Speech recognition (briefly)

Chapter 15, Section 6

Outline

◇ Speech as probabilistic inference

◇ Speech sounds

◇ Word pronunciation

◇ Word sequences

Speech as probabilistic inference

It's not easy to wreck a nice beach

Speech signals are noisy, variable, ambiguous

What is the **most likely** word sequence, given the speech signal?
I.e., choose $Words$ to maximize $P(Words|signal)$

$Words$ are the hidden state sequence, $signal$ is the observation sequence

We need to define an acoustic model (sensor model) + language model (transition model).

Phones

All human speech is composed from 40-50 phones, determined by the configuration of articulators (lips, teeth, tongue, vocal cords, air flow)

Form an intermediate level of hidden states between words and signal
$\Rightarrow$ acoustic model = pronunciation model + phone model

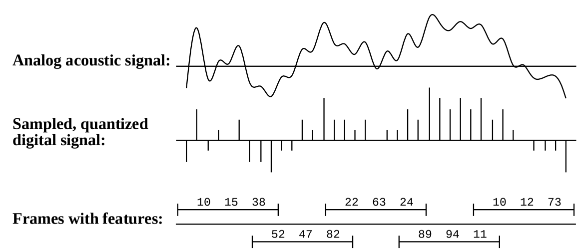
ARPAbet designed for American English

[iy]	beat	[b]	bet	[p]	pet
[ih]	bit	[ch]	Chet	[r]	rat
[ey]	bet	[d]	debt	[s]	set
[ao]	bought	[hh]	hat	[th]	thick
[ow]	boat	[hv]	high	[dh]	that
[er]	Bert	[l]	let	[w]	wet
[ix]	roses	[ng]	sing	[en]	button
⋮	⋮	⋮	⋮	⋮	⋮

E.g., "ceiling" is [s iy l ih ng] / [s iy l ix ng] / [s iy l en]

Speech sounds

Raw signal is the microphone displacement as a function of time; processed into overlapping 30ms frames, each described by features



Frame features are typically formants—peaks in the power spectrum

Phone models

Frame features in $P(features|phone)$ summarized by
– an integer in $[0 \ldots 255]$ (using vector quantization); or
– the parameters of a mixture of Gaussians

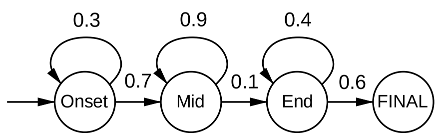
Three-state phones: each phone has three phases (Onset, Mid, End)
E.g., [t] has silent Onset, explosive Mid, hissing End
$\Rightarrow P(features|phone, phase)$

Triphone context: each phone becomes n^2 distinct phones, depending on the phones to its left and right
E.g., [t] in "star" is written [t(s,aa)] (different from "tar"!)

Triphones useful for handling coarticulation effects: the articulators have inertia and cannot switch instantaneously between positions
E.g., [t] in "eighth" has tongue against front teeth

Phone model example

Phone HMM for [m]:



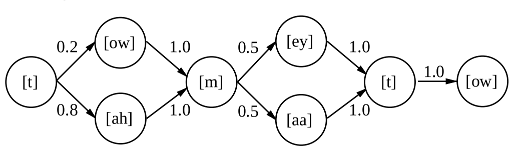
Output probabilities for the phone HMM:

Onset:	Mid:	End:
C1: 0.5	C3: 0.2	C4: 0.1
C2: 0.2	C4: 0.7	C6: 0.5
C3: 0.3	C5: 0.1	C7: 0.4

Word pronunciation models

Each word is described as a distribution over phone sequences

Distribution represented as an HMM transition model



$P([towmeytow]|\text{"tomato"}) = P([towmaatow]|\text{"tomato"}) = 0.1$
$P([tahmeytow]|\text{"tomato"}) = P([tahmaatow]|\text{"tomato"}) = 0.4$

Structure is created manually, transition probabilities learned from data

Isolated words

Phone models + word models fix likelihood $P(e_{1:t}|word)$ for isolated word

$P(word|e_{1:t}) = \alpha P(e_{1:t}|word)P(word)$

Prior probability $P(word)$ obtained simply by counting word frequencies

$P(e_{1:t}|word)$ is computed using the HMM for each word.

Isolated-word dictation systems with training reach 95–99% accuracy

Continuous speech

Not just a sequence of isolated-word recognition problems!
- Adjacent words highly correlated
- Sequence of most likely words $\neq$ most likely sequence of words
- Segmentation: there are few gaps in speech
- Cross-word coarticulation—e.g., "next thing"

Continuous speech systems manage 60–80% accuracy on a good day

Language model

Prior probability of a word sequence is given by chain rule:

$P(w_1 \cdots w_n) = \prod_{i=1}^{n} P(w_i|w_1 \cdots w_{i-1})$

Bigram model:

$P(w_i|w_1 \cdots w_{i-1}) \approx P(w_i|w_{i-1})$

Train by counting all word pairs in a large text corpus

More sophisticated models (trigrams, grammars, etc.) help a little bit

Trigram model

Mark V. Shaney, a program created by Bruce Ellis at Bell Labs
- Used alt.singles as training corpus
- Generated text using tri-gram model (second order model)

Examples:
"As I've commented before, really relating to someone involves standing next to impossible."

"I spent an interesting evening recently with a grain of salt"

Combined HMM

States of the combined language+word+phone model are labelled by
the word we're in + the phone in that word + the phone state in that phone

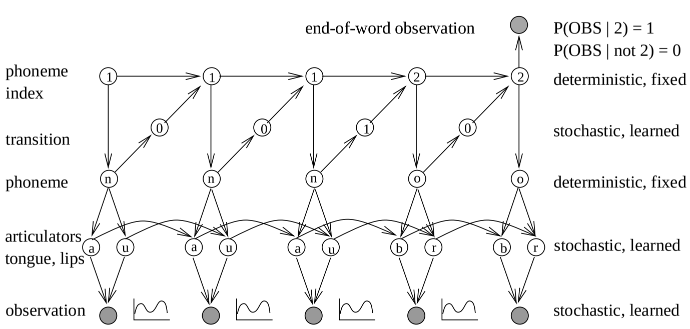
Viterbi algorithm finds the most likely **phone state** sequence

Does segmentation by considering all possible word sequences and boundaries

Doesn't always give the most likely word sequence because
each word sequence is the sum over many state sequences

Jelinek invented a way to find most likely word sequence using A^*
where "step cost" is $-\log P(w_i|w_{i-1})$

DBNs for speech recognition



Also easy to add variables for, e.g., gender, accent, speed.

Summary

Since the mid-1970s, speech recognition has been formulated as probabilistic inference

Evidence = speech signal, hidden variables = word and phone sequences

"Context" effects (coarticulation etc.) are handled by augmenting state

Variability in human speech (speed, timbre, etc., etc.) and background noise make continuous speech recognition in real settings an open problem