Grading

- Class is now big enough for big-class policies
- Late days: 7 total, use whenever
- Grading: Projects out of 10
  - 6 Points: Successfully implemented what we asked
  - 2 Point: 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
- Letter Grades:
  - 10=A, 9=A-, 8=B+, 7=B, 6=B-, 5=C+, lower handled case-by-case
  - Cutoffs at 9.5, 8.5, etc., A+ by discretion

FSA for Lexicon + Bigram LM

Figure from Huang et al page 618

State Model

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, lec[1-mid]ure)
  - Acoustic model depends on clustered phone context
    - But this doesn’t grow the state space

Decoding
Beam Search

- At each time step
  - Start: Beam (collection) \( \nu_t \) of hypotheses \( s \) at time \( t \)
  - For each \( s \) in \( \nu_t \)
    - Compute all extensions \( s' \) at time \( t+1 \)
    - Score \( s' \) from \( s \)
    - Put \( s' \) in \( \nu_{t+1} \) 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

Prefix Trie Encodings

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

LM Score Integration

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

LM Factoring

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

Note that you might have two hypotheses on the beam that differ only in LM context, but are doing the same within-word work
**LM Reweighting**

- Noisy channel suggests
  \[ P(x|w)P(w) \]
- In practice, want to boost LM
  \[ P(x|w)P(w)^a \]
- Also, good to have a “word bonus” to offset LM costs
  \[ P(x|w)P(w)^a|w|^b \]
- 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

**Other Good Ideas**

- When computing emission scores, \( P(x|s) \) depends on only a projection \( \pi(s) \), so use caching
- Beam search is still dynamic programming, so make sure you check for hypotheses that reach the same HMM state (so you can delete the suboptimal one).
- Beams require priority queues, and beam search implementations can get object-heavy. Remember to intern / canonicalize objects when appropriate.

**Training**

**What Needs to be Learned?**

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

**Estimation from Aligned Data**

- What if each time step was labeled with its (context-dependent sub) phone?
  \[ /s/ /ae/ /ae/ /ae/ /ae/ /s/ \]
  \[ x \ x \ x \ x \ x \]
- Can estimate \( P(x/\text{ae}/) \) 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 \( P(x \mid \text{phone}) \) was known?
  - ... and also the correct sequences of words / phones
- Can predict the best alignment of frames to phones
  - “speech lab”
- 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: \( P(s_t | x) \)

Computing Marginals

\[
P(s_t | x) = \frac{P(s_t, x)}{P(x)}
= \frac{\text{sum of all paths through } s_t \text{ at } t}{\text{sum of all paths}}
\]

Forward Scores

\[
\alpha_t(s_t) = \max_{s_{t-1}} \eta_{t-1}(s_{t-1}) \alpha_{t-1}(s_{t-1}, s_t)
\]

\[
\alpha_t(s_t) = \sum_{s_{t-1}} \omega_{t-1}(s_{t-1}) \alpha_{t-1}(s_{t-1}, s_t)
\]

Backward Scores

\[
\beta_t(s_t) = \sum_{s_{t+1}} \eta_t(s_{t+1}) \beta_{t+1}(s_{t+1})
\]
Total Scores

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

$P(x) = \sum_{s_t} \alpha_t(s_t) \beta_t(s_t)$

$= \nu_t^n(\text{start})$

$= \nu_0(\text{start})$

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