Speaker Adaptation in Sphinx 3.x and CALO

David Huggins-Daines
dhuggins@cs.cmu.edu

Overview

- Background of speaker adaptation
- Types of speaker adaptation tasks
- Goal of current developments in Sphinx and CALO projects
- Methods for adaptation
- SphinxTrain adaptation tools and results
- Plan of development

Acoustic Modeling

- Speaker-Dependent Models
  - Widely used; high accuracy for restricted tasks
  - Impractical for LVCSR due to amount of training data required - must be retrained for every user
- Speaker-Independent Models
  - Trained from a broad selection of speakers intended to cover the space of potential users
- Speaker-Specific Models
  - Knowing some information (e.g. gender, dialect) about the speaker can allow us to select from among multiple SI models.

Speaker Adaptation

- A small amount of observed data from an individual speaker is used to improve a speaker-independent model
  - Much less data than required for SD training
- Humans are really good at this
  - Acoustic adaptation occurs unconsciously within the first few seconds
- For ASR, we would like to:
  - Adapt rapidly to new speakers
  - Asymptotically approximate SD performance
  - Do all this in unsupervised fashion

Adaptation Data

- The adaptation data set is much smaller than a speaker-dependent training set
  - Less than 1 minute of data is required
  - Many experiments use 3-10 phonetically balanced “rapid adaptation” sentences

Supervised and Unsupervised Adaptation

- Like acoustic model training, the adaptation task can be done in supervised (with a transcript) or unsupervised (no transcript) fashion
- Unsupervised adaptation is straightforward since we assume the existence of a baseline model
  - Decode and align the adaptation data with the baseline model, then use this transcription to do adaptation.
  - This may not work well if recognition accuracy is poor
  - Some adaptation methods are more robust than others
  - Confidence measures for the adaptation data

Incremental and Batch Adaptation

- Batch adaptation
  - Adaptation data is predetermined
  - Often obtained through “enrollment”
- Incremental adaptation
  - Models are updated as the system is used
  - Requires unsupervised adaptation
  - Requires objective comparison between adapted and baseline model
    - Likelihood gain

Goals for CALO Project

- CALO must learn and adapt to its users
  - Speaker adaptation is thus an essential part of the ASR component of CALO
  - Currently, we will be doing offline, unsupervised batch adaptation - to improve recognition for each individual speaker over the course of several multiparticipant meetings
  - In the future we will also do on-line, incremental adaptation
  - For the meeting domain, adaptation is important for improving overall recognition accuracy

Types of Adaptation

- Feature-based Adaptation a.k.a. Speaker Transformation a.k.a. VTLN
  - A transformation is applied in the front-end to the observation vectors
  - Acoustic warping of speaker towards the mean of the model
  - Can be done in spectral or cepstral domain
- Model-based Adaptation
  - The parameters of the acoustic model are modified based on the adaptation data
  - Can be done on-line or off-line

"Classical" Adaptation Methods

- There are two well-established methods for model-based speaker adaptation
- Each has given rise to a class of related techniques.
- It is possible to combine different techniques, with an additive effect on accuracy.

MAP (Bayesian Adaptation)

- Uses MAP estimation, based on Bayes' decision rule, to update the parameters of the model given the adaptation data
  - Maximizes the posterior probability given the model and the observation data.
  - Asymptotically equivalent to ML estimation
  - Given enough adaptation data, it will converge to a speaker-dependent model

MAP (Bayesian Adaptation)

- Good for large amounts of data, off-line adaptation
- Can only update parameters for HMM states seen in the adaptation data
  - Use smoothing to mitigate this problem
  - Or you can combine it with MLLR…
- Also unsuitable for unsupervised adaptation

MLLR (Transformation Adaptation)

- Calculates one or more linear transformations of the means of the Gaussians in an acoustic model
  - Find the matrix W which, when applied to the extended mean vector, maximizes the likelihood of the adaptation data
- Gaussians are tied into *regression classes*
  - Usually done at the GMM or phone level
  - If each GMM has its own class, MLLR is equivalent to a single iteration of Baum-Welch

MLLR (Transformation Adaptation)

- MLLR is robust for unsupervised adaptation
- MLLR is effective for very small amounts of data
  - Regression class tying allows adaptation of states not observed in the adaptation data
  - But… word error for a given number of classes levels off (and may increase slightly) as the amount of adaptation data increases
- Solution: Increase the number of regression classes
  - Or use MAP as well (if you can)

Determination of transformation classes

- Assumption:
  - Things which are close to each other in acoustic space will move similarly from one speaker to another
- Generate transformation classes using:
  - Linguistic criteria of similarity
  - Data-driven clustering
- Fixed regression classes
  - Suitable if the amount of adaptation data is known in advance
- Regression class tree
  - Generate classes of optimal size dynamically

Other methods

- ABC (Adaptation by Correlation)
- MAPLR
  - MAP estimation of the mean transformation
- EMAP
- Eigenspace methods
- MLLR variants
  - Matrix analysis to optimize transformation (PC-MLLR, WPC-MLLR)
  - Restricted form of transformation matrix (BD-MLLR)
- PLSA adaptation (for SCHMM)
- Stochastic Transformation (MLST)

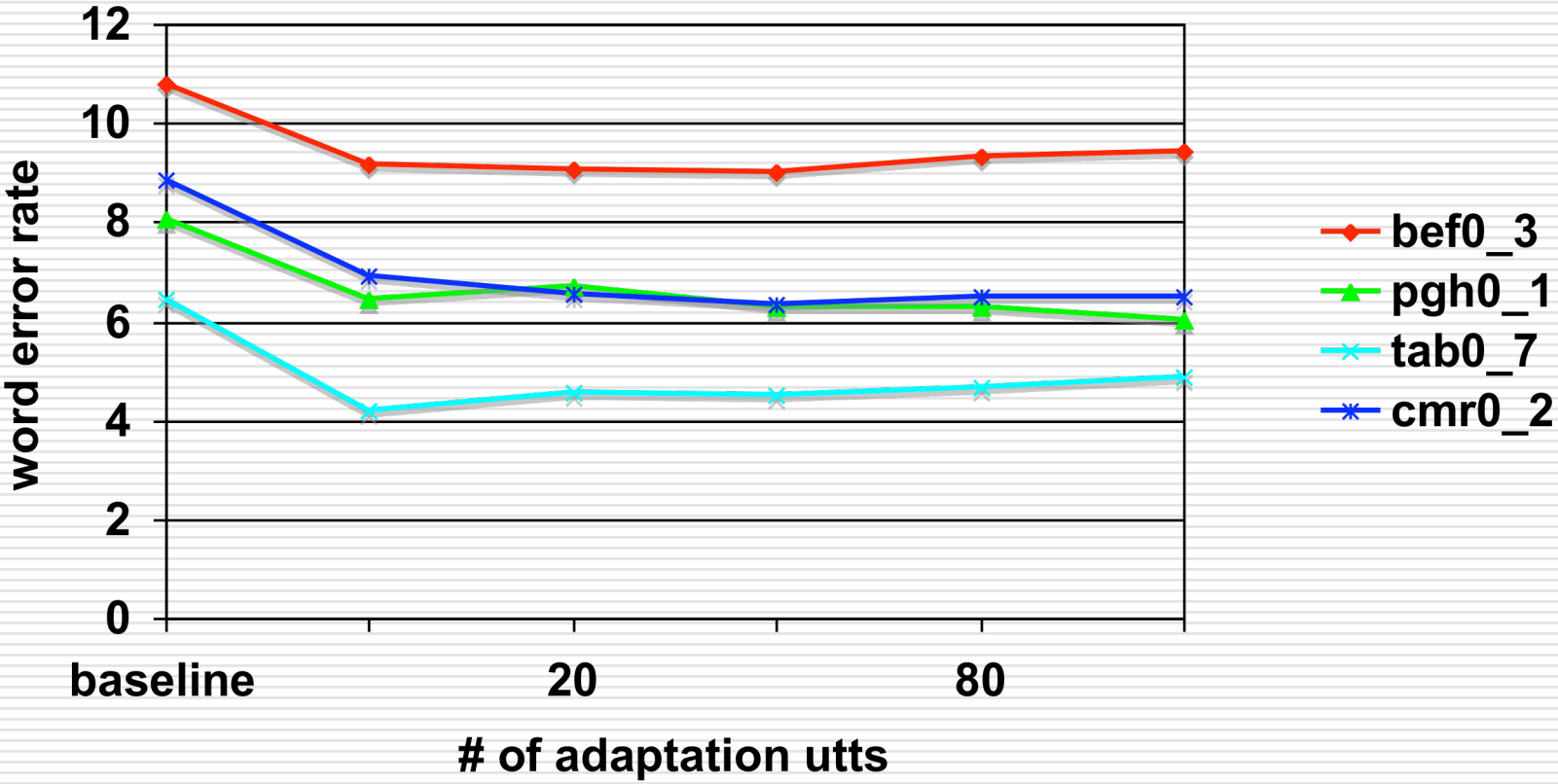
Adaptation with SphinxTrain

- Code from Sam-Joo Doh's thesis work
  - Other contributors: Rita Singh, Richard Stern, Arthur Chan, Evandro Gouvêa
- Single iteration of Baum-Welch
  - bw [baseline model] [adaptation data]
- Create MLLR matrix file
  - mllr_solve [baseline means] [gauden_counts]
- Apply to mean vectors (on-line or off-line):
  - mllr_adapt [baseline means] [matrix]
  - decode -mllrctl [matrix control file]

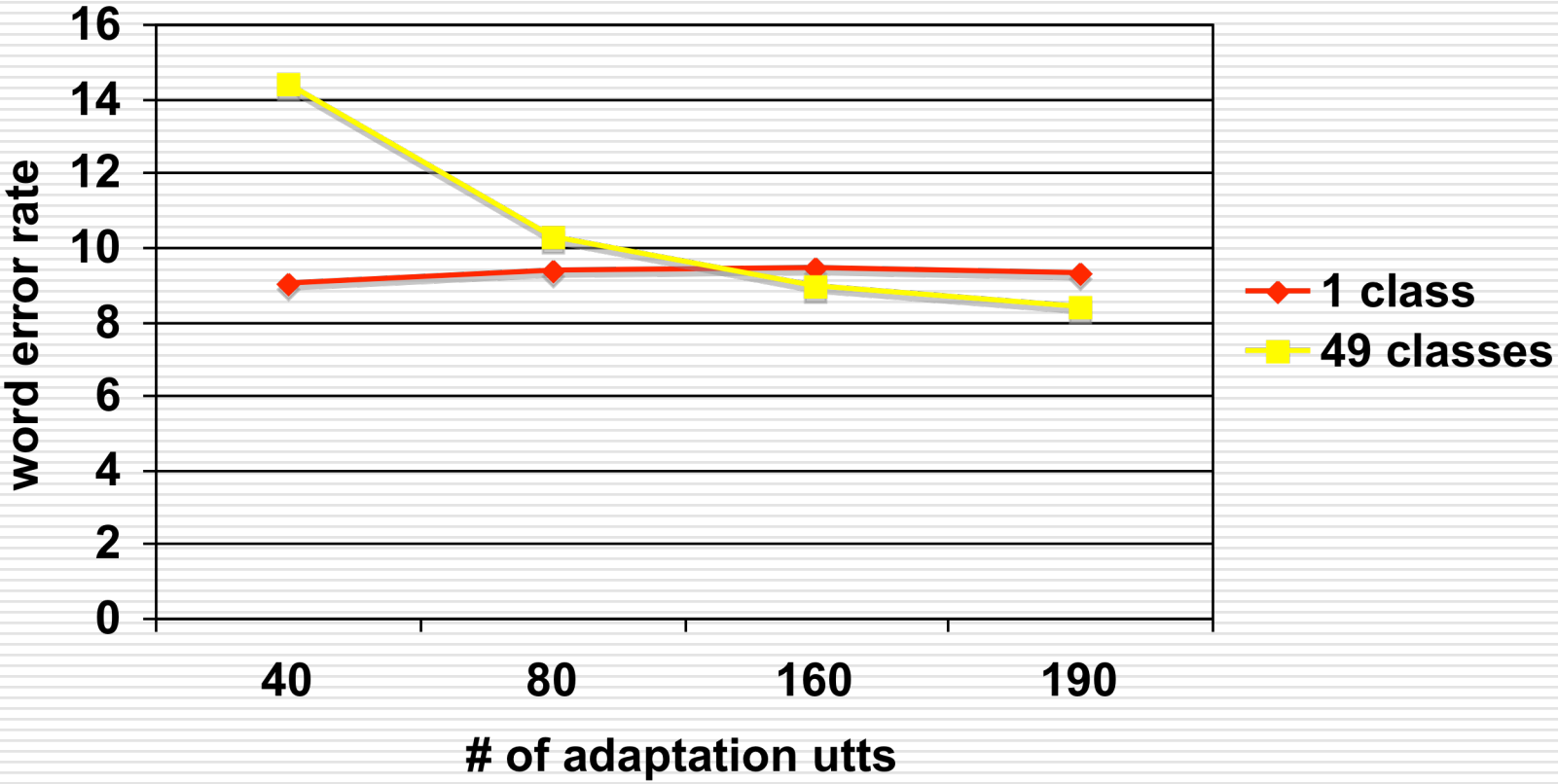
Multi-Class MLLR

- Do Baum-Welch as above
- Read model definition file, find transformation classes and output listing (one line per senone)
- Convert to binary class mapping file
  - mk_mllr_class < [listing file]
- Use in computing MLLR matrix file
  - mllr_solve -cb2mllrfn [class mapping file]

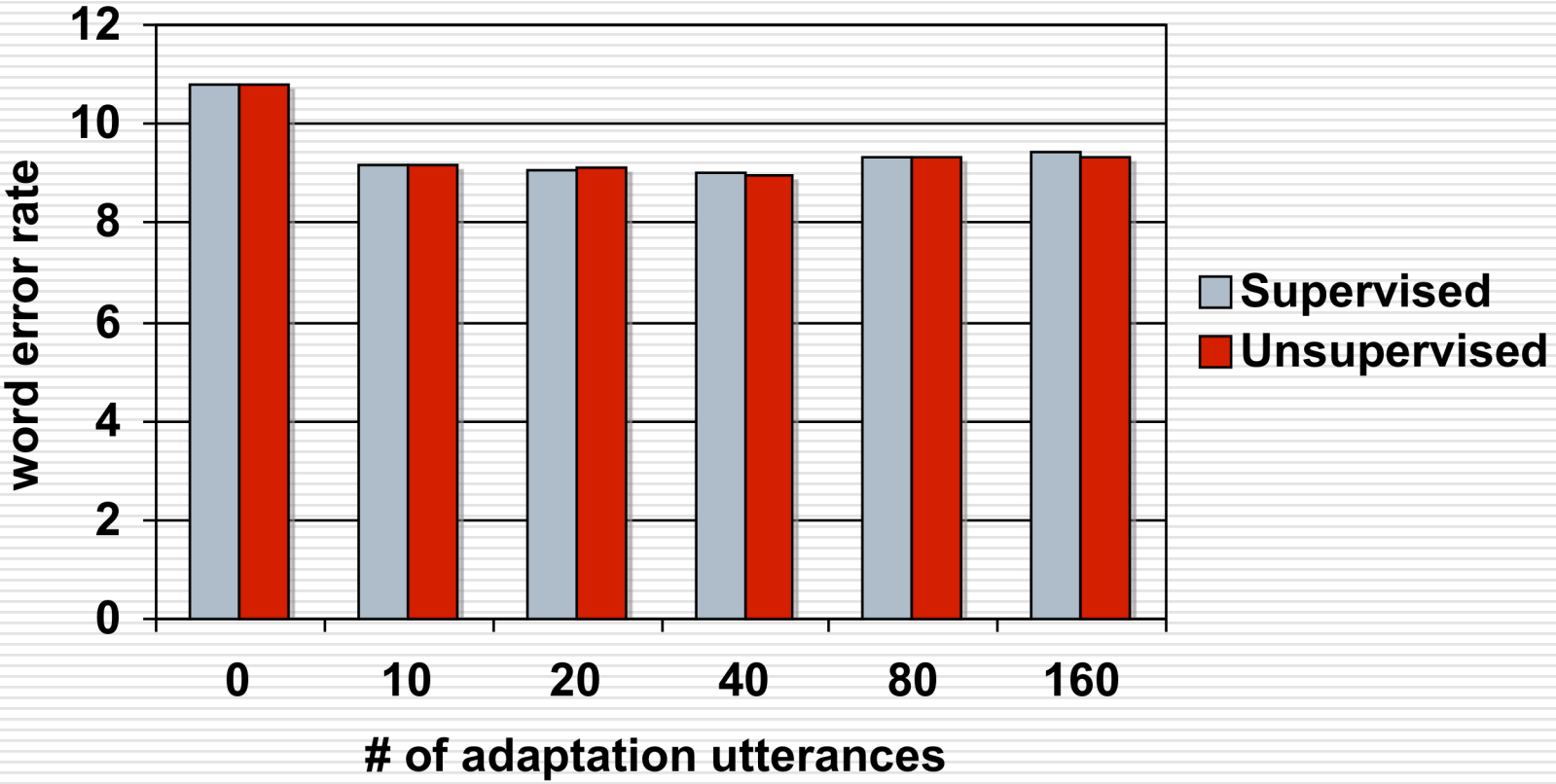
RM1, 1 regression class



RM1, 49 classes, 1 speaker



RM1, Supervised vs. Unsupervised



Current Development

- Clustering and regression class trees for multi-class MLLR (Q4 2004)
- Application to meeting domain (Q4 2004)
  - ICSI and CMU meeting data
- Unsupervised incremental adaptation
  - Confidence scoring, likelihood tracking
  - Integration of higher-level information for confidence estimation
- MAP

Thanks

- The usual suspects:
  - Alex Rudnicky
  - Arthur Chan
  - Evandro Gouvêa
  - Rita Singh
  - Richard Stern
- Any questions?