Phonetic Speaker Recognition

Winter School on Speech and Audio Processing

IIT Kanpur, January 2009

Andreas Stolcke

Speech Technology and Research Laboratory

SRI International, Menlo Park, Calif., U.S.A.

Joint work with:

A. Hatch (ICSI) , S. Kajarekar, L. Ferrer

Overview

"Higher-level features", Part 2

- Phonetic speaker recognition
- History
- Variants
  - Likelihood-ratio based
  - ASR-conditioned
  - SVM- based
  - Lattice-based
- Rank normalization
  - Word N-grams and SNERFs revisited

Motivation

- Most applied speaker recognition is based on **short-term cepstral** features
  - Cepstral features are primarily a function of speakers vocal tract shape
  - Cepstral features are affected by extraneous variables, like channel and acoustic environment
- Phone-based approaches
  - Also model acoustics
  - But at a different level of granularity
  - Capture pronunciation variation between speakers
  - Discretize the acoustic space (into phone categories)
  - Enable the modeling of longer-term patterns (phone N-grams)

History

- Phone N-gram language modeling (Andrews et al. ‘01)
- Open-loop phones conditioned on word recognition (Johns Hopkins SuperSID Workshop, Klusacek et al. ‘03)
- Phone sequence modeling with decision trees (Johns Hopkins SuperSID Workshop, Navrátil et al. ‘03)
  – Jiri’s lecture will explain this in the context of language ID
- SVM-based modeling (Campbell et al. ’04a)
  – Replaces likelihood ratios with SVM kernel function
- Lattice-based modeling (Hatch et al. ’05a)
  – Leverages multiple recognition hypotheses
- Rank normalization (Stolcke et al. ‘08)
  – Improved feature scaling for SVM modeling

Phonetic SR Compared to Other Approaches

Feature Type	Feature Description	Time Span	ASR to Find Unit	ASR to Condition
Cepstral	phone-conditioned text-conditioned GMMs phone HMMs whole word	▪ ▪ ▪ —	Ø Ø phone, word Ø	phone word, syll. phone N-gram
Cepstral-Derived	MLLR adapt. transforms	▪	word, unc. phone	phone
Acoustic Tokenization	phone N-gram freq. conditioned pron. model	— —	Unconstrained phone rec.	Ø phones
Prosodic	dynamics duration syllable-pros. sequences	— — —	Ø state, phone, syllable	Ø phone, word word
Lexical	word N-grams	—	word	Ø

Disclaimer on Results (again!)

- Many of the results presented are historical
- Results obtained on different training/test sets
- Baselines vary and get better the more recent the results
- Gains over baseline may also vary
  - The better the baseline, the less typically the gain

- **Your mileage may vary !**

Phonetic Modeling

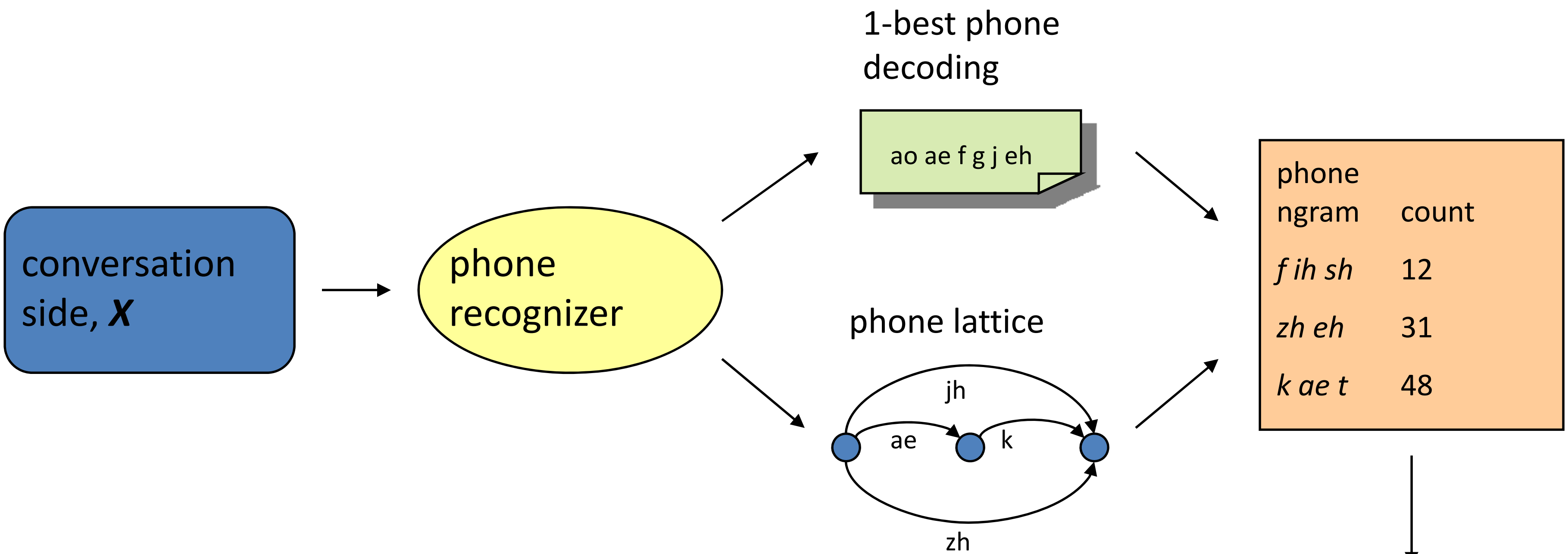
Phone N-gram Features

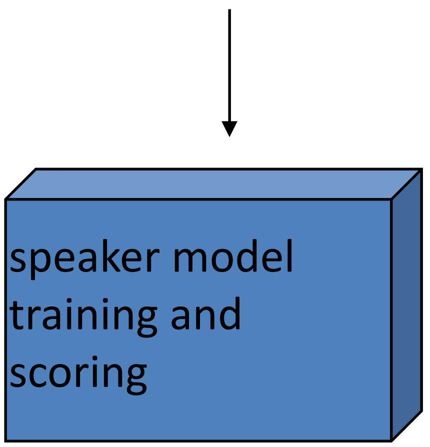
- Idea:
  - Map continuous speech signal into a string of phone labels: **tokenization**
  - Phone frequencies will reflect phonetic idiosyncrasies
  - We are not aiming to do accurate phone recognition …
  - Therefore: phone recognition best **without** phonotactic constraint (language model): **open-loop recognition**
  - Approach was first used for language ID (Zissman et al. '94)

- Implementation:
  - Get phone recognition output
  - Extract N-gram frequencies
  - Model likelihood ratio OR
  - Model frequency vectors by SVM
  - Note: this is just like for word N-grams!

phone ngram	count
f ih sh	12
zh eh	31
k ae t	48

Phonetic Processing



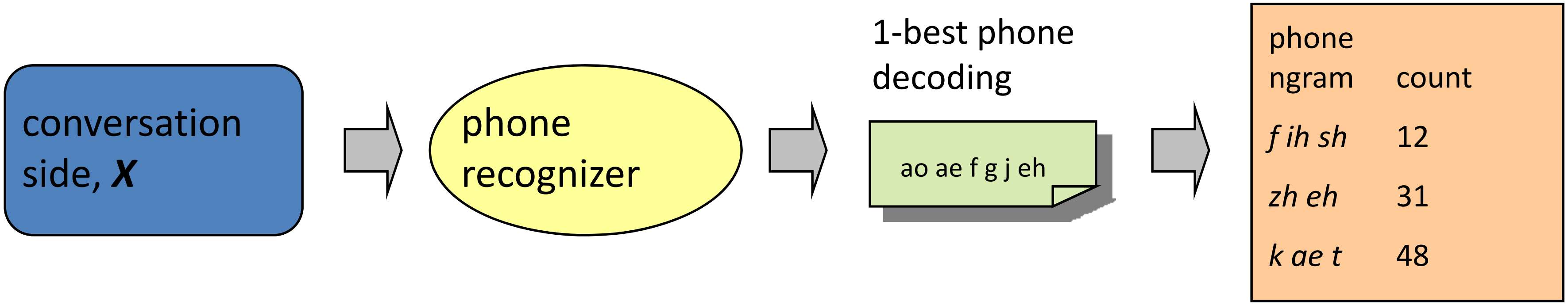


- Why phone lattices?
  – More robust counts
  – Finer granularity in features

1-Best Decoding vs. Lattice Decoding

- **1-best phone decoding**

  counts of phone ngrams are obtained directly from the output phone stream:



- **Lattice phone decoding**
  - same as above except we use a lattice to compute *expected counts.*
  - the *expected count* of phone ngram d_i in conversation side X is computed over all phone sequences, Q, within X:

$$E[count(d_i \mid X)] = \sum_{Q} p(Q \mid X) \cdot count(d_i \mid Q)$$

phone ngram, $\boldsymbol{d_i}$

conv. side, $\boldsymbol{X}$

phone sequence, $\boldsymbol{Q}$

Computing Expected N-gram Counts

- Computed efficiently by dynamic programming over the lattice
  - Compute posterior probabilities for each node and transition, using forward-backward algorithm (based on recognizer scores)
  - Implicitly expand lattice to create unique N-gram histories at each node
  - Forward dynamic programming: sum expected counts occurring between initial node and each node in lattice
  - Totals at final node contain results
- Implemented in SRI LM toolkit
  - Open source, free for non-commercial use
  - Accepts input lattices in HTK standard lattice format
  - http://www.speech.sri.com/projects/srilm/

Phone N-gram Modeling: Log-Likelihood Ratios

- Speaker model training: use relative frequencies of phone ngrams within speaker's training data, e.g.

  Spkr *A* model = $\{ p_s(d_1 \mid spk_A), p_s(d_2 \mid spk_A), \ldots, p_s(d_M \mid spk_A) \}$

- Scoring: LLR for conv. side *A* given speaker model *B* is

$$LLR(A,B) = \sum_{d_i} p(d_i \mid convSide_A) \log \frac{p_s(d_i \mid spk_B)}{p(d_i \mid bkg)}$$

- Here, $p(d_i \mid convSide_A)$, $p(d_i \mid spk_B)$, and $p(d_i \mid bkg)$ represent the relative frequencies of phone ngram d_i within conv. side *A*, speaker model *B*, and the background model, resp.

- **MAP smoothing** was applied to the relative frequencies of the speaker models:

$$p_s(d_i \mid spk_A) = (1-\alpha) \cdot p(d_i \mid spk_A) + \alpha \cdot p(d_i \mid bkg)$$

Phone N-gram Modeling with SVM

- **Speaker model training:** relative frequencies of phone ngrams within conv. sides are used to train target speaker SVM
- **Kernel selection:** Choose the **TFLLR** kernel function that approximates log likelihood ratio, following Campbell et al. (2004a):

$$k(A,B)=\sum_{i=1}^{M}\frac{p(d_i\,|\,convSide_A)}{\sqrt{p(d_i\,|\,bkg)}}\frac{p(d_i\,|\,convSide_B)}{\sqrt{p(d_i\,|\,bkg)}}$$

- **LLR kernel** reduces to a standard **linear kernel** if Input feature vectors consist of **scaled** versions of relative frequencies. Feature vector for speaker *A*:

$$x_A=\left\{\frac{p(d_1\,|\,convSide_A)}{\sqrt{p(d_1\,|\,bkg)}},\frac{p(d_2\,|\,convSide_A)}{\sqrt{p(d_2\,|\,bkg)}},\ldots,\frac{p(d_M\,|\,convSide_A)}{\sqrt{p(d_M\,|\,bkg)}}\right\}$$

Conditional Phone Modeling (Klusacek et al. ’03)

- **Aim:** Model speaker-dependent pronunciations by aligning word-constrained ASR phones with open-loop phones
- **Approach:** Align ASR phones with open loop phones at frame level and compute conditional probabilities

Pr(OL_phone | ASR_phone, speaker) =
#(OL_phone, ASR_phone) /
#(ASR_phone)

WORD	TIME	ASR	EG	GE	SP	JA	MA
TO	24964	t	n	n	n	sh	N
	24965	t	s	h	s	sh	N
	24966	t	s	h	s	sh	N
	24967	t	s	h	s	sh	S
	24968	t	s	h	s	sh	S
	24969	t	s	h	s	sh	S
	24970	t	s	h	s	rx	S
	24971	ax	l	h	s	rx	i:
	24972	ax	l	h	iy	rx	i:
	24973	ax	l	h	iy	y	i:

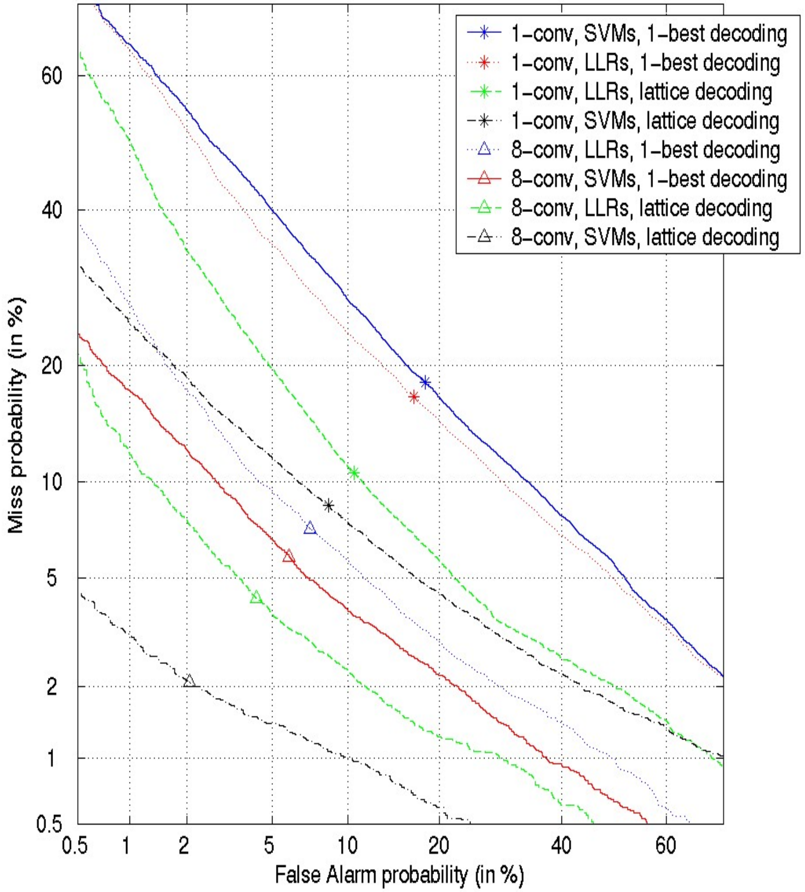
- During scoring compute likelihood of observed (OL_phone,ASR_phone) sequence against speaker and background models
- Scores from five language-specific open-loop phone streams are combined linearly

Phone N-gram Experiments

- **Data:** NIST SRE-03
  - Uses phases II and III of the Switchboard-2 corpus
  - Approx. 14000 conversation sides, each containing about 2.5 minutes of speech
- **Phone recognizer**
  - SRI Decipher™ system
  - Trained on Switchboard-1 and other conversational telephone data
  - 47 phones (including laughter, nonspeech)
  - No phonotactic language model (open-loop decoding)
- **Experiments:**
  - Training on 1-conv and 8-conv sides
  - Compare LLR vs. SVM modeling, and 1-best vs. lattice decoding
  - All experiments used phone **bigrams** features only
  - Half the data was used for background training, remainder for target training + test; then both data sets were swapped and results aggregated (jackknifing)
  - MAP smoothing parameters for LLR scoring were tuned on Switchboard-1 data

Phone N-gram Modeling: Results

Modeling	Training data	
	1 side	8 sides
LLR, 1-best	16.4	6.1
LLR, lattice	10.5	4.2
Improvement	*36%*	*31%*
SVM, 1-best	18.2	5.9
SVM, lattice	8.5	2.0
Improvement	*53%*	*66%*
Improvement over LLR	*19%*	*52%*



LLR MAP Smoothing Parameters

- Recall that MAP smoothing was used in for LLR scoring:

$$p_s(d_i \mid spk_A) = (1-\alpha) \cdot p(d_i \mid spk_A) + \alpha \cdot p(d_i \mid bkg)$$

- α was estimated on Switchboard-1 (disjoint from test data)
- We can compare α values for different systems:

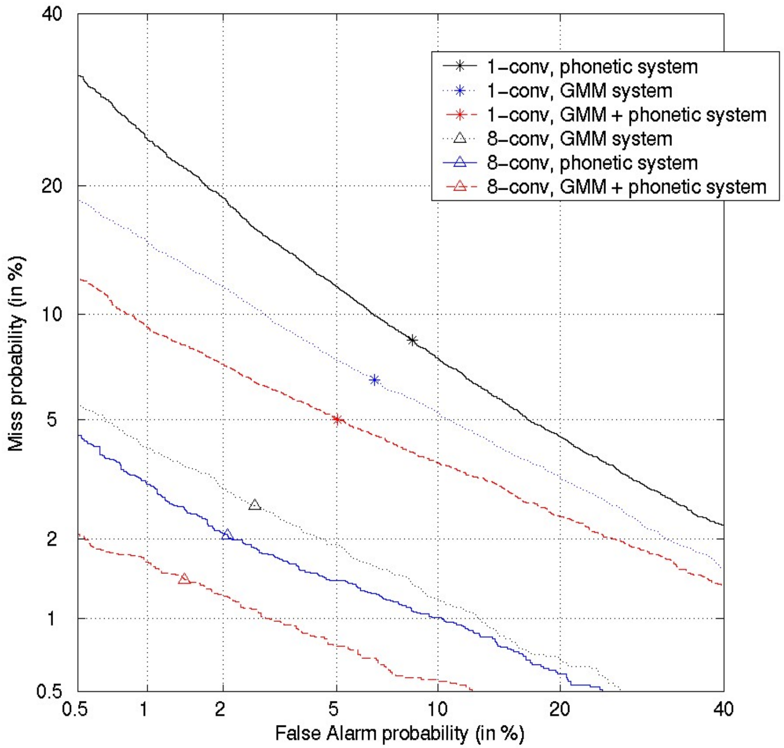
	Training data	
	1 side	8 sides
1-best decoding	0.955	0.670
lattice- decoding	0.920	0.040

- We see that lattice decoding decreases the need for smoothing or counts, since lattice counts are less noisy than 1-best

Phone N-grams Combined with Baseline

- Baseline: cepstral GMM
- Linear score combination

Sstem	Training data	
	1 side	8 sid
Phone lattice SVM	8.5	2.0
Cepstral GMM	6.6	2.6
Phonetic + Cepstral	5.0	1.4
Improvement	24%	46%



Rank Normalization

SVM Modeling Revisited

1. **Raw feature extraction**: compute cepstral features, prosodic features, phone or word n-grams, etc.
2. **Feature reduction transform:** condense all observations for a speech sample into a single feature vector of fixed length, e.g.,

   Cepstral features $\Rightarrow$ Gaussian or MLLR supervector

   Phone/word N-grams $\Rightarrow$ relative N-gram frequencies
3. **Feature normalization:** scale or warp features to improve modeling
4. **Kernel computation:** apply a standard SVM kernel function, such as linear (inner product), quadratic, exponential.

Note: Boundaries between these steps are arbitrary, but useful because a range of common choices at each step are combined in practice.

SVM Feature Normalization

SVM kernel functions are sensitive to the **dynamic range** of features dimensions

- Multiplying a feature by a constant factor increases feature's relative contribution to kernel function
- Therefore, absent prior knowledge, we should *equate* dynamic ranges of feature dimensions
- Alternatively, one can optimize scaling factors according to SVM loss function (Hatch et al. '05b)

Let's look at various choices for feature normalization

- as applied to **a variety of raw features**
- always using a **linear kernel** function

Method 1: Mean and Variance Normalization

- Subtract feature component means, divide by standard deviations
- Commonly used in many machine learning scenarios
- Equates feature ranges only if distributions have similar shapes
- We only need variance scaling – don't subtract the means
  - SVMs with linear kernel are invariant to constant offsets in feature space
  - Preserved sparseness of features vectors
  - Makes SVM processing more efficient with suitable implementation
- Scaling function:

$x_i' = d_i x_i$ scaled feature value

$d_i = 1/\sigma_i$ scaling factor

σ_i = standard deviation of feature x_i

Method 2: TFLLR Scaling

- Designed for N-gram frequency features
  - E.g., phones and words
- Proposed by Campbell et al. (2004a) to approximate LLR scoring of phone N-gram frequencies
- Each feature dimension is scaled by inverse square root of the N-gram corpus frequency:

$x_i' = d_i x_i$	scaled feature value
$d_i = f_i^{-1/2}$	scaling factor

- Gives more importance to rare (hence more informative) N-grams

Method 3: TFLOG Scaling

- Proposed by Campbell et al. (2004b) for word N-gram features
- Inspired by TF-IDF weighting used in information retrieval (term frequency – inverse document frequency)
- Similar to TFLLR, but scaling factor is given by a log function, with a maximum value *C*:

$x_i' = d_i \, x_i$	scaled feature value
$d_i = \min \{ -\log f_i + 1, C \}$	scaling factor

Method 4: Rank Normalization

- Non-parametric distribution scaling/warping
- First, replace each feature value by its rank in the sorted background data
- Then, scale ranks to unit interval: [0 … 1], e.g.,

10th out of 100 $\Rightarrow$ 0.1

- Formally:

$$x_i' = \frac{\left|\{y_i \in B : y_i < x_i\}\right|}{|B|}$$

where *B is the background data*

Rank Normalization (cont.)

- Intuitive interpretation:
  - Any distribution is warped to a uniform distribution, assuming background data is representative of test data
  - Distance between mapped data points is proportional to the percentage of the population that lies between them
  - High-density regions are expanded, low-density regions are compressed
- If non-negative, sparse feature vectors remains sparse

0th out of 100 $\Rightarrow$ 0

Features Used in Experiment

SNERF Prosodic feature sequences [recall 1st lecture]: Syllable-based pitch, energy, and duration features, as well as sequences of same for two and three syllables, mapped to **38,314 dense** feature dimensions via GMM weight transform

Phone N-grams: relative frequencies of the **8,483** most frequent phone bigrams and trigrams, obtained from phone lattices; **somewhat sparse**

Word N-grams [recall 1st lecture] relative frequencies of **126k** word unigrams, bigrams, and trigrams from 1-best ASR output; **very sparse** feature vectors

MLLR transform features [to be explained in 3rd lecture]: Coefficients of PLP-based speaker adaptation transforms from a speech recognizer, for 8 difference phone classes, yielding **24,960 dense** feature dimensions

Note: no other score or feature normalizations

Experiment Data

- Data from ‘05 and ’06 NIST SRE
- English telephone conversations
- About 2.5 minutes of speech per side
- Speaker models trained and tested on 1 conversation side
- Compare EERs

Feature Scaling: Results

Normalization Method	SRE'05	SRE'06
Phone N-grams		
None	14.64	12.30
Variance	12.62	10.84
TFLLR	12.66	10.73
Rank	**12.18**	**10.30**
Word N-grams		
None	24.76	22.98
Variance	32.04	31.07
TFLOG, $C = 10$	23.10	21.79
TFLOG, $C = \infty$	23.14	**21.63**
Rank	**22.49**	23.19

Feature Scaling: Results (cont.)

Normalization Method	SRE'05	SRE'06
Prosody SNERFs		
None	15.57	14.19
Variance	13.96	14.08
Rank	**13.88**	**13.65**
MLLR Transforms		
None	6.15	5.29
Variance	5.34	3.94
Rank	**5.22**	**3.61**

Feature Scaling: Conclusions

- Ranknorm is uniformly best or near-best for all feature types
- Variance normalization breaks down for very sparse features (word N-grams)
  - Variance estimates become too noisy
- TFLLR no better than variance (or rank) normalization for phone N-grams
- TFLOG works well for word N-grams, though we found that limit parameter *C* is not required
- Rank normalization gives largest relative gains for MLLR features
- Need to study possible interactions of component-level feature normalization with global transform methods, such as nuisance attribute projection (NAP)

Summary

- Phone N-grams can yield a powerful speaker model by themselves
- SVM modeling is better than likelihood ratios
- Lattice recognition greatly improves accuracy
- Choice of SVM kernels and/or different feature scaling is important
- Rank normalization is a nonparametric feature scaling method that seems to work well for a wide range of speaker features

Thank you – Questions?

References (1)

A. G. Adami, R. Mihaescu, D. A. Reynolds, and J. J. Godfrey (2003), Modeling Prosodic Dynamics for Speaker Recognition, *Proc. IEEE ICASSP*, vol. 4, pp. 788-791, Hong Kong.

W. D. Andrews, M. A. Kohler, and J. P. Campbell (2001), Phonetic Speaker Recognition, *Proc. Eurospeech, pp. 149–153, Aalborg.*

B. Baker, R. Vogt, and S. Sridharan (2005), Gaussian Mixture Modelling of Broad Phonetic and Syllabic Events for Text-Independent SpeakerVerification, *Proc. Eurospeech*, pp. 2429–2432, Lisbon.

K. Boakye and B. Peskin (2004), Text-Constrained Speaker Recognition on a Text-Independent Task, *Proc. Odyssey Speaker and Language Recognition Workshop*, pp. 129-134, Toledo, Spain.

T. Bocklet and E. Shriberg (2009), Speaker Recognition Using Syllable-Based Constraints for Cepstral Frame Selection, *Proc. IEEE ICASSP*, Taipei, to appear.

W. M. Campbell (2002), Generalized Linear Discriminant Sequence Kernels for Speaker Recognition, *Proc. IEEE ICASSP*, vol. 1, pp. 161-164, Orlando, FL.

W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek (2004a), Phonetic Speaker Recognition with Support Vector Machines, in *Advances in Neural Processing Systems 16*, pp. 1377-1384, MIT Press, Cambridge, MA.

W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek (2004b), High-level speaker verification with support vector machines, *Proc. IEEE ICASSP*, vol. 1, pp. 73-76, Montreal.

W. M. Campbell, D. E. Sturim, D. A. Reynolds (2006), Support vector machines using GMM supervectors for speaker verification, *IEEE Signal Proc. Letters* 13(5), 308-311.

N. Dehak, P. Dumouchel, and P. Kenny (2007), Modeling Prosodic Features With Joint Factor Analysis for Speaker Verification, *IEEE Trans. Audio Speech Lang. Proc.* 15(7), 2095-2103.

G. Doddington (2001), Speaker Recognition based on Idiolectal Differences between Speakers, *Proc. Eurospeech*, pp. 2521-2524, Aalborg.

References (2)

M. Ferras, C. C. Leung, C. Barras, and J.-L. Gauvain (2007), Constrained MLLR for Speaker Recognition, *Proc. IEEE ICASSP*, vol. 4, pp. 53-56, Honolulu.

L. Ferrer, E. Shriberg, S. Kajarekar, and K. Sonmez (2007), Parameterization of Prosodic Feature Distributions for SVM Modeling in Speaker Recognition, *Proc. IEEE ICASSP*, vol. 4, pp. 233-236, Honolulu, Hawaii.

L. Ferrer, K. Sonmez, and E. Shriberg (2008a), An Anticorrelation Kernel for Improved System Combination in Speaker Verification. *Proc. Odyssey Speaker and Language Recognition Workshop*, Stellenbosch, South Africa.

L. Ferrer, M. Graciarena, A. Zymnis, and E. Shriberg (2008b), System Combination Using Auxiliary Information for Speaker Verification, *Proc. IEEE ICASSP*, pp. 4853-4857, Las Vegas.

L. Ferrer (2008), Modeling Prior Belief for Speaker Verification SVM Systems, *Proc. Interspeech*, pp. 1385-1388, Brisbane, Australia.

V. R. R. Gadde (2000), Modeling word duration, Proc. ICSLP, pp. 601-604, Beijing.

A. O. Hatch, B. Peskin, and A. Stolcke (2005a), Improved Phonetic Speaker Recognition using Lattice Decoding, *Proc. IEEE ICASSP*, vol. 1, pp. 169-172, Philadelphia.

A. O. Hatch, A. Stolcke, and B. Peskin (2005b), Combining Feature Sets with Support Vector Machines: Application to Speaker Recognition. *Proc. IEEE Speech Recognition and Understanding Workshop*, pp. 75-79, San Juan, Puerto Rico.

L. Heck et al. (1998), SRI System Description, NIST SRE-98 evaluation.

S. Kajarekar, L. Ferrer, K. Sonmez, J. Zheng, E. Shriberg, and A. Stolcke (2004), Modeling NERFs for Speaker Recogniition, *Proc. Odyssey Speaker Recognition Workshop*, pp. 51-56, Toledo, Spain.

S. S. Kajarekar (2005), Four Weightings and a Fusion: A Cepstral-SVM System for Speaker Recognition. *Proc. IEEE Speech Recognition and Understanding Workshop*, pp. 17-22, San Juan, Puerto Rico.

Z. N. Karam and W. M. Campbell (2008), A Multi-class MLLR Kernel for SVM Speaker Recognition, *Proc. IEEE ICASSP* pp. 4117-4120, Las Vegas.

References (3)

P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel (2005), Factor Analysis Simplified, *Proc. IEEE ICASSP*, vol. 1, pp. 637-640, Philadelphia.

P. Kenny, G. Boulianne, P.Ouellet, and P. Dumouchel (2006), Improvements in Factor Analysis Based Speaker Verification, *Proc. IEEE ICASSP*, vol. 1, pp. 113-116, Toulouse.

D. Klusacek, J. Navrátil, D. A. Reynolds, and J. P. Campbell (2003), Conditional pronunciation modeling in speaker detection, *Proc. IEEE ICASSP*, vol. 4, pp. 804-807, Hong Kong.

J. Navrátil, Q. Jin, W. D. Andrews, and J. P. Campbell (2003), Phonetic Speaker Recognition Using Maximum-Likelihood Binary-Decision Tree Models, *Proc. IEEE ICASSP*, vol. 4, pp. 796-799, Hong Kong.

A. Park and T. J. Hazen (2002), ASR Dependent Techniques for Speaker Identification, *Proc. ICSLP*, pp. 1337–1340, Denver.

D. A. Reynolds, T. F. Quatieri, and R. B. Dunn (2000), Speaker Verification Using Adapted Gaussian Mixture Models, *Digital Signal Processing* 10, *181-202.*

D. Reynolds (2003), Channel Robust Speaker Verification via Feature Mapping, *Proc. IEEE ICASSP*, vol. 2, pp. 53-56, Hong Kong.

E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke (2005), Modeling prosodic feature sequences for speaker recognition, *Speech Communication* 46(3-4), 455-472.

E. E. Shriberg (2007), Higher Level Features in Speaker Recognition, in C. Müller (Ed.) *Speaker Classification I.* Volume 4343 of Lecture Notes in Computer Science / Artificial Intelligence. Springer: Heidelberg / Berlin / New York, pp. 241-259.

E. Shriberg and L. Ferrer (2007), A Text-Constrained Prosodic System for Speaker Verification, *Proc. Eurospeech*, pp. 1226-1229, Antwerp.

E. Shriberg, L. Ferrer, S. Kajarekar, N. Scheffer, A. Stolcke, and M. Akbacak (2008), Detecting Nonnative Speech Using Speaker Recognition Approaches. *Proc. Odyssey Speaker and Language Recognition Workshop*, Stellenbosch, South Africa.

References (4)

A. Solomonoff, C. Quillen, and I. Boardman (2004), Channel Compensation for SVM Speaker Recognition, *Proc. Odyssey Speaker and Language Recognition Workshop*, pp. 57-62, Toledo, Spain.

K. Sonmez, E. Shriberg, L. Heck, and M. Weintraub (1998), Modeling Dynamic Prosodic Variation for Speaker Verification, *Proc. ICSLP*, pp. 3189-3192, Sydney.

A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman (2005), MLLR Transforms as Features in Speaker Recognition, *Proc. Eurospeech*, pp. 2425-2428, Lisbon.

A. Stolcke, S. Kajarekar, L. Ferrer, and E. Shriberg (2007), Speaker Recognition with Session Variability Normalization Based on MLLR Adaptation Transforms, *IEEE Transactions on Audio, Speech, and Language Processing*, 15(7), 1987-1998.

A. Stolcke and S. Kajarekar (2008), Recognizing Arabic Speakers with English Phones. *Proc. Odyssey Speaker and Language Recognition Workshop*, Stellenbosch, South Africa.

A. Stolcke, S. Kajarekar, and L. Ferrer (2008), Nonparametric Feature Normalization for SVM-based Speaker Verification, *Proc. IEEE ICASSP*, pp. 1577-1580, Las Vegas.

D. E. Sturim, D. A. Reynolds, R. B. Dunn, and T. F. Quatieri (2002), Speaker Verification Using Text-Constrained Gaussian Mixture Models, *Proc. IEEE ICASSP*, vol. 1, pp. 677-680, Orlando.

G. Tur, E. Shriberg, A. Stolcke, and S. Kajarekar (2007), Duration and Pronunciation Conditioned Lexical Modeling for Speaker Recognition, *Proc. Eurospeech*, pp. 2049-2052, Antwerp.

R. Vogt, B. Baker, and S. Sridharan (2005), Modelling Session Variability in Text-independent Speaker Verification, *Proc. Eurospeech*, pp. 3117-3120, Lisbon.

M. A. Zissman and E. Singer (1994), Automatic language identification of telephone speech messages using phoneme recognition and N-gram modeling, *Proc. IEEE ICASSP*, vol. 1, pp. 305-308, Adelaide.