Lecture 7

LVCSR Training and Decoding (Part A)

Michael Picheny, Bhuvana Ramabhadran, Stanley F. Chen, Markus Nussbaum-Thom

Watson Group
IBM T.J. Watson Research Center
Yorktown Heights, New York, USA
{picheny,bhuvana,stanchen,nussbaum}@us.ibm.com

Mar 9, 2016

Administrivia

- Lab 2
  - Handed back this lecture or next.
- Lab 3 extension
  - Due nine days from now (Friday, Mar. 18) at 6pm.
- Visit to IBM Watson Astor Place
  - April 1, 11am. (About 1h?)
- Spring recess next week; no lecture.

Feedback

- Clear (9)
- Pace: fast (1)
- Muddiest: context models (3); diagonal GMM splitting (2); arcs v. state probs (1)
- Comments (2+ votes):
  - Nice song (4)
  - Hard to see chalk on blackboard (3)
  - Lab 3 better than Lab 2 (2)
  - Miss Michael on right (1); prefer Michael on left (1)

The Big Picture

- Weeks 1–4: Signal Processing, Small vocabulary ASR.
- Weeks 5–8: Large vocabulary ASR.
  - Week 5: Language modeling (for large vocabularies).
  - Week 6: Pronunciation modeling — acoustic modeling for large vocabularies.
  - Week 7, 8: Training, decoding for large vocabularies.
- Weeks 9–13: Advanced topics.

Outline

- Part I: The LVCSR acoustic model.
- Part II: Acoustic model training for LVCSR.
- Part III: Decoding for LVCSR (inefficient).
  - Part IV: Introduction to finite-state transducers.
- Part V: Search (Lecture 8).
  - Making decoding for LVCSR efficient.

Part I

The LVCSR Acoustic Model

What is LVCSR?

Demo from https://speech-to-text-demo.mybluemix.net/

What is LVCSR?

- Large vocabulary Continuous Speech Recognition.
  - Phone-based modeling *vs.* word-based modeling.
- Continuous.
  - No pauses between words.

How do you evaluate such an LVCSR system?

Ground Truth	Hello how can I help you today
ASR	Hello ___ can EYE help you TO today

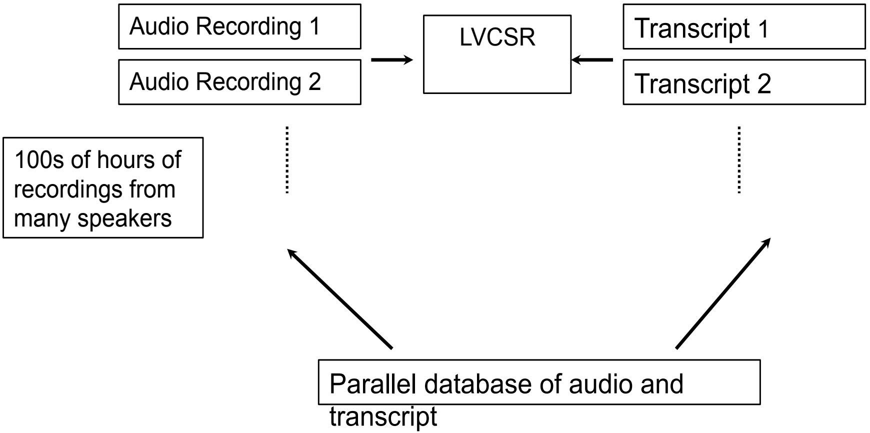
Deletion Error

Substitution Error

Insertion Error

$$WER = \frac{\#Ins + \#Sub + \#Del}{\#Words}$$

What do we have to begin training an LVCSR system?



Lexicon or dictionary

The Fundamental Equation of ASR

$$
\begin{aligned}
w^* &= \arg\max_{\omega} P(\omega|\mathbf{x}) && (1) \\
&= \arg\max_{\omega} \frac{P(\omega)P(\mathbf{x}|\omega)}{P(\mathbf{x})} && (2) \\
&= \arg\max_{\omega} P(\omega)P(\mathbf{x}|\omega) && (3)
\end{aligned}
$$

- w^* — best sequence of words (class)
- $\mathbf{x}$ — sequence of acoustic vectors
- $P(\mathbf{x}|\omega)$ — acoustic model.
- $P(\omega)$ — language model.

The Acoustic Model: Small Vocabulary

$$
\begin{aligned}
P_\omega(\mathbf{x}) &= \sum_A P_\omega(\mathbf{x}, A) = \sum_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (4)\\
&=\approx \max_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (5)\\
&= \max_A \prod_{t=1}^{T} P(a_t) \prod_{t=1}^{T} P(\vec{x}_t|a_t) && (6)\\
\log P_\omega(\mathbf{x}) &= \max_A \left[\sum_{t=1}^{T} \log P(a_t) + \sum_{t=1}^{T} \log P(\vec{x}_t|a_t) \right] && (7)\\
P(\vec{x}_t|a_t) &= \sum_{m=1}^{M} \lambda_{a_t,m} \prod_{\text{dim } d}^{D} \mathcal{N}(x_{t,d}; \mu_{a_t,m,d}, \sigma_{a_t,m,d}) && (8)
\end{aligned}
$$

The Acoustic Model: Large Vocabulary

$$
\begin{aligned}
P_\omega(\mathbf{x}) &= \sum_A P_\omega(\mathbf{x}, A) = \sum_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (9) \\
&=\approx \max_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (10) \\
&= \max_A \prod_{t=1}^{T} P(a_t) \prod_{t=1}^{T} P(\vec{x}_t|a_t) && (11) \\
\log P_\omega(\mathbf{x}) &= \max_A \left[\sum_{t=1}^{T} \log P(a_t) + \sum_{t=1}^{T} \log P(\vec{x}_t|a_t) \right] && (12) \\
P(\vec{x}_t|a_t) &= \sum_{m=1}^{M} \lambda_{a_t,m} \prod_{\text{dim } d}^{D} \mathcal{N}(x_{t,d}; \mu_{a_t,m,d}, \sigma_{a_t,m,d}) && (13)
\end{aligned}
$$

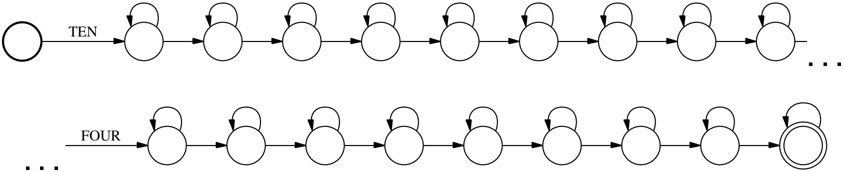
What Has Changed?

- The HMM.
  - Each alignment A describes a path through an HMM.
- Its parameterization.
  - In $P(\vec{x}_t | a_t)$, how many GMM's to use? (Share between HMM's?)

Describing the Underlying HMM

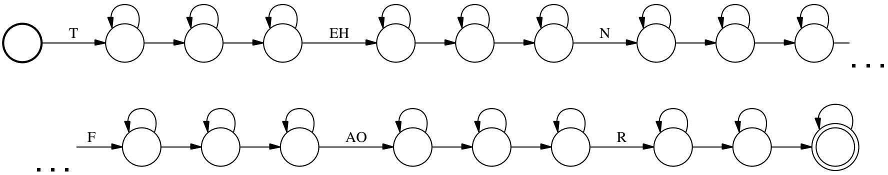
- Fundamental concept: how to map a word (or baseform) sequence to its HMM.
  - In training, map reference transcript to its HMM.
  - In decoding, glue together HMM's for all allowable word sequences.

The HMM: Small Vocabulary



- One HMM per word.
- Glue together HMM for each word in word sequence.

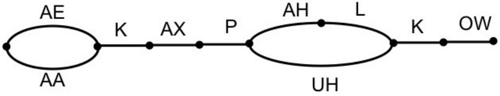
The HMM: Large Vocabulary



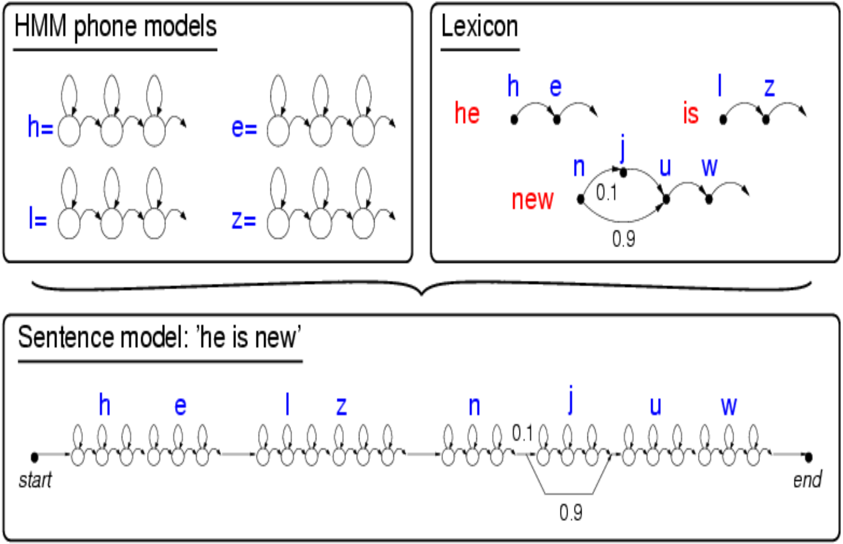
- One HMM per phone.
- Glue together HMM for each phone in phone sequence.
  - Map word sequence to phone sequence using baseform dictionary.
    - The rain in Spain falls . . .
    - DH AX | R EY N | IX N | S P EY N | F AA L Z | . . .

An Example: Word to HMM

acapulco AE K AX P AH L K OW
acapulco AA K AX P UH K OW

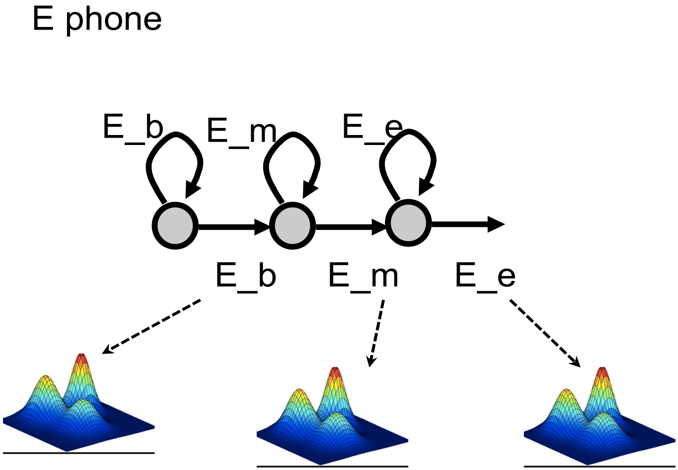


An Example: Words to HMMs



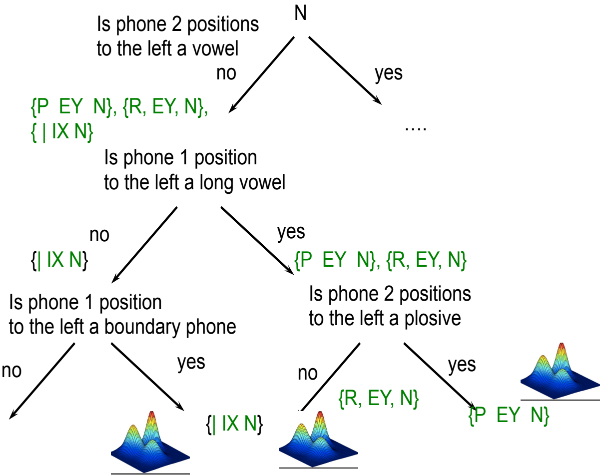
An Example: Word to HMM to GMMs

A set of arcs in a Markov model are tied to one another if they are constrained to have identical output distributions



Now, in this example . . .

- The rain in Spain falls . . .
- DH AX | R EY N | IX N | S P EY N | F AA L Z | . . .



38

I Still Don't See What's Changed

- HMM topology typically doesn't change.
- HMM parameterization changes.

Parameterization

- Small vocabulary.
  - One GMM per state (three states per phone).
  - No sharing between phones in different words.
- Large vocabulary, context-independent (CI).
  - One GMM per state.
  - *Tying* between phones in different words.
- Large vocabulary, context-dependent (CD).
  - Many GMM's per state; GMM to use depends on phonetic context.
  - Tying between phones in different words.

Context-Dependent Parameterization

- Each phone HMM state has its own decision tree.
  - Decision tree asks questions about phonetic context. (Why?)
  - One GMM per leaf in the tree. (Up to 200+ leaves/tree.)
- How will tree for first state of a phone tend to differ …
  - From tree for last state of a phone?
- Terminology.
  - *triphone* model — ± 1 phones of context.
  - *quinphone* model — ± 2 phones of context.

0	Z IY R OW
1	W AA N
2	T UW
3	TH R IY
4	F AO R
5	F AY V
6	S IH K S
7	S EH V AX N
8	EY T
9	N AY N

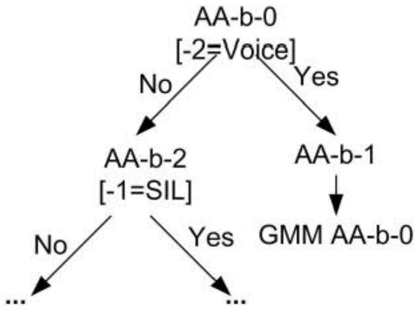
- Examples of “0” will affect models for “3” and “4”
- Useful in large vocabulary systems (why?)

A Real-Life Tree

In practice:

- These trees are built on one-third of a phone, i.e., the three states of the HMM for a phone correspond to the beginning, middle and end of a phone.
- Context-independent versions
- Context-dependent versions

Another Sample Tree



Pop Quiz

- System description:
  - 1000 words in lexicon; average word length = 5 phones.
  - There are 50 phones; each phone HMM has three states.
  - Each decision tree contains 100 leaves on average.
- How many GMM's are there in:
  - A small vocabulary system (word models)?
  - A CI large vocabulary system?
  - A CD large vocabulary system?

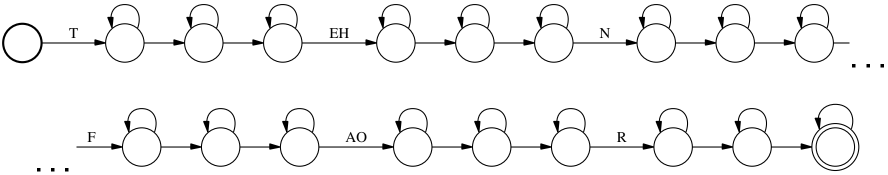
Context-Dependent Phone Models

- Typical model sizes:

type	HMM	GMM's/ state	GMM's	Gaussians
word	per word	1	10–500	100–10k
CI phone	per phone	1	$\sim$150	1k–3k
CD phone	per phone	1–200	1k–10k	10k–300k

- 39-dimensional feature vectors $\Rightarrow$ $\sim$80 parameters/Gaussian.
- Big models can have tens of millions of parameters.

Any Questions?



- Given a word sequence, you should understand how to ...
  - Layout the corresponding HMM topology.
  - Determine which GMM to use at each state, for CI and CD models.

What About Transition Probabilities?

- This slide only included for completeness.
- Small vocabulary.
  - One set of transition probabilities per state.
  - No sharing between phones in different words.
- Large vocabulary.
  - One set of transition probabilities per state.
  - Sharing between phones in different words.
- What about context-dependent transition modeling?

Recap

- Main difference between small vocabulary and large vocabulary:
  - Allocation of GMM's.
  - Sharing GMM's between words: needs less GMM's.
  - Modeling context-dependence: needs more GMM's.
  - Hybrid allocation is possible.
- Training and decoding for LVCSR.
  - In theory, any reason why small vocabulary techniques won't work?
  - In practice, yikes!

Points to Ponder

- Why deterministic mapping?
  - DID YOU $\Rightarrow$ D IH D JH UW
  - The area of *pronunciation modeling*.
- Why decision trees?
  - Unsupervised clustering.

Part II

Acoustic Model Training for LVCSR

Small Vocabulary Training — Lab 2

- Phase 1: Collect underpants.
  - Initialize all Gaussian means to 0, variances to 1.
- Phase 2: Iterate over training data.
  - For each word, train associated word HMM . . .
  - On all samples of that word in the training data . . .
  - Using the Forward-Backward algorithm.
- Phase 3: Profit!

Large Vocabulary Training

- What's changed going to LVCSR?
  - Same HMM topology; just more Gaussians and GMM's.
- Can we just use the same training algorithm as before?

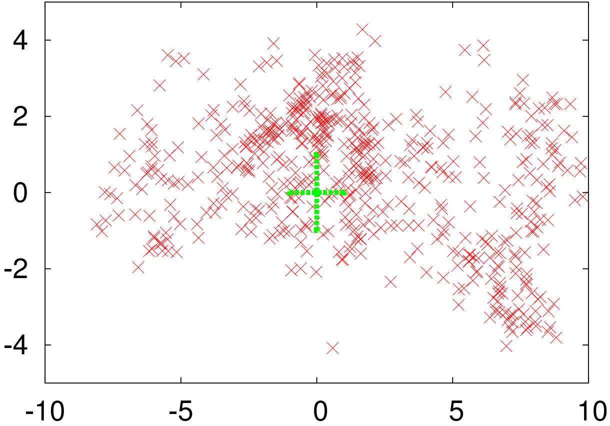
Where Are We?

Flat or Random Start

- Why does this work for small models?
  - We believe there's a huge global minimum . . .
  - In the "middle" of the parameter search space.
  - With a neutral starting point, we're apt to fall into it.
  - (Who knows if this is actually true.)
- Why doesn't this work for large models?

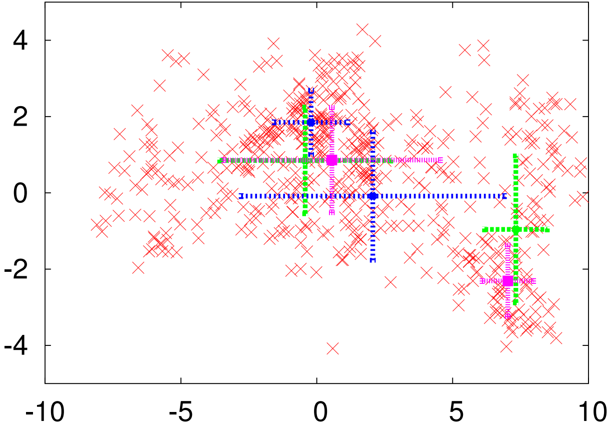
Training a Mixture of Two 2-D Gaussians

- Flat start?
  - Initialize mean of each Gaussian to 0, variance to 1.



Training a Mixture of Two 2-D Gaussians

- Random seeding?
  - Picked 8 random starting points ⇒ 3 different optima.



Training Hidden Models

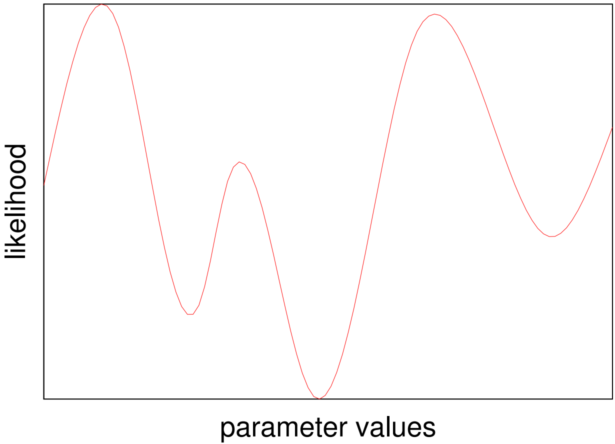
- (MLE) training of models with hidden variables has local minima.
- What are the hidden variables in ASR?
  - *i.e.*, what variables are in our model . . .
  - That are not observed.

How To Spot Hidden Variables

$$
\begin{aligned}
P_\omega(\mathbf{x}) &= \sum_A P_\omega(\mathbf{x}, A) = \textstyle\sum_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (14) \\
&=\approx \max_A P_\omega(A) \times P_\omega(\mathbf{x}|A) && (15) \\
&= \max_A \prod_{t=1}^{T} P(a_t) \prod_{t=1}^{T} P(\vec{x}_t|a_t) && (16) \\
\log P_\omega(\mathbf{x}) &= \max_A \left[\sum_{t=1}^{T} \log P(a_t) + \sum_{t=1}^{T} \log P(\vec{x}_t|a_t) \right] && (17) \\
P(\vec{x}_t|a_t) &= \textstyle\sum_{m=1}^{M} \lambda_{a_t,m} \displaystyle\prod_{\text{dim } d}^{D} \mathcal{N}(x_{t,d}; \mu_{a_t,m,d}, \sigma_{a_t,m,d}) && (18)
\end{aligned}
$$

Gradient Descent and Local Minima

- EM training does hill-climbing/gradient descent.
  - Finds "nearest" optimum to where you started.



What To Do?

- Insight: If we know the "correct" hidden values for a model:
  - *e.g.*, which arc and which Gaussian for each frame ...
  - Training is easy! (No local minima.)
  - Remember Viterbi training given fixed alignment in Lab 2.
- Is there a way to guess the correct hidden values for a large model?

Bootstrapping Alignments

- Recall that all of our acoustic models, from simple to complex:
  - Generally use the same HMM topology!
  - (All that differs is how we assign GMM's to each arc.)
- Given an alignment (from arc/phone states to frames) for simple model . . .
  - It is straightforward to compute analogous alignment for complex model!

Bootstrapping Big Models From Small

- Recipe:
  - Start with model simple enough that flat start works.
  - Iteratively build more and more complex models . . .
  - By using last model to seed hidden values for next.
- Need to come up with sequence of successively more complex models . . .
  - With related hidden structure.

How To Seed Next Model From Last

- Directly via hidden values, *e.g.*, alignment.
  - *e.g.*, *single-pass retraining*.
  - Can be used between very different models.
- Via parameters.
  - Seed parameters in complex model so that ...
  - Initially, will yield same/similar alignment as in simple model.
  - *e.g.*, moving from CI to CD GMM's.

Bootstrapping Big Models From Small

- Recurring motif in acoustic model training.
- The reason why state-of-the-art systems . . .
  - Require many, many training passes, as you will see.
- Recipes handed down through the generations.
  - Discovered via sweat and tears.
  - Art, not science.
  - But no one believes these find global optima . . .
  - Even for small problems.

Overview of Training Process

- Build CI single Gaussian model from flat start.
- Use CI single Gaussian model to seed CI GMM model.
- Build phonetic decision tree (using CI GMM model to help).
- Use CI GMM model to seed CD GMM model.

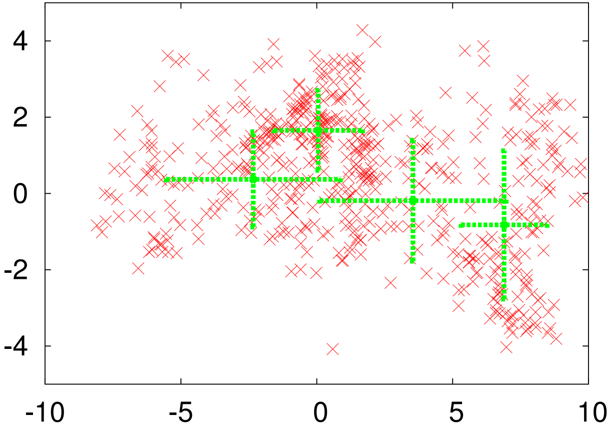
Where Are We?

Case Study: Training a GMM

- Recursive mixture splitting.
  - A sequence of successively more complex models.
  - Perturb means in opposite directions; same variance; Train.
  - (Discard Gaussians with insufficient counts.)
- *k*-means clustering.
  - Seed means in one shot.

Mixture Splitting Example

- Split each Gaussian in two ($\pm 0.2 \times \vec{\sigma}$)



Applying Mixture Splitting in ASR

- Recipe:
  - Start with model with 1-component GMM's (à la Lab 2).
  - Split Gaussians in each output distribution simultaneously.
  - Do many iterations of FB.
  - Repeat.
- Real-life numbers:
  - Five splits spread within 30 iterations of FB.

Another Way: Automatic Clustering

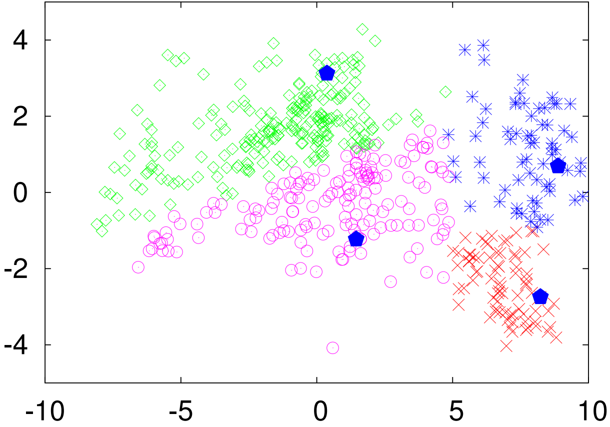
- Use unsupervised clustering algorithm to find clusters (k-Means Clustering)
- Given clusters . . .
  - Use cluster centers to seed Gaussian means.
  - FB training.
  - (Discard Gaussians with insufficient counts.)

k-Means Clustering

- Select desired number of clusters k.
- Choose k data points randomly.
  - Use these as initial cluster centers.
- “Assign” each data point to nearest cluster center.
- Recompute each cluster center as ...
  - Mean of data points “assigned” to it.
- Repeat until convergence.

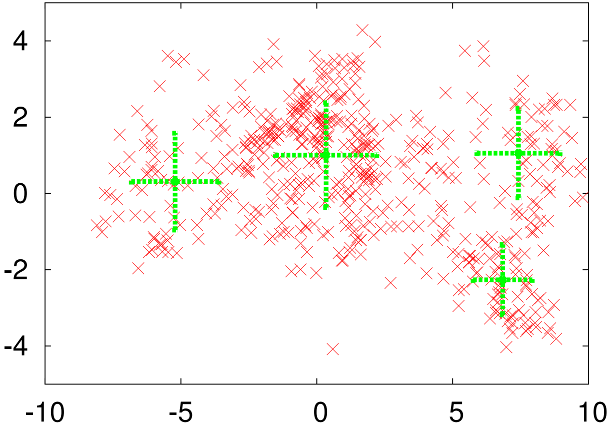
k-Means Example

- Pick random cluster centers; assign points to nearest center.

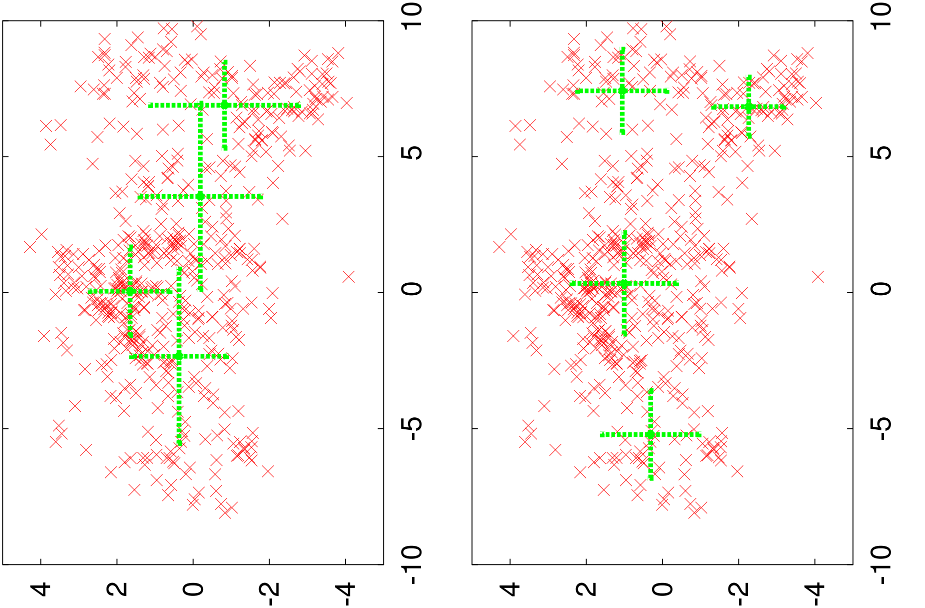


k-Means Example

- Use centers as means of Gaussians; train, yep.



The Final Mixtures, Splitting *vs.* *k*-Means



Technical Aside: k-Means Clustering

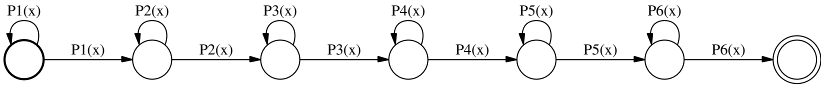
- When using Euclidean distance ...
- k-means clustering is equivalent to ...
  - Seeding Gaussian means with the k initial centers.
  - Doing Viterbi EM update, keeping variances constant.

Applying k-Means Clustering in ASR

- To train each GMM, use k-means clustering ...
  - On what data? Which frames?
- Huh?
  - How to decide which frames *align* to each GMM?
- This issue is evaded for mixture splitting.
  - Can we avoid it here?

Forced Alignment

- Viterbi algorithm.
  - Finds most likely alignment of HMM to data.



frame	0	1	2	3	4	5	6	7	8	9	10	11	12
arc	P_1	P_1	P_1	P_2	P_3	P_4	P_4	P_5	P_5	P_5	P_5	P_6	P_6

- Need existing model to create alignment. (Which?)

Recap

- You can use single Gaussian models to seed GMM models.
  - Mixture splitting: use c-component GMM to seed $2c$-component GMM.
  - k-means: use single Gaussian model to find alignment.
- Both of these techniques work about the same.
  - Nowadays, we primarily use mixture splitting.

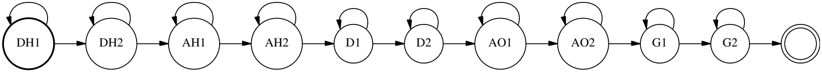
Where Are We?

What Do We Need?

- For each tree/phone state . . .
  - List of frames/feature vectors associated with that tree.
  - (This is the data we are optimizing the likelihood of.)
  - For each frame, the phonetic context.
- A list of candidate questions about the phonetic context.
  - Ask about phonetic concepts; *e.g.*, vowel or consonant?
  - Expressed as list of phones in set.
  - Allow same questions to be asked about each phone position.
  - Handed down through the generations.

Training Data for Decision Trees

- Forced alignment/Viterbi decoding!
- Where do we get the model to align with?
  - Use CI phone model or other pre-existing model.



frame	0	1	2	3	4	5	6	7	8	9	···
arc	DH_1	DH_2	AH_1	AH_2	D_1	D_1	D_2	D_2	D_2	AO_1	···

Building the Tree

- A set of events $\{(\vec{x}_i, p_L, p_R)\}$ (possibly subsampled).
- Given current tree:
  - Choose question of the form ...
  - "*Does the phone in position j belong to the set q?*" ...
  - That optimizes $\prod_i P(\vec{x}_i | \text{leaf}(p_L, p_R))$ ...
  - Where we model each leaf using a single Gaussian.
- Can efficiently build whole level of tree in single pass.
- See Lecture 6 slides and readings for the gory details.

Seeding the Context-Dependent GMM's

- Context-independent GMM's: one GMM per phone state.
- Context-dependent GMM's: l GMM's per phone state.
- How to seed context-dependent GMM's?
  - *e.g.*, so that initial alignment matches CI alignment?

Where Are We?

Where Are We?

4 Details

- Maximum Likelihood Training?
- Viterbi vs. Non-Viterbi Training
- Graph Building

The Original Story, Small Vocabulary

- One HMM for each word; flat start.
- Collect all examples of each word.
  - Run FB on those examples to do maximum likelihood training of that HMM.

The New Story

- One HMM for each word sequence!?
  - But tie parameters across HMM's!
- Do complex multi-phase training.
- Are we still doing anything resembling maximum likelihood training?

Maximum Likelihood Training?

- Regular training iterations (FB, Viterbi EM).
  - Increase (Viterbi) likelihood of data.
- Seeding last model from next model.
  - Mixture splitting.
  - CI $\Rightarrow$ CD models.
- (Decision-tree building.)

Maximum Likelihood Training?

- Just as LM's need to be smoothed or *regularized*.
  - So do acoustic models.
  - Prevent extreme likelihood values (*e.g.*, 0 or ∞).
- ML training maximizes training data likelihood.
  - We actually want to optimize test data likelihood.
  - Let's call the difference the *overfitting penalty*.
- The overfitting penalty tends to increase as ...
  - The number of parameters increase and/or ...
  - Parameter magnitudes increase.

Regularization/Capacity Control

- Limit size of model.
  - Will training likelihood continue to increase as model grows?
  - Limit components per GMM.
  - Limit number of leaves in decision tree, *i.e.*, number of GMM's.
- Variance flooring.
  - Don't let variances go to 0 $\Rightarrow$ infinite likelihood.

Where Are We?

4 Details

- Maximum Likelihood Training?
- Viterbi vs. Non-Viterbi Training
- Graph Building

Two Types of Updates

- "Full" EM.
  - Compute true posterior of each hidden configuration.
- Viterbi EM.
  - Use Viterbi algorithm to find most likely hidden configuration.
  - Assign posterior of 1 to this configuration.
- Both are valid updates; instances of generalized EM.

Examples

- Training GMM's.
  - Mixture splitting *vs.* k-means clustering.
- Training HMM's.
  - Forward-backward *vs.* Viterbi EM (Lab 2).
- Everywhere you do a forced alignment.
  - Refining the reference transcript.
  - What is non-Viterbi version of decision-tree building?

When To Use One or the Other?

- Which version is more expensive computationally?
  - Optimization: need not realign every iteration.
- Which version finds better minima?
- If posteriors are very sharp, they do almost the same thing.
  - Remember example posteriors in Lab 2?
- Rule of thumb:
  - When you're first training a "new" model, use full EM.
  - Once you're "locked in" to an optimum, Viterbi is fine.

Where Are We?

4 Details

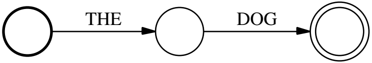
- Maximum Likelihood Training?
- Viterbi vs. Non-Viterbi Training
- Graph Building

Building HMM's For Training

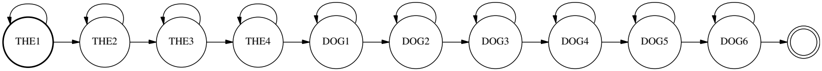
- When doing Forward-Backward on an utterance ...
  - We need the HMM corresponding to the reference transcript.
- Can we use the same techniques as for small vocabularies?

Word Models

- Reference transcript

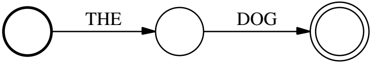


- Replace each word with its HMM

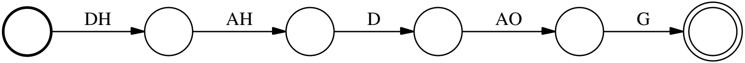


Context-Independent Phone Models

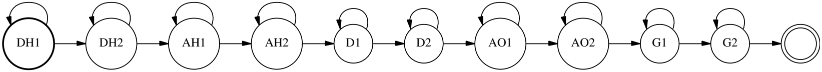
- Reference transcript



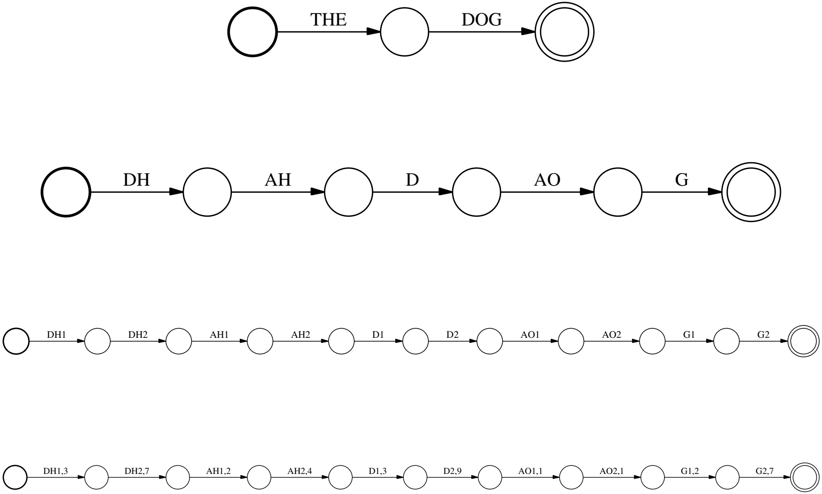
- Pronunciation dictionary.
  - Maps each word to a sequence of phonemes.



- Replace each phone with its HMM



Context-Dependent Phone Models



The Pronunciation Dictionary

- Need pronunciation of *every* word in training data.
  - Including pronunciation variants

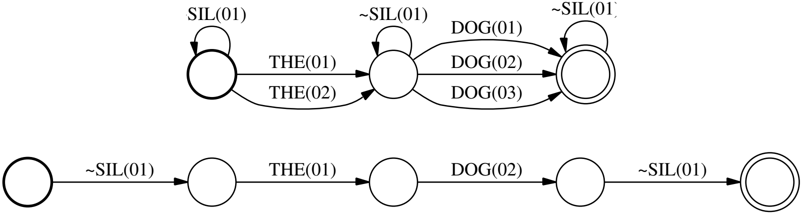
    ```
    THE(01)    DH  AH
    THE(02)    DH  IY
    ```
  - Listen to data?
  - Use automatic spelling-to-sound models?
- Why not consider multiple baseforms/word for word models?

But Wait, It's More Complicated Than That!

- Reference transcripts are created by humans . . .
  - Who, by their nature, are *human* (*i.e.*, fallible)
- Typical transcripts don't contain everything an ASR system wants.
  - Where silence occurred; noises like coughs, door slams, etc.
  - Pronunciation information, *e.g.*, was THE pronounced as `DH UH` or `DH IY`?

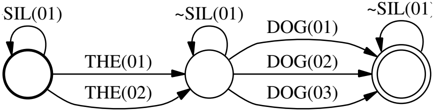
Pronunciation Variants, Silence, and Stuff

- How can we produce a more "complete" reference transcript?
- Viterbi decoding!
  - Build HMM accepting all word (HMM) sequences consistent with reference transcript.
  - Compute best path/word HMM sequence.
  - Where does this initial acoustic model come from?



Another Way

- Just use the whole expanded graph during training.



- The problem: how to do context-dependent phone expansion?
    - Use same techniques as in building graphs for decoding.

Where Are We?

Prerequisites

- Audio data with reference transcripts.
- What two other things?

The Training Recipe

- Find/make baseforms for all words in reference transcripts.
- Train single Gaussian models (flat start; many iters of FB).
- Do mixture splitting, say.
  - Split each Gaussian in two; do many iterations of FB.
  - Repeat until desired number of Gaussians per mixture.
- (Use initial system to refine reference transcripts.)
  - Select pronunciation variants, where silence occurs.
  - Do more FB training given refined transcripts.
- Build phonetic decision tree.
  - Use CI model to align training data.
- Seed CD model from CI; train using FB or Viterbi EM.
  - Possibly doing more mixture splitting.

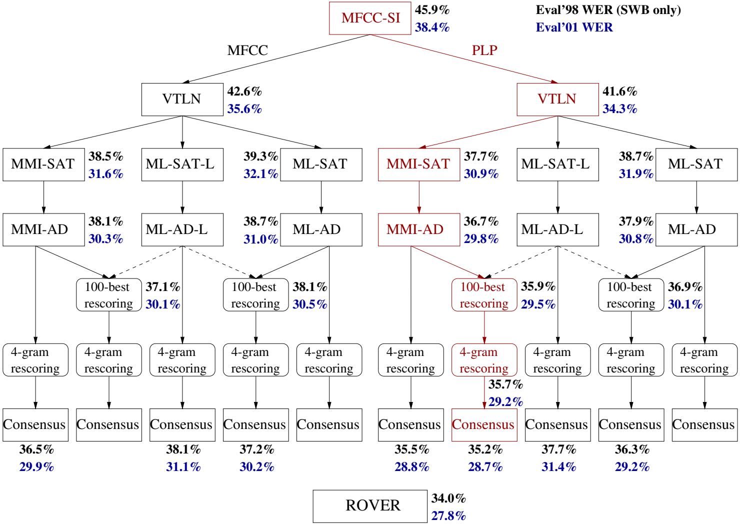
How Long Does Training Take?

- It's a secret.
- We think in terms of *real-time factor*.
  - How many hours does it take to process one hour of speech?

Whew, That Was Pretty Complicated!

- Adaptation (VTLN, fMLLR, mMLLR)
- Discriminative training (LDA, MMI, MPE, fMPE)
- Model combination (cross adaptation, ROVER)
- Iteration.
  - Repeat steps using better model for seeding.
  - Alignment is only as good as model that created it.

Things Can Get Pretty Hairy



Recap: Acoustic Model Training for LVCSR

- Take-home messages.
  - Hidden model training is fraught with local minima.
  - Seeding more complex models with simpler models helps avoid terrible local minima.
  - People have developed many recipes/heuristics to try to improve the minimum you end up in.
  - Training is insanely complicated for state-of-the-art research models.
- The good news . . .
  - I just saved a bunch on money on my car insurance by switching to GEICO.

Outline

- Part I: The LVCSR acoustic model.
- Part II: Acoustic model training for LVCSR.
- Part III: Decoding for LVCSR (inefficient).
  - Part IV: Introduction to finite-state transducers.
- Part V: Search (Lecture 8).
  - Making decoding for LVCSR efficient.

Course Feedback

1. Was this lecture mostly clear or unclear? What was the muddiest topic?
2. Other feedback (pace, content, atmosphere)?