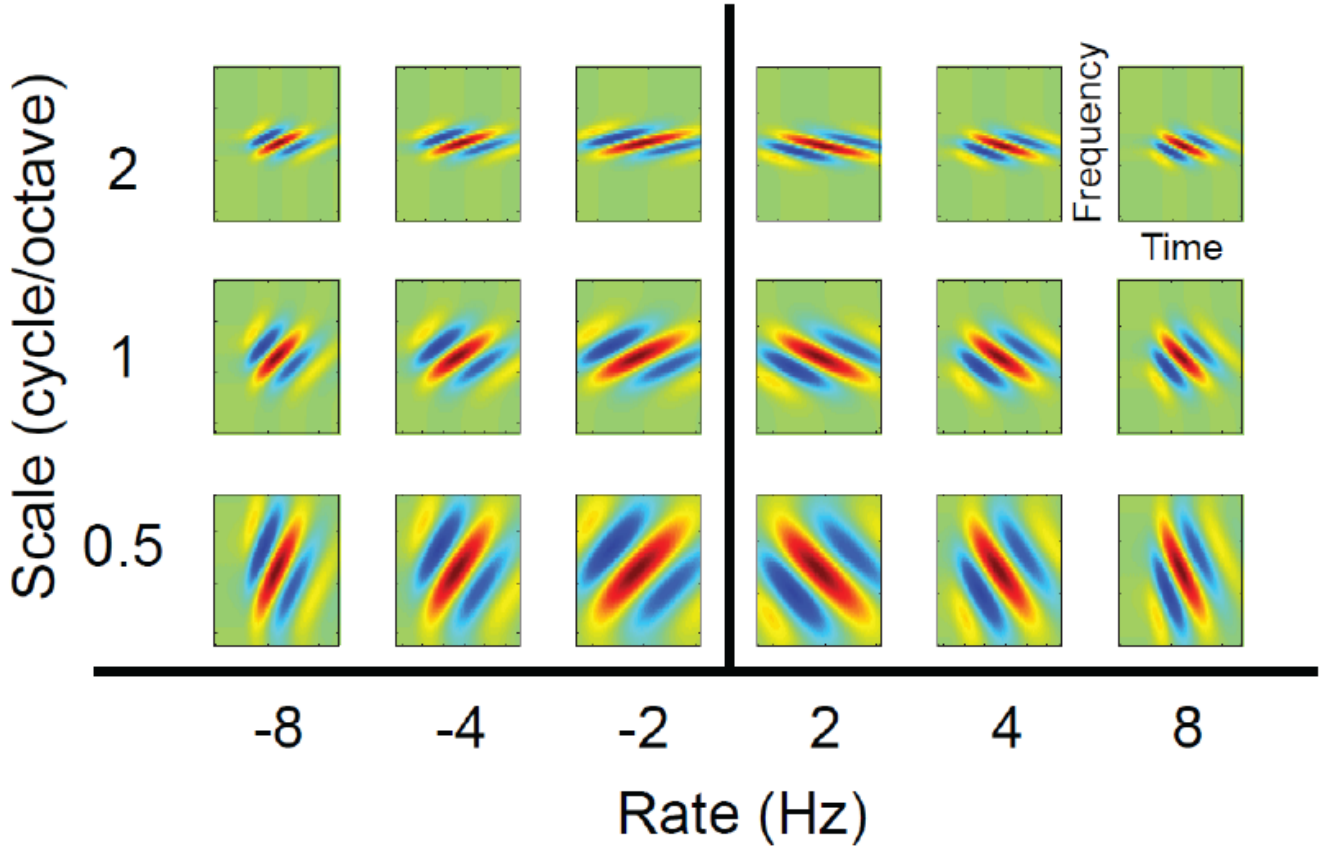
- Auditory neuron STRFs are tuned to a variety of frequencies, scales of spectral modulations, and rates of temporal modulations [Mesgarani, David, Fritz & Shamma, JASA 2008]



Filters Inspired by STRFs

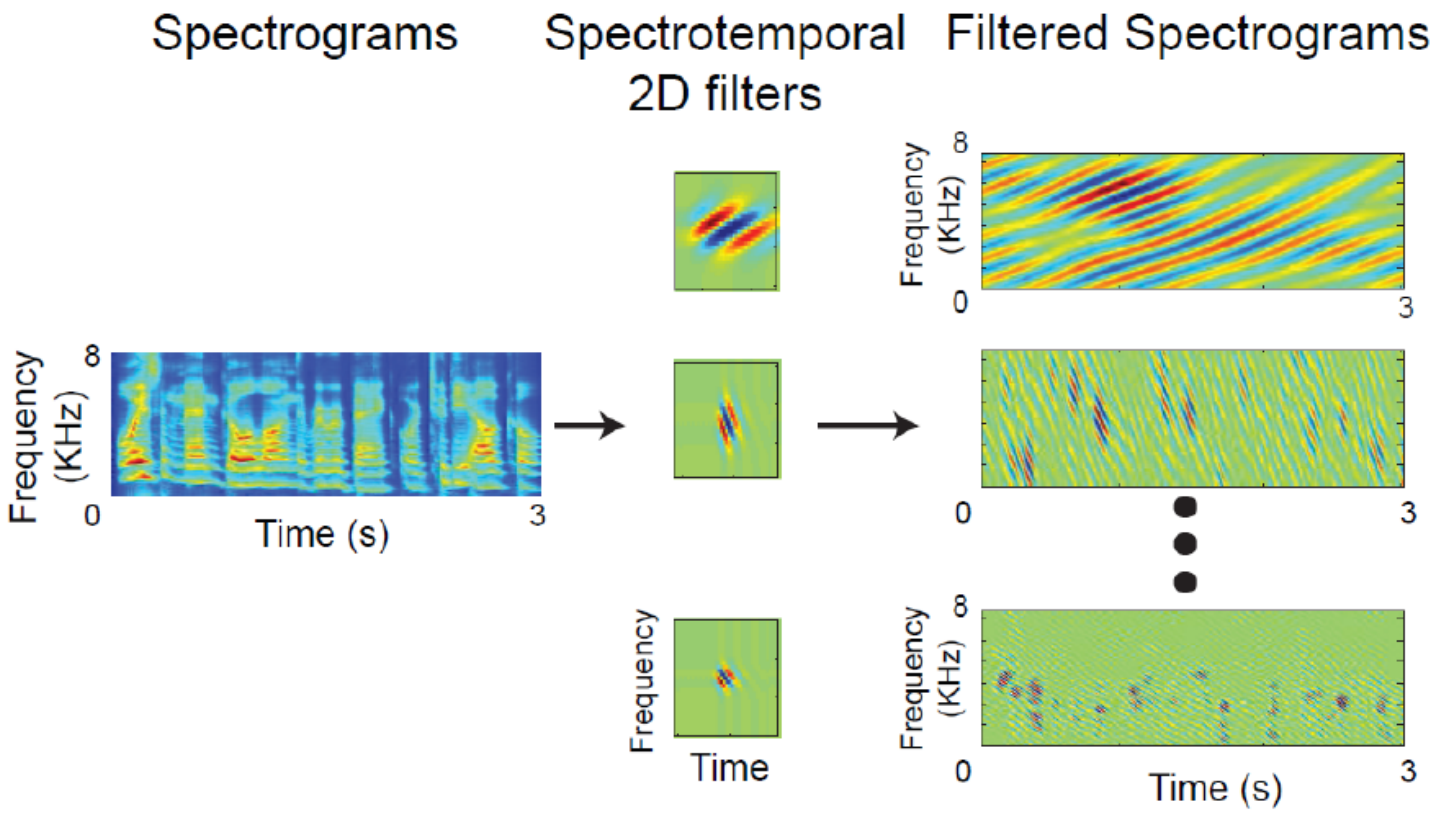
- Model real STRFs with the set of 2-D Gabor filters with the same tuning variations: frequency, rate, and scale

  [Mesgarani, Slaney & Shamma, TASLP 2006]

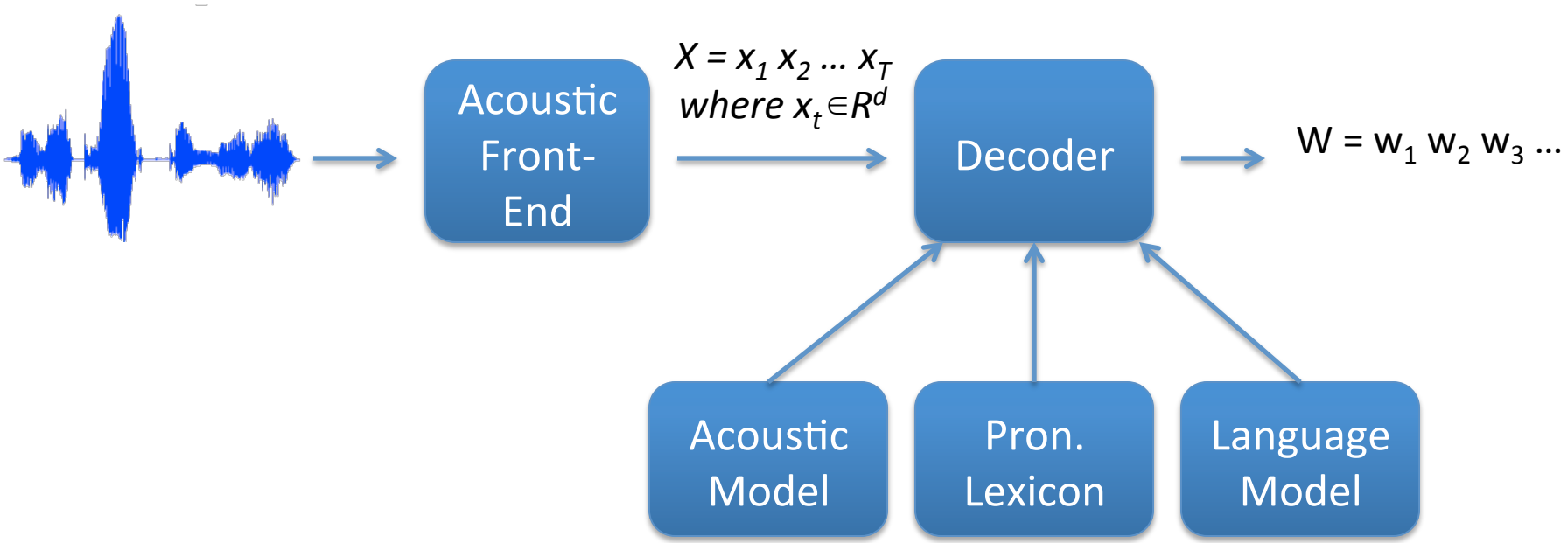


Spectro-Temporal Modulation Features

- Convolve set of spectro-temporal modulation filters with auditory spectrogram to produce a 4032-dimensional vector time series



Decoder



$$\widehat{\mathbf{W}} = \underset{\mathbf{W}}{\arg\max}\, P(\mathbf{X}|\mathbf{W};\Theta)P(\mathbf{W};\Theta)$$

Acoustic Model

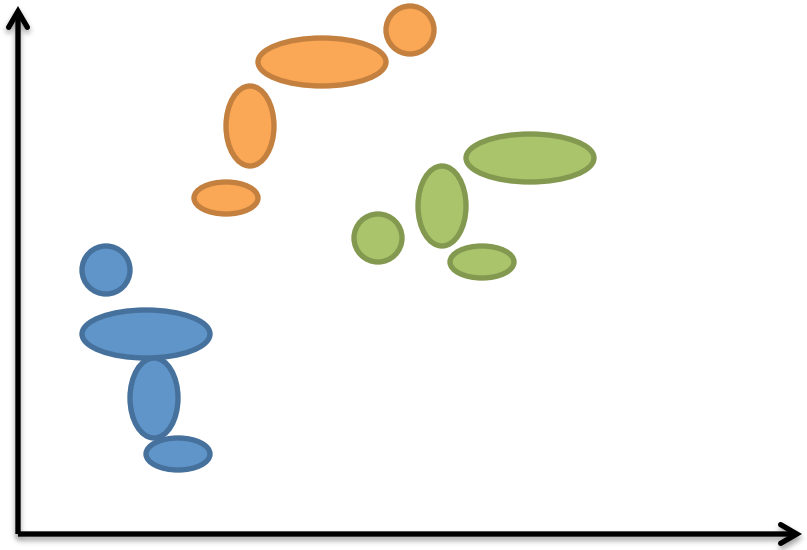
- Most acoustic models (AMs) are characterized in terms of phonemes
  - Phonemes are the atomic sounds of a given language
  - E.g. Cat = / k ae t /, Robot = /r ow b aa t/, The = /dh ah/ OR /th iy/
  - Natural classes exist in terms of confusions and production mechanisms
  - About 45 phones in English (depends on how you count)

- Phonetic AMs allow sharing of observations across context, reducing training data dependence

- Phonetic AMs allow generalization to new words (given pronunciation lexicon)

Modeling Individual Observations

- Each phonetic class is modeled with Gaussian Mixture Model (GMM)



$$p(x_t|q_t = i) = \sum_{j=1}^{K} \frac{w_{ij}}{|2\pi\Sigma_{ij}|^{1/2}} e^{-\frac{1}{2}(x-\mu_{ij})^T \Sigma_{ij}^{-1}(x-\mu_{ij})}$$

Context Dependent Phonemes

- Increase model complexity with context dependent phones:
  - One class for each phone in a particular phonetic context
  - E.g. triphones: (aa: k, t) OR (t: s, iy)
  - Not all 45^3 possibilities occur, so a fair amount of pruning is done

- Typically: pool of Gaussians shared by GMMs for all context dependent phonetic units (simple means of parameter sharing)

- Decision trees typically used to prune and determine how best to share parameters

GMM Training w/ Expectation-Maximization

- **E-step:** Given current GMM parameters θ, compute the posterior probability of each GMM component given the observation:

$$p(j|x_t, q_t = i) = \frac{\frac{w_{ij}}{|2\pi\Sigma_{ij}|^{1/2}} e^{-\frac{1}{2}(x-\mu_{ij})^T \Sigma_{ij}^{-1}(x-\mu_{ij})}}{p(x_t|q_t = i)}$$

GMM Training w/ Expectation-Maximization

- **M-step:** Compute the new expected maximum likelihood estimates θ' of the GMM means and covariances:

$$\mu_{ij} = \frac{1}{N_{ij}} \sum_t p(j|x_t, q_t = i) x_t$$

$$\Sigma_{ij} = \frac{1}{N_{ij}} \sum_t p(j|x_t, q_t = i)(x_t - \mu_{ij})^2$$

$$N_{ij} = \sum_t p(j|x_t, q_t = i)$$

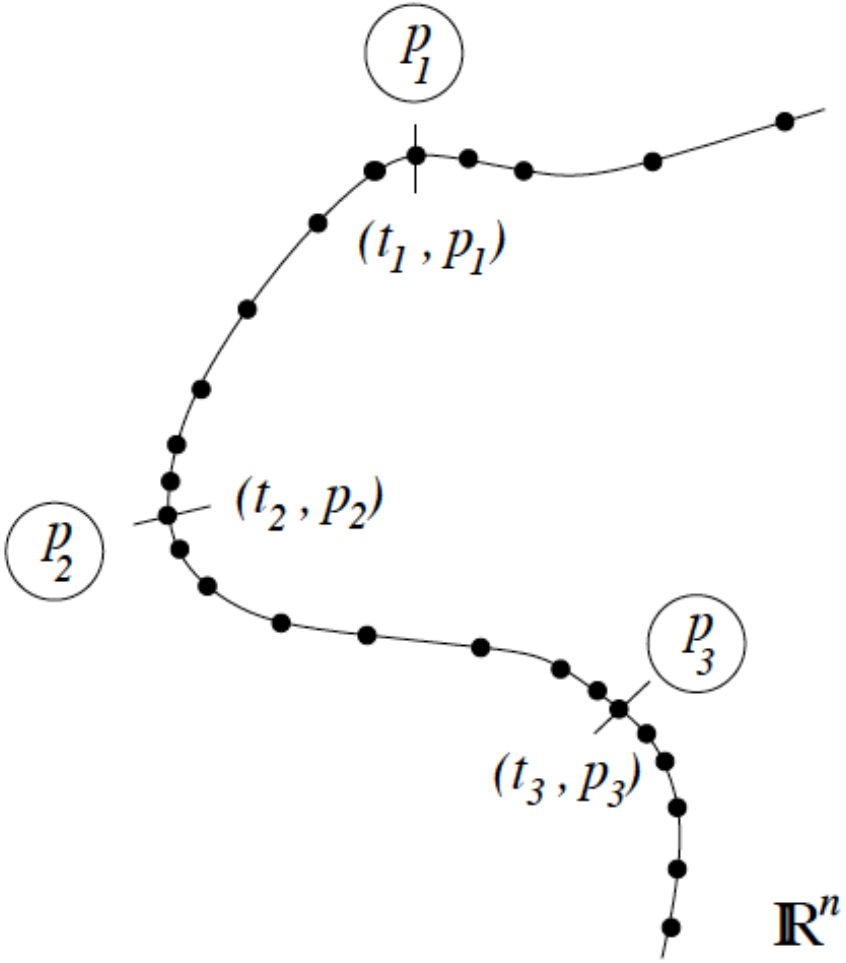
- **Iterate** E and M step until the total data likelihood converges

But We Don’t Have Frame Labels!

- We will also need to use E-M algorithm to decide which frames in training data belong to which phonetic class

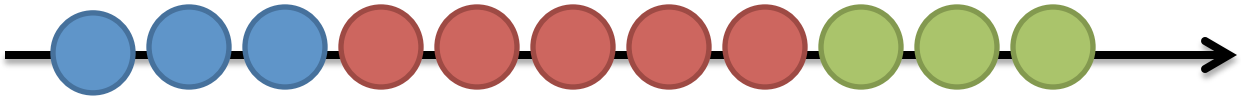
- **But:** We first have some temporal constraints to exploit

Trajectories ~~are~~ should be smooth



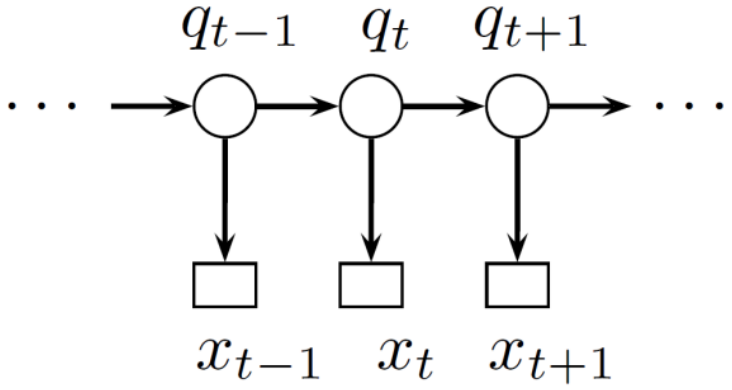
Modeling Temporal Dynamics

- Beads on a string model:



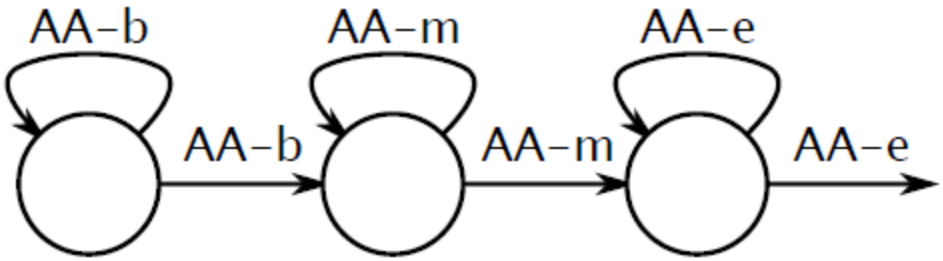
Time →

- Enter the Hidden Markov Model (HMM):



Typical Phone HMM Topology

- Three states per context dependent phone unit (states model entry, stable part, and exit of the phoneme)



- In total, three states per context dependent phone, $O(48^3)$ context dependent units per phoneme (a very large number)

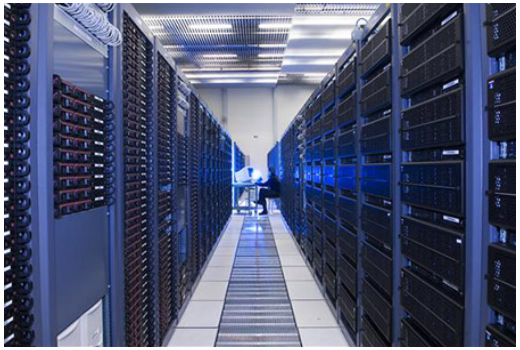
Expectation-Maximization Training (Again)

- HMM acoustic model trained with forward-backward algorithm (Baum-Welch)
  - Allows us to compute $P(q_t = i \mid x_t)$ given HMM-GMM parameters
  - A polynomial time dynamic program to an otherwise exponential time problem
  - Made possible by first order Markov property of HMM

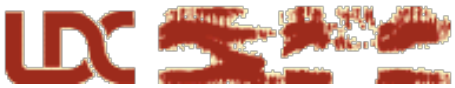
1. Use Baum-Welch to get responsibility of each frame to each state
2. Run E-M training of GMMs given these responsibilities
3. Estimate maximum expected log likelihood HMM parameters
4. Iterate until total data likelihood converges (under whole HMM-GMM model)

Acoustic Modeling, 1990s-present

- Basic prescription hasn't changed since late 80s
- Most advances from fast computers and big data



+



- 200k+ Gaussians (1 hour of speech!)
- 100k+ words
- Quinphone states
- Speaker/noise model adaptation
- Discriminative model re-training
- **2000+ hours** of transcribed speech

Alternatives

- Still only given labels at the segment level, not frame level
- Alternatives typically rely on some sort of EM procedure to get word or frame alignments
- **Most Common:** Use HMM-GMM recognizer for frame level alignments
- E.g. neural networks (tandem and hybrid)

Language Model

- Necessary for several reasons:
  - Simplify decoding by ruling out impossibilities (e.g. “She can’t do it”, NOT “she cat do hit”)
  - Compensate for bad acoustics / bad acoustic model
  - Disambiguate homonyms: “write a letter to Ms. Wright, right now”

N-gram Language Model

$$
\begin{aligned}
P(\mathbf{W}) = & p(w_1|<\mathrm{s}>)\times \\
& p(w_2|w_1<\mathrm{s}>)\times \\
& p(w_3|w_2w_1<\mathrm{s}>)\times \\
& \cdots \\
& p(</\mathrm{s}>|w_m w_{m-1}\cdots w_2 w_1<\mathrm{s}>)
\end{aligned}
$$

is approximated by

$$
\begin{aligned}
P(\mathbf{W}) \approx & p(w_1|<\mathrm{s}><\mathrm{s}>)\times \\
& p(w_2|w_1<\mathrm{s}>)\times \\
& p(w_3|w_2w_1)\times \\
& \cdots \\
& p(</\mathrm{s}>|w_m w_{m-1})
\end{aligned}
$$

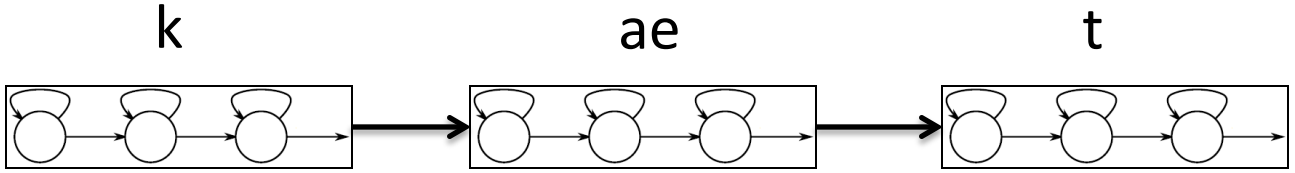
Note: smoothing and back-off required

N-grams Still Reign

- The simplest answer is still the most common in practice
- More complicated models (synatactic, recurrent neural networks) provide improvements but are intractable
- N-best list/Lattice rescoring are work-arounds
- Discriminative training has also been useful

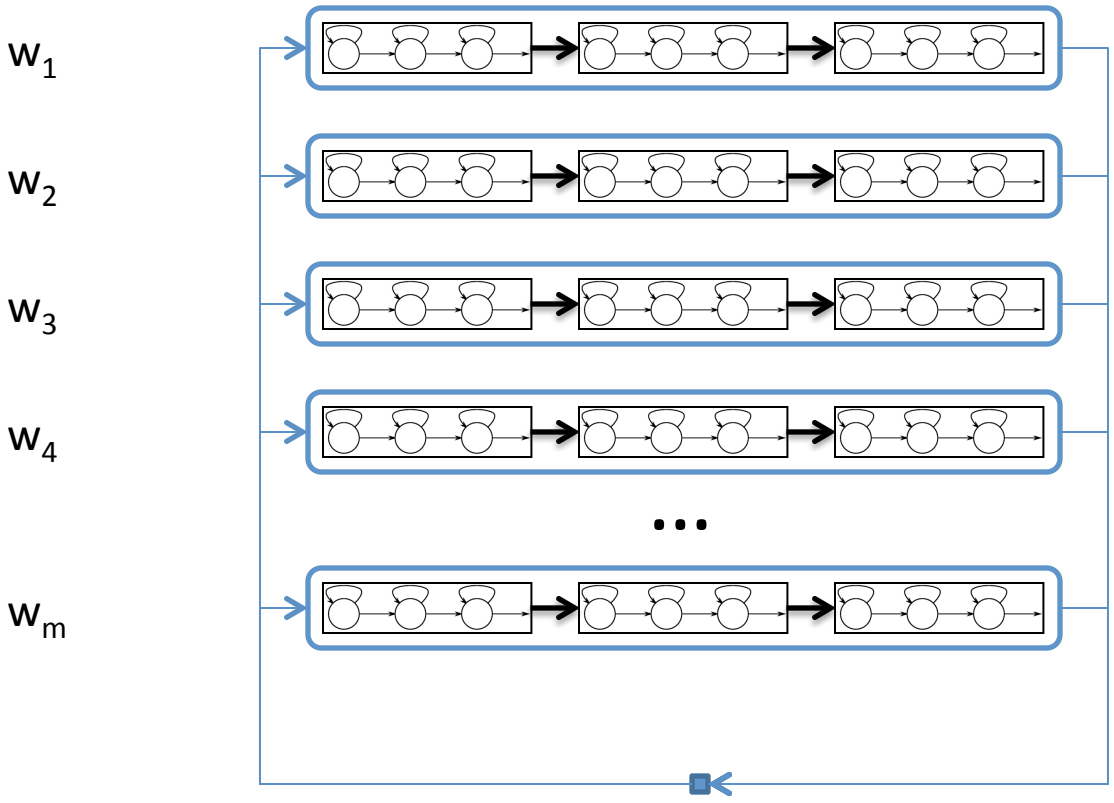
Pronunciation Lexicon

- The lexicon (dictionary) is the connecting glue between phonetic acoustic model and the language model
- Word models are constructed from dictionary, e.g. cat:



Decoding

- Decoding Graph (common: 45M states, 190M arcs)



Viterbi Algorithm

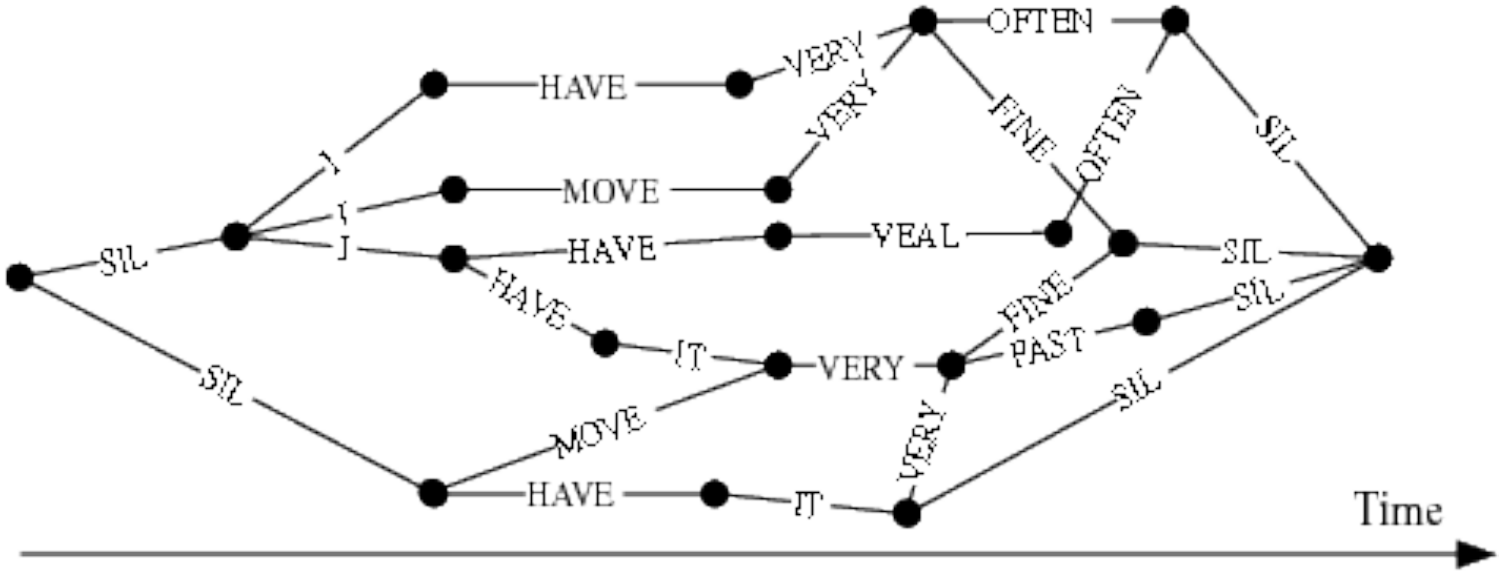
- Find the most likely state sequence $Q = q_1 \ldots q_T$ given the acoustics $X = x_1 \ldots x_T$
- Under 1st order Markov property, this simplifies:

$$Q^* = \max_Q P(Q \mid X) = \max_Q P(X \mid Q)P(Q)$$

$$= \prod_{t=1}^{T} P(x_t \mid q_t) \prod_{t=2}^{T} P(q_t \mid q_{t-1})$$

- Viterbi algorithm is a quadratic time dynamic program that takes advantage of the Markov property

Lattice Output



Word Error Rate

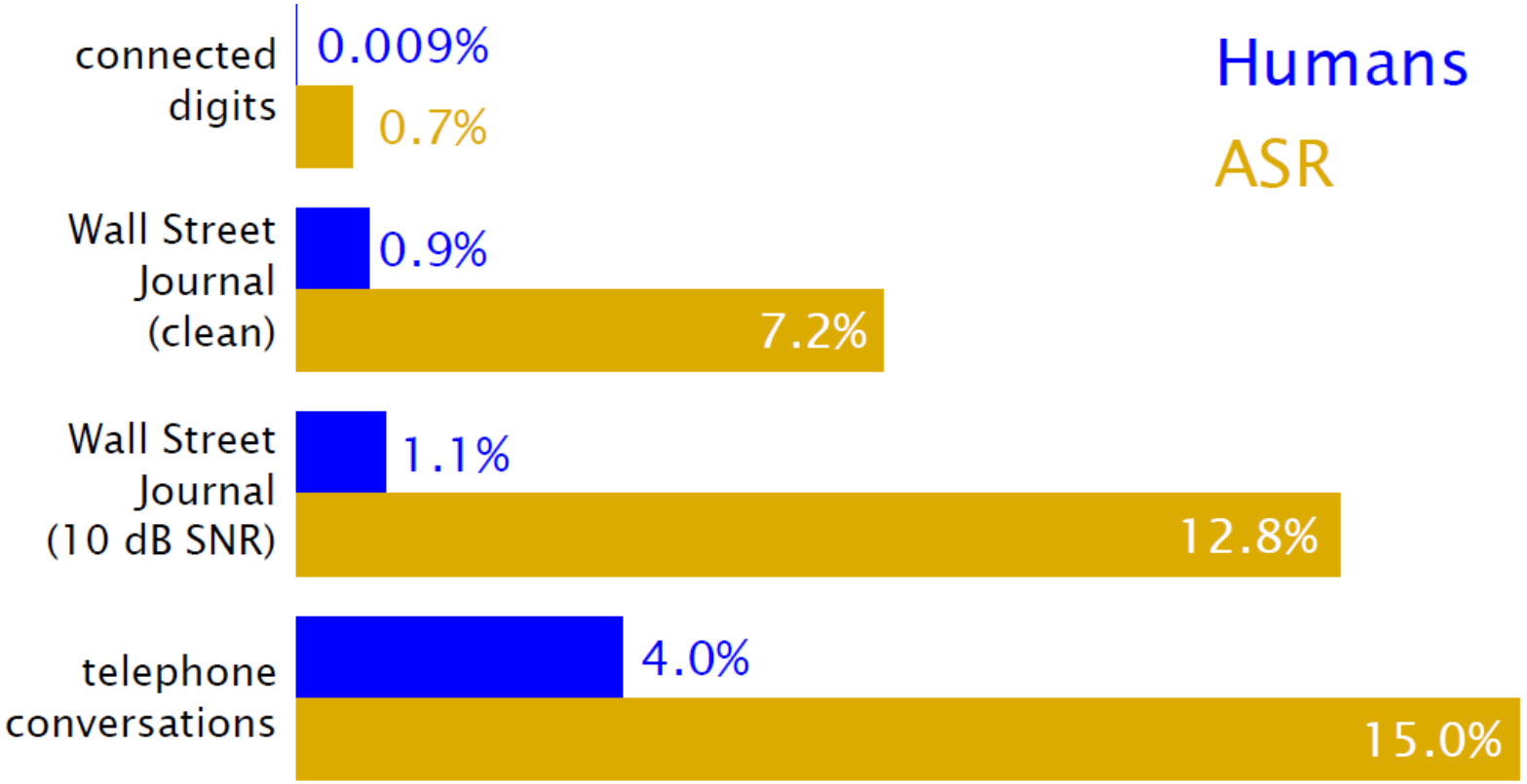
- String edit alignment between reference and hypothesized word strings (as word token sequences)

- WER = 100 x $(N_{sub} + N_{ins} + N_{del})/N_{ref}$

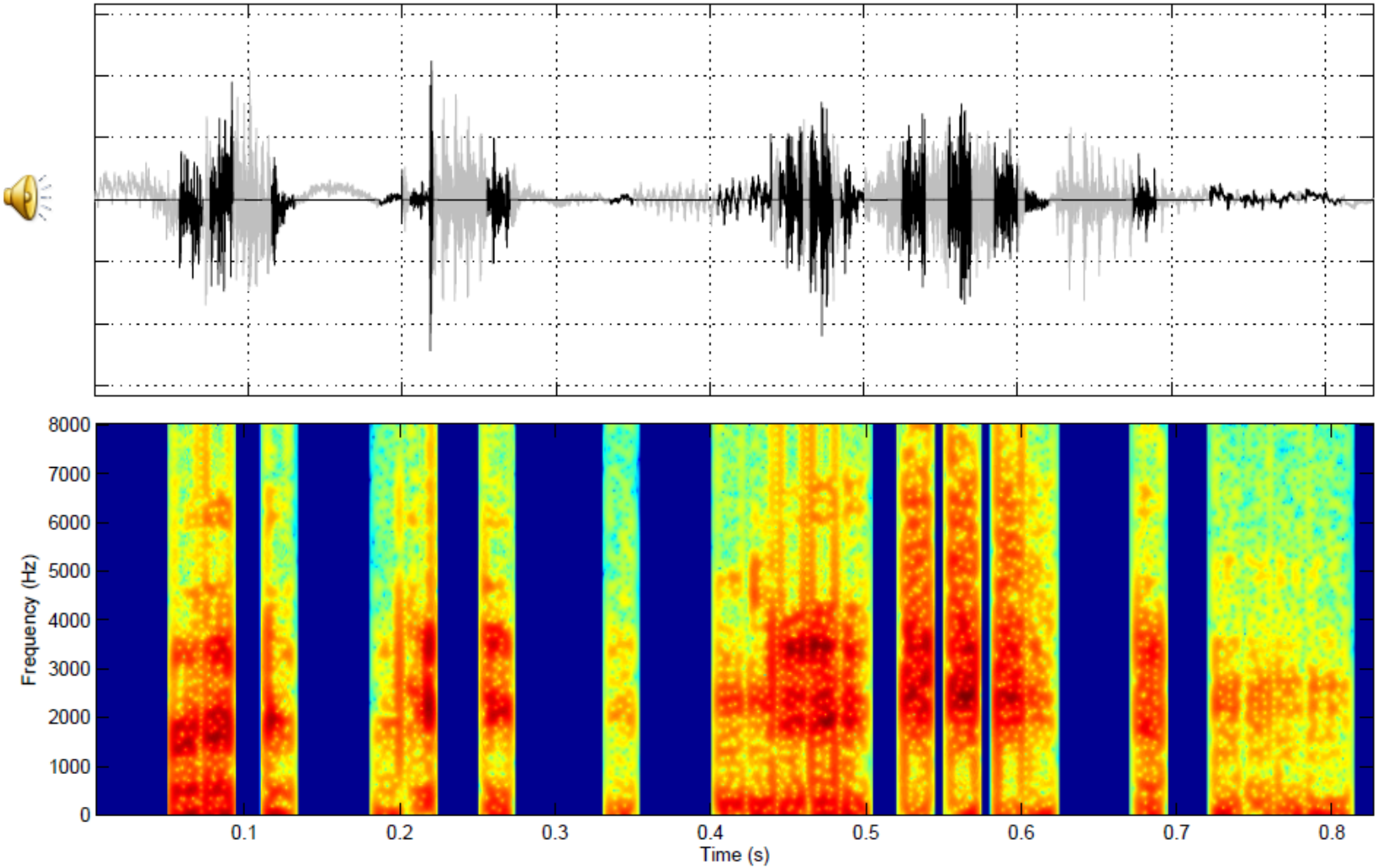
REF	The		dog	ate	my	homework
HYP	A	a	dog	ate		homework
ERR	sub	ins			del	

A Long Way To Go

- Humans vs. Machines (as of late 1990s):



Not All Frames Are Created Equal



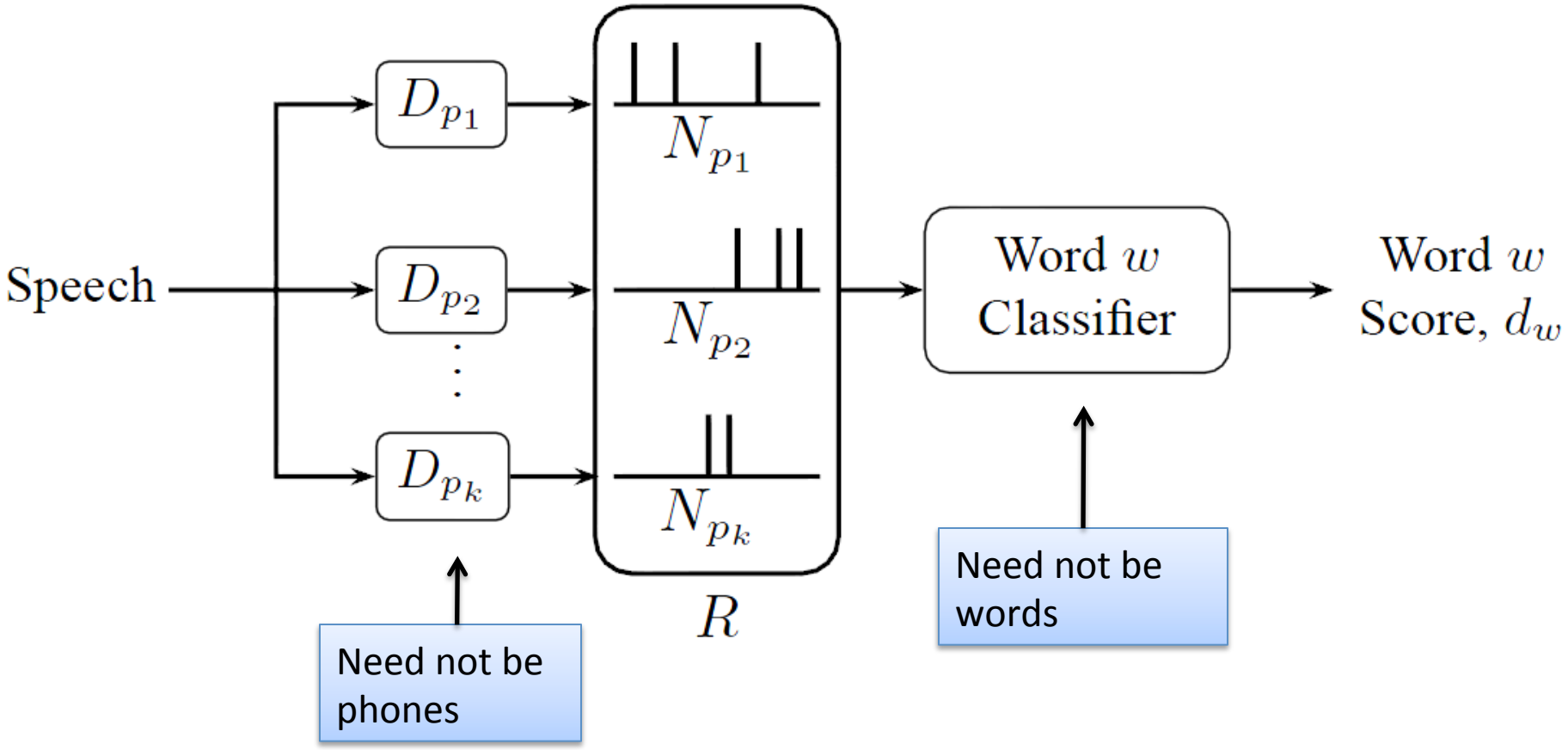
High entropy frames removed to mask 60% of the signal!

Point Process Models

1. Transform the signal into sparse temporal point patterns of acoustic events

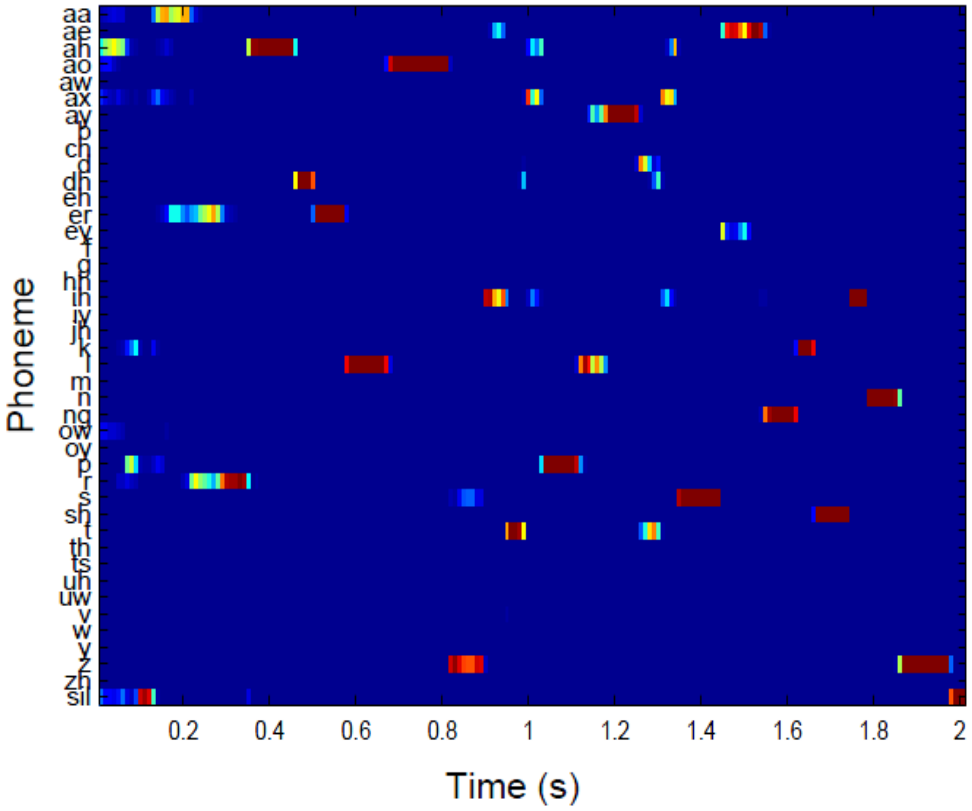
2. Explicitly model **whole words or common phrases** according to the temporal statistics of these patterns

PPM Architecture



Phonetic Events

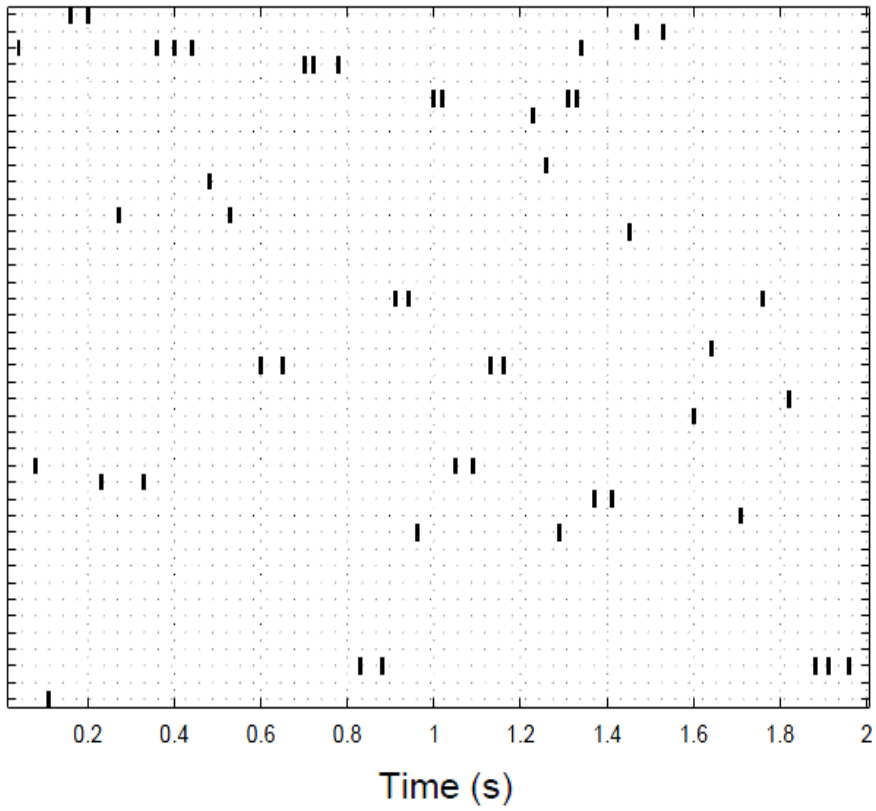
Phonetic Posteriorgram



8000+ real-valued probabilities

Sparse across phones, not time

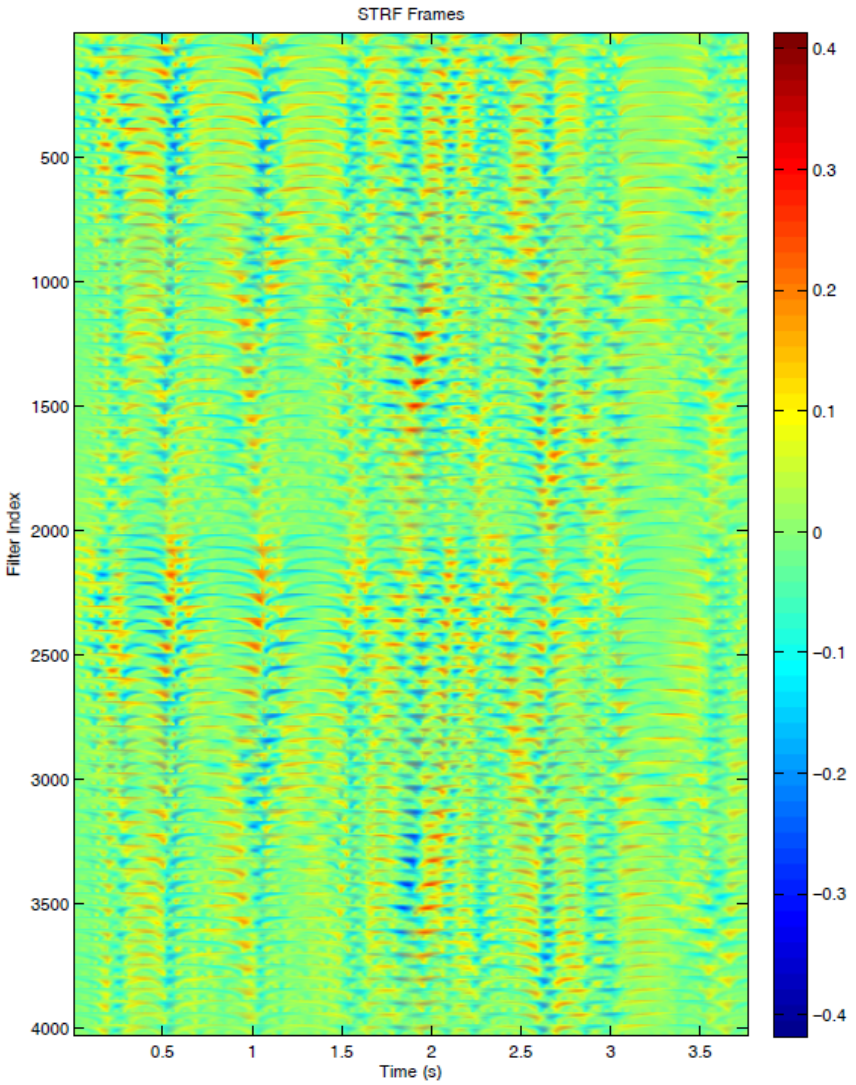
Phonetic Events

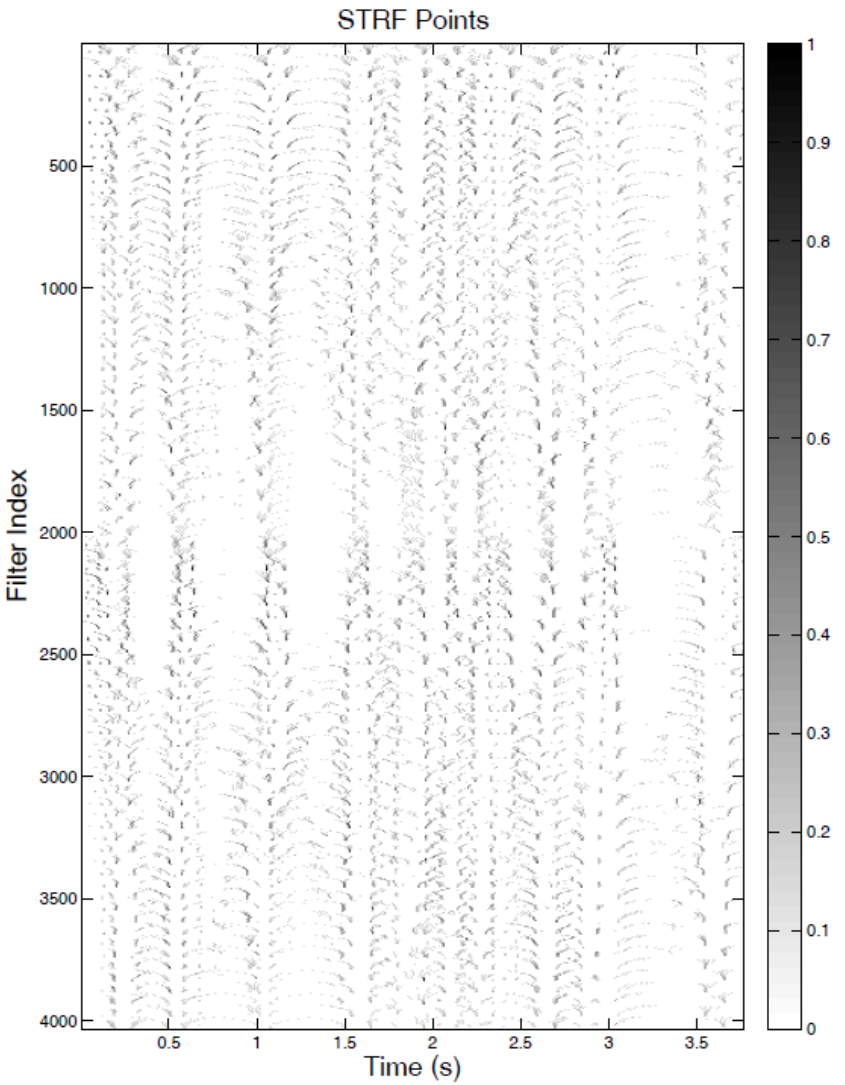


48 real-valued event times

Sparse in time

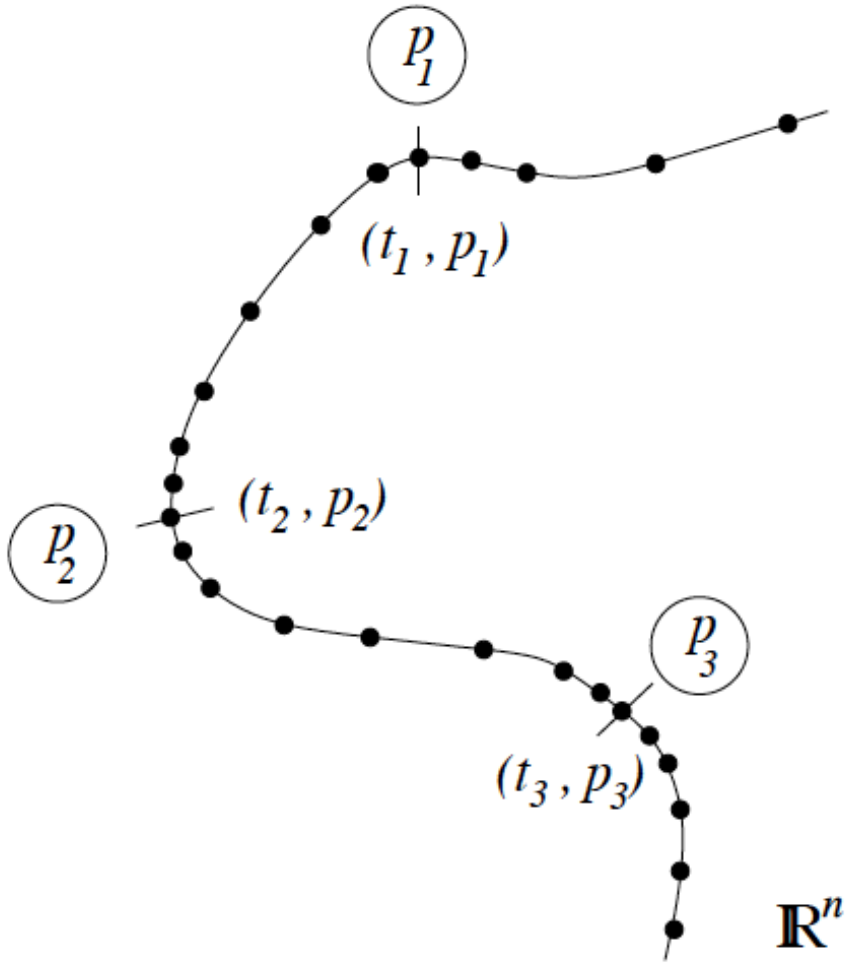
Spectro-Temporal Modulation Events



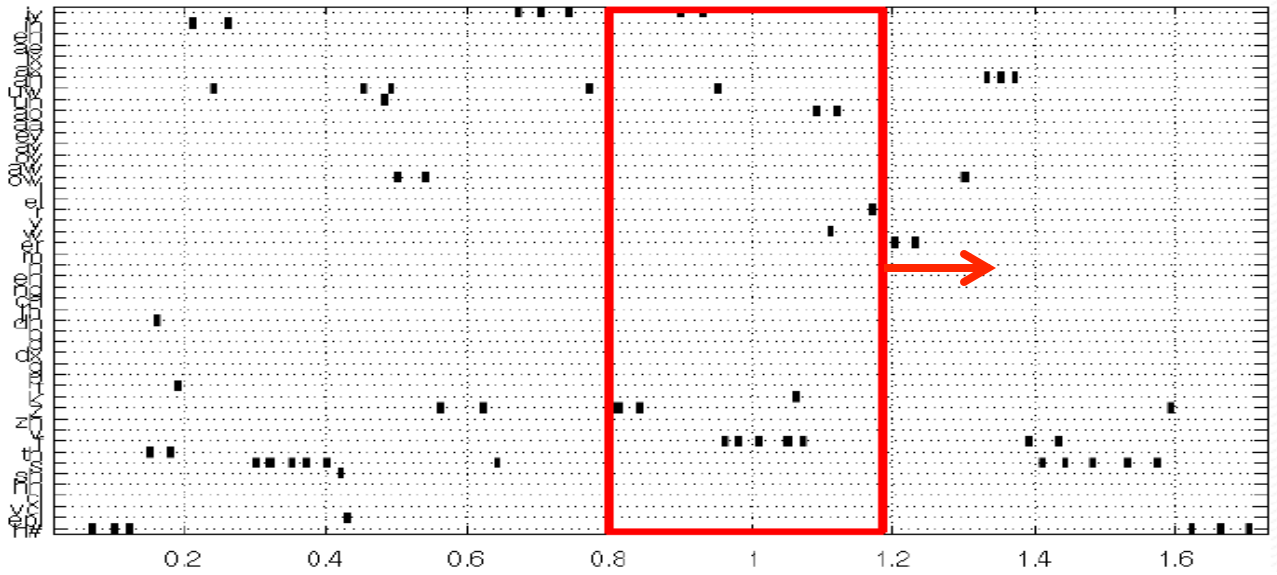


Geometric Interpretation

- **Frames:** Model trajectory as series of points in $\mathbb{R}^n$
- **Points:** Model trajectory as the times of closest approach to each categorical center, each marked with the category identity



Sliding Model Word Detectors



1. Let $\theta_w : \mathbb{R} \rightarrow \{0, 1\}$ be indicator function of word w occurrence
2. Let $R_{t,T} = \{N_{\phi_i}\}$ be the point pattern in the interval $[t, t+T]$
3. Define LLR detector function

$$d_w(t) = \log \int \frac{P(R_{t,T}|T, \theta_w(t)=1)}{P(R_{t,T}|T, \theta_w(t)=0)} P(T|\theta_w(t)=1) dT.$$

Word Model, $P(R_{t,T}|T, \theta_w(t)=1)$

Inhomogeneous Poisson Process Definition

Memoryless point process with feature ϕ_i arrival probability $\lambda_{\phi_i}(t)dt$ in differential time element dt at time t

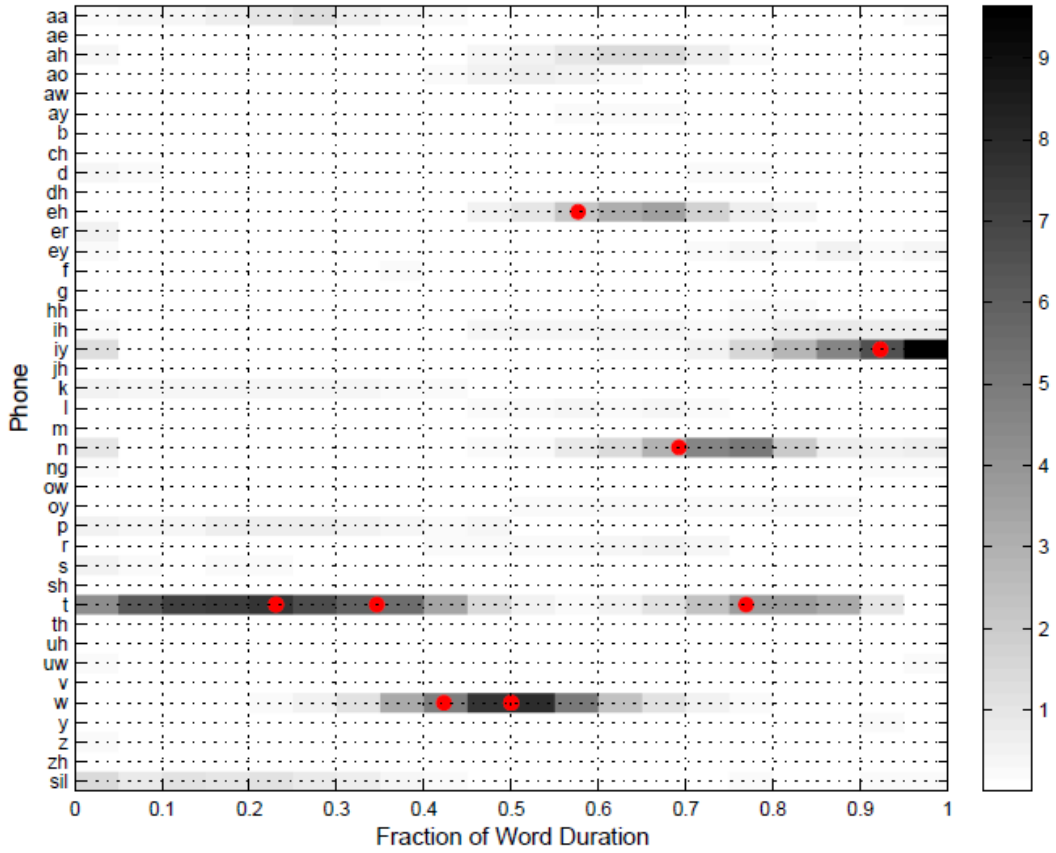
1. Normalize all $t \in R_{t,T}$ to the interval $[0, 1]$, yielding $R' = \{N'_{\phi_i}\}$
2. Assume T-independence of R', independent feature detectors, and inhomogeneous Poisson process model for each feature:

$$P(R_{t,T}|T, \theta_w(t)=1) = \frac{1}{T^{|R_{t,T}|}} \prod_i e^{-\int_0^1 \lambda_{\phi_i}(s)ds} \prod_{s \in N'_{\phi_i}} \lambda_{\phi_i}(s),$$

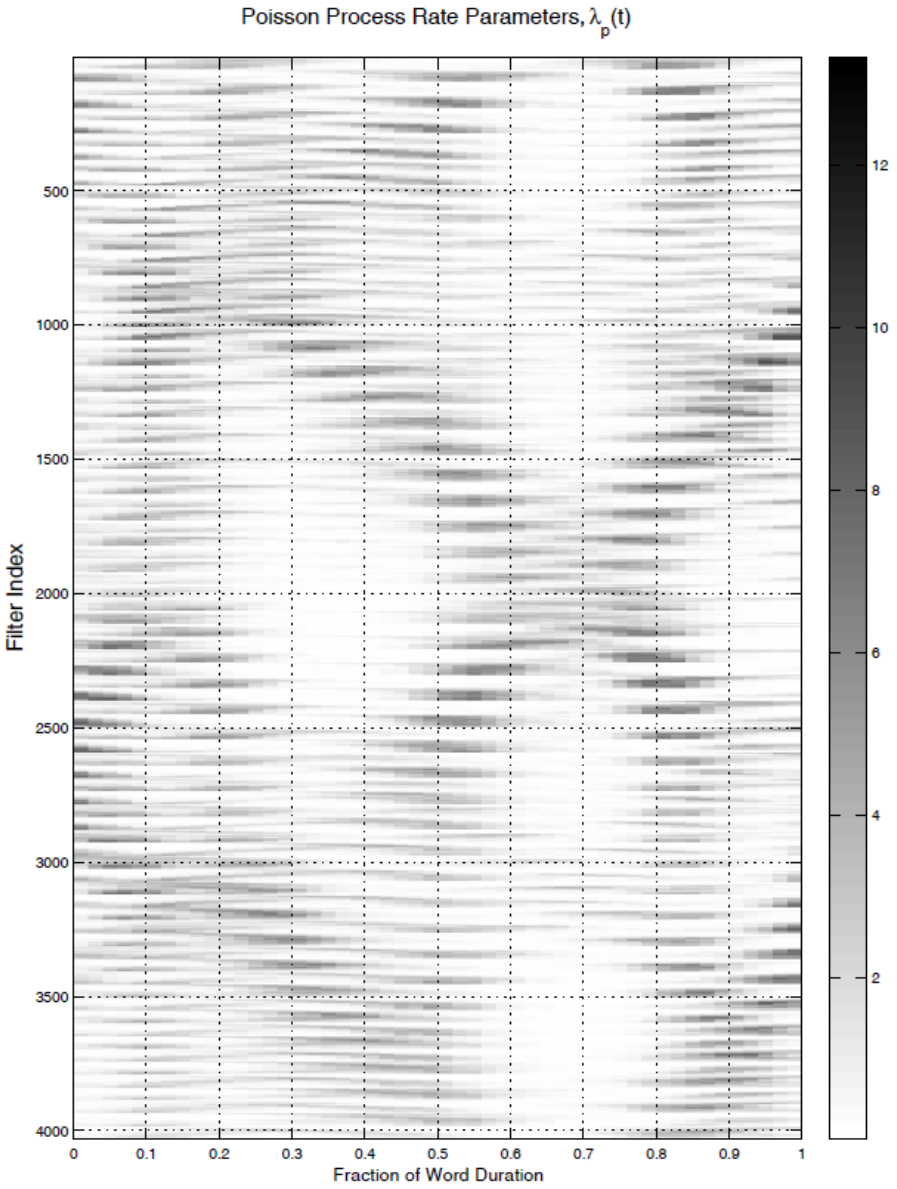
3. Rate functions $\{\lambda_{\phi_i}(t)\}$ are estimated with parametric model or KDE

Example: “twenty” PPM

- Given a word, learn likelihoods of each event type as a function of time in the word



Example: “greasy” PPM



Demonstrated PPM Advantages

- **Robustness:** AURORA2 Robust Digit Recognition Evaluation

Train: Clean, Test: Babble

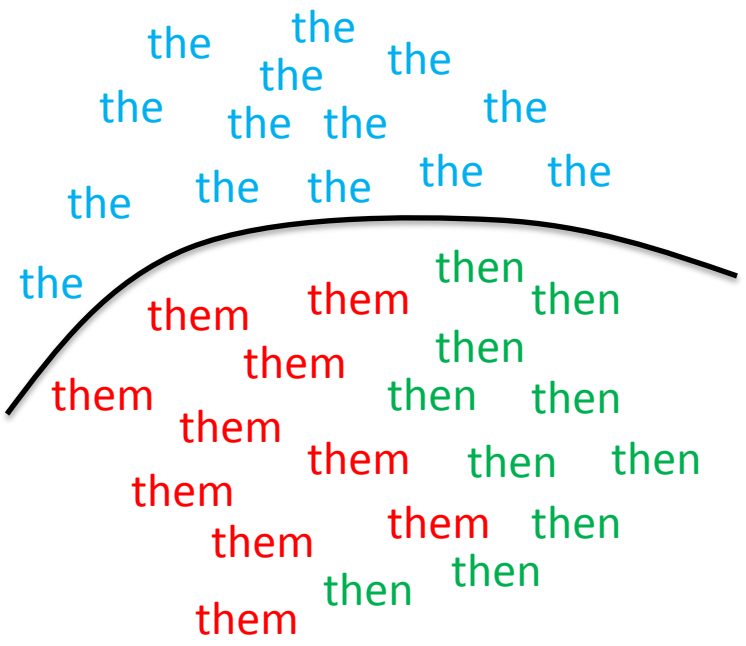
SNR	HMM	PPM
clean	99.0	98.5
20 dB	90.2	93.6
15 dB	73.8	89.7
10 dB	49.4	80.2
5 dB	26.8	62.5
0 dB	9.3	35.8

Less is More:
Huge recognition improvements from modeling only the important parts of the signal

- **Speed:** Run-time linear in *number of events*, allowing keyword search > **500,000X** faster than real time

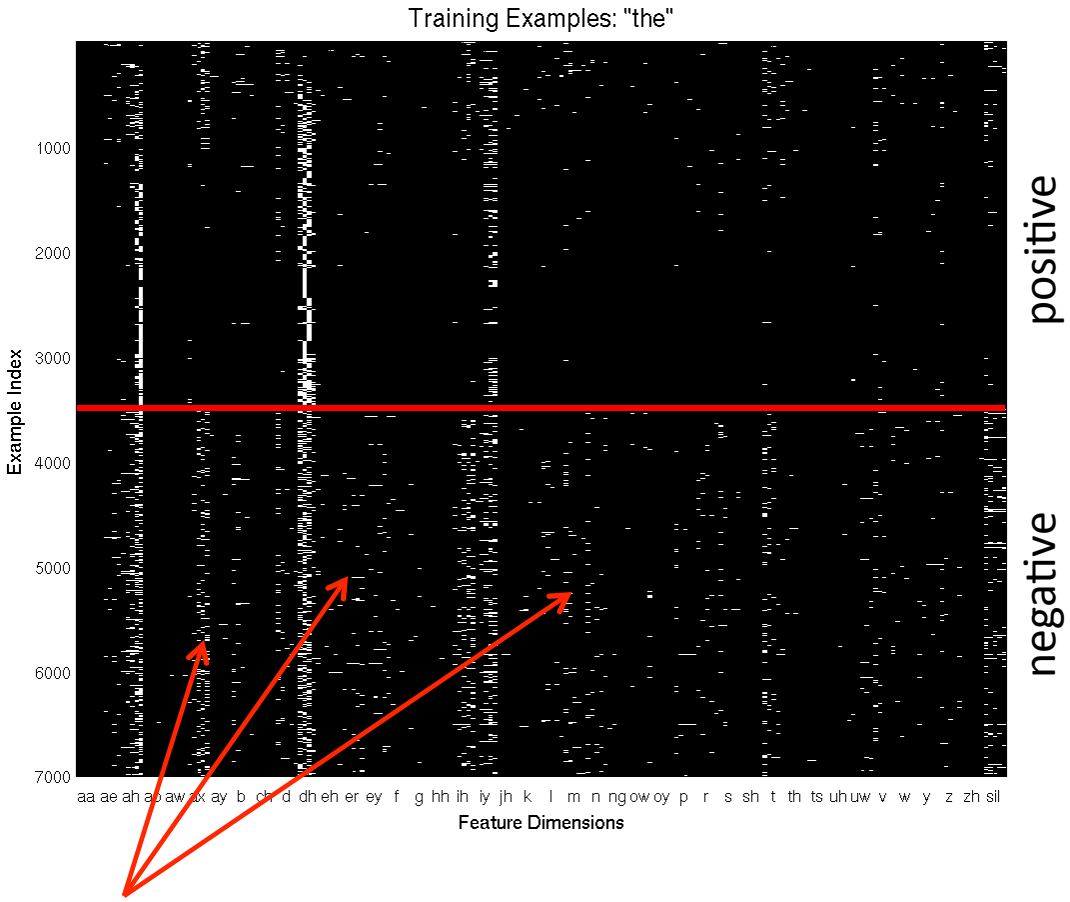
Machine Learning in Word Space

- Imagine words occupying some abstract space (need not be vectors)
- Define a distance metric between "points" in that space
- Train word models with machine learning methods that require only pair-wise distances



Discriminative PPM

- Compute phone events from phonetic posteriorgrams
- Collect positive/negative point patterns for each word **from training lattices**
- Rescore lattice arcs with RLS +RBF word classifiers



Random phone events present in negative examples only

PPM vs IBM Attila

- **Attila:** LDA, VTLN, fMMI, fMLLR, MLLR, bMMI, Quinphones
- **PPM:** MLP monophone events, whole-word classifiers