SUBSPACE GAUSSIAN MIXTURE MODELS FOR SPEECH RECOGNITION

Daniel Povey[1], Lukáš Burget[2],
Mohit Agarwal[3], Pinar Akyazi[4], Kai Feng[5], Arnab Ghoshal[6], Ondřej Glembek[2], Nagendra Kumar Goel[7],
Martin Karafiát[2], Ariya Rastrow[8], Richard C. Rose[9], Petr Schwarz[2], Samuel Thomas[8]

[1] Microsoft Research, Redmond, WA, dpovey@microsoft.com;
[2] Brno University of Technology, Czech Republic, burget@fit.vutbr.cz;
[3] IIIT Allahabad, India; [4] Boğaziçi University, Turkey; [5] HKUST, Hong Kong;
[6] Saarland University, Germany; [7] Go-Vivace Inc., Virginia, USA;
[8] Johns Hopkins University, MD; [9] McGill University, Canada

ABSTRACT

We describe an acoustic modeling approach in which all phonetic states share a common Gaussian Mixture Model structure, and the means and mixture weights vary in a subspace of the total parameter space. We call this a Subspace Gaussian Mixture Model (SGMM). Globally shared parameters define the subspace. This style of acoustic model allows for a much more compact representation and gives better results than a conventional modeling approach, particularly with smaller amounts of training data.

Index Terms— Speech Recognition, Hidden Markov Models, Gaussian Mixture Models

1. INTRODUCTION

This paper describes work done during the Johns Hopkins University 2009 summer workshop by the group titled "Low Development Cost, High Quality Speech Recognition for New Languages and Domains". For other work also done by the same team also see [1] which describes work on lexicon learning, [2] which describes the use of this approach in conjunction with out-of-language training data, and [3] which provides more details on issues of speaker adaptation in this framework. In [4] the technical details of the approach are presented more thoroughly than is possible here.

In the acoustic modeling approach we explore here, each speech state is a Gaussian Mixture Model (GMM) but the parameters of the GMM are not the parameters of our overall model. Instead, each state is associated with a vector-valued quantity of dimension similar to the feature dimension, and there is a globally shared mapping from this "state vector" to the means and weights of the state's GMM. This approach has some similarities to Eigenvoices [5] and Cluster Adaptive Training [6], except that we are using a subspace to model the variablility between phones rather than the secondary effect of speaker variation. There is also some relationship to the Joint Factor Analysis approach used in speaker identification [7]. Because we do not expect a single shared GMM to model all phones very well, we learn the mixture weights in the GMM which allows Gaussians in irrelevant locations in acoustic space to be turned off. We will show that learning the weights is an important feature of the model. The acoustic model we describe here seems to give substantially better results than a conventionally trained acoustic model.

Section 2 introduces the model; Section 3 discusses the model and the reasons why we have chosen this particular form; Section 4 discusses the framework of our experiments in terms of code and tools; Section 5 describes the software we used in testing it; Section 6 describes our experimental setup and training procedures, Section 7 gives experimental results, and Section 8 gives conclusions.

This work was conducted at the Johns Hopkins University Summer Workshop which was supported by National Science Foundation Grant Number IIS-0833652, with supplemental funding from Google Research, DARPA's GALE program and the Johns Hopkins University Human Language Technology Center of Excellence. BUT researchers were partially supported by Czech MPO project No. FR-TI1/034. Thanks to CLSP staff and faculty, to Tomas Kašpárek for system support, to Patrick Nguyen for introducing the participants, to Mark Gales for advice and HTK help, and to Jan Černocký for proofreading and useful comments.

2. SUBSPACE GMM ACOUSTIC MODEL

The most basic form of the model can be expressed in the following three equations:

$$p(\mathbf{x}|j) = \sum_{i=1}^{I} w_{ji} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{ji}, \boldsymbol{\Sigma}_i) \quad (1)$$

$$\boldsymbol{\mu}_{ji} = \mathbf{M}_i \mathbf{v}_j \quad (2)$$

$$w_{ji} = \frac{\exp \mathbf{w}_i^T \mathbf{v}_j}{\sum_{i'=1}^{I} \exp \mathbf{w}_{i'}^T \mathbf{v}_j}, \quad (3)$$

where $\mathbf{x} \in \Re^D$ is the feature, j is the speech state, $\mathbf{v}_j \in \Re^S$ is the "state vector" with $S \simeq D$ being the subspace dimension, and the model in each state is a simple GMM with I Gaussians, mixture weights w_{ji}, means $\boldsymbol{\mu}_{ji}$ and covariances $\boldsymbol{\Sigma}_i$ which are shared between states. The means and mixture weights are not parameters of the model. Instead they are derived from a state-specific vector $\mathbf{v}_j \in \Re^S$ with the "subspace dimension" S typically being around the same as the feature dimension D, via globally shared parameters $\mathbf{M}_i$ and $\mathbf{w}_i$. The reason why we describe it as a "subspace" model is that the state-specific parameters $\mathbf{v}_j$ determine the means $\boldsymbol{\mu}_{ji}$ and weights w_{ji} for all i, which is $I(D+1)$ parameters per state, but the dimension of S will typically be much less than $I(D+1)$ so the models span a subspace of the total parameter space.

For a typical setup the number of parameters in the vectors $\mathbf{v}_j$ would be very small relative to the globally shared parameters $\mathbf{w}_i$ and $\mathbf{M}_i$, so we introduce the notion of a "sub-state" where each state j has M_j sub-states each with its own mixture weight c_{jm} and

vector $\mathbf{v}_{jm}$ and the equations become:

$$p(\mathbf{x}|j) = \sum_{m=1}^{M_j} c_{jm} \sum_{i=1}^{I} w_{jmi} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{jmi}, \boldsymbol{\Sigma}_i) \quad (4)$$

$$\boldsymbol{\mu}_{jmi} = \mathbf{M}_i \mathbf{v}_{jm} \quad (5)$$

$$w_{jmi} = \frac{\exp \mathbf{w}_i^T \mathbf{v}_{jm}}{\sum_{i'=1}^{I} \exp \mathbf{w}_{i'}^T \mathbf{v}_{jm}}. \quad (6)$$

A further modification we make is to add an additional "speaker vector" $\mathbf{v}^{(s)} \in \Re^T$, which lives in a "speaker subspace" of dimension T (typically $T \simeq S \simeq D$). The speaker-adapted mean now becomes:

$$\boldsymbol{\mu}_{jmi}^{(s)} = \mathbf{M}_i \mathbf{v}_{jm} + \mathbf{N}_i \mathbf{v}^{(s)}, \quad (7)$$

where the $\mathbf{N}_i$ matrices define the "speaker subspace". The equation for the weights remains the same to avoid excessive per-speaker computation. The use of two symmetric terms in (7) is reminiscent of the Joint Factor Analysis approach in speaker identification [7].

2.1. Characteristics of the Subspace GMM model

Before going into detail on the training procedure of this model, we will summarize some of its main properties. Firstly, it is a Gaussian Mixture Model. This means that most standard techniques used in conventional modeling, such as VTLN, Constrained MLLR, normal feature extraction procedures etc., are applicable. Although the *expanded* GMM will typically be much larger than a normally configured GMM system, our model has fewer parameters than a normal system. By this we mean that a well-tuned SGMM system will typically have fewer parameters than a well-tuned GMM system, by typically a factor of two to four. With smaller amounts of training data, the parameter size of the SGMM system will actually be dominated by the *shared* parameters $\mathbf{M}_i$ and $\boldsymbol{\Sigma}_i$, which introduces the possibility of training the shared parameters on out-of-domain data and training the state-specific parameters on a smaller amount of in-domain data. We explore this in [2].

2.2. Training SGMM models

The training of this model is an Expectation-Maximization (E-M) procedure just like normal HMM training. In principle we need to alternate training different parameter types on different E-M iterations (e.g. $\mathbf{v}$ parameters then $\mathbf{M}$ parameters). We initialize the model by training a single GMM on all speech classes pooled together. We call this global model the "Universal Background Model" or UBM, and write its parameters as $\bar{\boldsymbol{\mu}}_i$, $\bar{\boldsymbol{\Sigma}}_i$ and $\bar{w}_i$. Although the UBM does not appear in Equations (4) to (6), it must be kept during later iterations of model training and during testing because it is used to prune the set of indexes i which we need to evaluate on each iteration. We initialize the parameters $\mathbf{M}_i$, $\mathbf{v}_{jm}$, $\boldsymbol{\Sigma}_i$ etc. in such a way that the means and variances in each state on the first iteration are the same as the UBM.

For the most part, training is fairly straightforward. The equations relating to the update of the parameters $\mathbf{M}$ and $\mathbf{v}$ are reminiscent of Speaker Adaptive Training (SAT) [8], in its original form as it relates to MLLR adaptation. The parameter updates for c_{jm} and $\boldsymbol{\Sigma}_i$ are very simple and analogous to normal GMM training. Updating the parameters $\mathbf{w}_i$ is slightly more difficult, as there is no natural E-M-like process to update it, but in [4] we describe a simple method that works well. Its derivation is based on a combination of Jensen-type inequalities, local second-order Taylor-series expansions, and a modification to the resulting quadratic auxiliary function which ensures stability while maintaining the same local gradient.

2.3. Decoding using SGMM models

In large vocabulary applications the decoding speed of the model is comparable to a normal HMM. This is so even though the expanded GMM is many times larger than a conventional system and uses full covariances. It is possible to evaluate likelihoods quickly because the extra structure of the model gives us opportunities for pre-computation and pruning that are not applicable in a conventional HMM-GMM system. As mentioned above, we use the UBM to prune the set of indexes i that we need to evaluate on each frame reducing it to a number (e.g. 10 or 20) that is comparable to the number of Gaussians in a state in a conventional system. We can structure the likelihood evaluation in such a way that evaluating each additional Gaussian is only $O(S)$; remember that $S \simeq D$. The memory requirements will usually be dominated by a single normalizing constant n_{jmi} that we compute for each Gaussian in the expanded GMMs; this contains data-independent terms in its contribution to the likelihood. The overall memory requirement is not much larger than a conventional model. Something we should note in connection with decoding is that the optimal language model weight with SGMM models is typically less than for conventional models, e.g. 10 rather than 14.

3. WHY THIS MODEL?

In this section we discuss why we have chosen the particular form of model of Equations (4) to (6). This is to address various questions and comments that we have encountered.

- **Is this model related to tied-mixture (semi-continuous) models?**
  It is different because the Gaussians within each state differ in mean as well as mixture weight. Also, the mixture weights are represented in a lower dimension rather than being parameters of the model.
- **Why introduce sub-states rather than simply increasing the subspace dimension?**
  Increasing the subspace dimension S would lead to an increase in the number of parameters in $\mathbf{M}_i$, which would lead to parameter estimation problems on modestly-sized systems. Also, we have never observed any benefit from increasing the subspace dimension beyond about 60 or so, whereas introducing sub-states consistently helps.
- **Is it necessary to model the mixture weights?**
  In speaker identification, the mixture weights are typically not modeled. However, in our experiments, when we turn off the estimation of $\mathbf{w}_i$ this model gives very bad results, as described in Section 7. So we believe the mixture weights are very important.
- **Why use the form $\frac{\exp \mathbf{w}_i^T \mathbf{v}_{jm}}{\sum_{i'=1}^{I} \exp \mathbf{w}_{i'}^T \mathbf{v}_{jm}}$ for the mixture weights?**
  This form makes the log mixture weights a linear function of the vector $\mathbf{v}_{jm}$ (the numerator: $\mathbf{w}_i^T \mathbf{v}_{jm}$), followed by a normalization to make them sum to one. It has the same form as multi-class logistic regression. It ensures that the weights are positive and sum to one.
- **Why use shared variances?**
  Making the variances a function of the vectors $\mathbf{v}_{jm}$ is very hard. The only way to make this fast in test-time is to make the precisions (inverse variances) linear functions of $\mathbf{v}_{jm}$, as

in SPAM [9]. This introduces difficulties ensuring positive definite variances[1]

- **Why use full covariances rather than diagonal?**
  Because of the structure of the model, using full covariances does not substantially change either the parameter count or the decoding speed. We will show in Section 7 that using full covariances helps substantially.

- **Is it possible to combine standard adaptation and discriminative training techniques with this model?**
  For the most part. Vocal Tract Length Normalization (VTLN) is trivial to combine with it. Constrained MLLR is also trivial, although because we use full covariances, if we want to estimate the transform from our model we need to compute it in a different way, which we describe in [3]. MLLR is very inefficient with this style of model because there is too much per-speaker precomputation to do, but it gives very little improvement on top of CMLLR in typical scenarios. We have previously successfully implemented discriminative training with this model (to be described in [11]).

4. CODE FRAMEWORK

The framework we developed to train and test SGMM models uses HTK [12] to do feature extraction and build the baseline models which are used to align the training data for the first few passes of training and initialize the UBM. After using HTK to build the initial models, we switch to our own `C++`-based framework for further training and decoding. Our tools for training and decoding have similar command-line options to the HTK tools `HERest` and `HVite`. We use the phonetic context tree of the HTK baseline models in our own system. Training and testing in our framework is based on Weighted Finite State Transducers (WFST) [13], for which we use the OpenFST tools and library [14]. We use WFSTs to obtain finite state acceptors at the HMM-state level for our training transcripts, and on the first few iterations of training, our tools evaluate the Viterbi path through this acceptor based on likelihoods we compute from the HTK models. Later iterations of training are based on a Viterbi alignment using our SGMM models' likelihoods. Decoding is done by reading in a Finite State Acceptor which contains the information compiled from the phonetic context tree, the lexicon, and the language model which we built using SRILM tools [15]. For WER results we report here, we used the NIST scoring tool `sclite`.

Our SGMM training and evaluation code makes heavy use of matrix operations, and for this we created a `C++` wrapper for standard matrix and vector libraries implemented in `C`. We used parts of `ATLAS`, `CLAPACK` and `TNT`. We intend to release the code under an open-source license; contact the authors for details.

[1] In fact, the first author has done experiments in which diagonal inverse variances were made a linear function of $\mathbf{v}_{jm}$ *with flooring*, e.g. $\frac{1}{\sigma^2}_{jmid} = \min(k_{id}, \mathbf{p}_{id} \cdot \mathbf{v}_{jm})$. This flooring sidesteps the issue of ensuring positivity, but it introduces difficulties for fast decoding. This is because to make the likelihood evaluation fast we need to remember which j, m, i, d were floored which is hard to do within acceptable memory limits. We combined this with a Semi-Tied Covariance transform [10] per i in place of the full covariances per i. No word error rate improvements were seen, although test-data likelihoods did improve.

5. DATABASES AND BASELINE SYSTEM

Here we report experiments on Callhome English. See [2] for further results on Spanish and German and on multi-lingual experiments. Callhome English is a part of the Callhome corpora [16] collected by LDC for languages including Spanish, Arabic, German, Mandarin and Japanese. The conversational nature of the speech database along with high out-of-vocabulary rates, use of foreign words and telephone channel distortions makes the task of speech recognition on this database challenging.

The English Callhome database consists of 120 spontaneous telephone conversations between native English speakers. Eighty conversations corresponding to about 15 hours of speech are used as training data. Two sets of 20 conversations, roughly containing 1.8 hours of speech each, form the test and development sets.

We use 39 dimensional PLP [17] features with energy and Δ and $\Delta\Delta$ and per-speaker mean and variance normalization to build a single pass HTK [12] based recognizer with 1920 tied states and 18 mixtures per state, tuned to optimize WER after adaptation. The same features and context tree were used for our system. We used a 62k word lexicon with an OOV rate of 0.4%, and a trigram language model with a perplexity of 95, built using the SRILM tools [15]. The language model is interpolated from individual models created from the English Callhome corpus, the Switchboard corpus [18], the Gigaword corpus and some web data. The web data is obtained by crawling the web for sentences containing high frequency bigrams and trigrams occurring in the training text of the Callhome corpus. The 90K PRONLEX dictionary with 47 phones is used as the pronunciation dictionary for the system.

6. TRAINING PROCEDURE

The training procedure for our SGMM models is as follows. We initialize the UBM by clustering the diagonal Gaussians in the HTK-derived HMM set to $I = 400$ Gaussians. We then train the UBM for eight iterations of full-covariance E-M on the full training set without class labels. We initialize the SGMM model from the UBM as described in [4], with the subspace dimension S the same as the feature dimension $D + 1$ (i.e. $S = 39 + 1 = 40$) and the matrices $\mathbf{M}_i$ initialized such that the last 39 dimensions of $\mathbf{v}_{jm}$ are interpreted as global offsets on the GMM's means. The initial SGMM model's mean and variance parameters are the same as the UBM in each state of the HMM. We train in epochs of 8 iterations. At the beginning of every epoch starting from the third epoch, we split substates up to some target value, perturbing the split vectors slightly as described in [4]; sub-states are allocated proportional to some small power (0.2) of the state count. On each iteration but the very first, we train all parameter types except $\mathbf{M}_i$, which are trained every other iteration. On the very first iteration, we only update $\mathbf{v}_{jm}$. Within each update phase, we update $\mathbf{w}_i$ for three iterations.

7. RESULTS

In Table 1, we show unadapted English results, with various modifications to show the relative importance of different features of the model. The SGMM system has a subspace dimension of 40. The results in Table 1 were obtained with a bigram language model; we used this for speed of turnaround and to keep memory requirements low in decoding; in the text we give selected trigram results.

The baseline WER is 54.7%; we tuned the size of the baseline system for best WER. The best SGMM result is 49.3%, for a 5.4% absolute (9.9% relative) WER reduction. Comparing the first and

GMM:	54.7 (1800 states, 16 Gauss/state)						
	#Substates						
	1800	2700	4k	6k	9k	12k	16k
SGMM:	51.6	50.9	50.6	50.1	49.9	**49.3**	49.4
Diag-var:	55.7	55.3	54.5	53.9	54.1	**53.7**	53.8
Fix-$\mathbf{\Sigma}_i$:	53.5	52.7	52.4	52.3	51.8	51.8	**51.5**
Fix-$\mathbf{w}_i$:	61.5	60.5	59.3	58.2	57.6	56.7	**56.2**
Fix-$\mathbf{M}_i$:	61.0	60.2	58.9	57.2	55.8	54.2	**53.1**

Table 1. Baseline and SGMM WERs: Callhome English, unadapted

GMM:	53.6						
+SAT:	49.3						
	#Substates						
	1800	2700	4k	6k	9k	12k	16k
SGMM:	50.1	49.8	49.2	48.9	48.5	48.0	47.9
+SAT:	50.0	49.6	49.1	48.5	48.2	47.8	47.7
+spk-vecs:		48.6	47.9	47.5	47.2	47.2	46.7
-SAT:		48.6	47.9	47.4	47.0	47.0	**46.7**

Table 2. Baseline and SGMM WERs with CMLLR adaptation

last columns of the "SGMM" row, we see that adding sub-states reduces the WER from 51.6% to 49.3%, or 4.5% relative. Forcing the model's covariances to be diagonal (the next row) gives a 7.9% relative degradation (the overall number of system parameters is only slightly reduced by this change). The last three rows show what happens if during training we do not update the parameters $\mathbf{\Sigma}_i$, $\mathbf{w}_i$ and $\mathbf{M}_i$ respectively. For the variances, this means they are set to the UBM's variance $\bar{\mathbf{\Sigma}}_i$; the un-trained $\mathbf{w}_i$ are left as zero vectors; and the un-trained $\mathbf{M}_i$ are all the same so the quantities $\mathbf{v}_{jm}$ are interpreted as offsets on the model means. The parameter type that makes the most difference is $\mathbf{w}_i$ which is interesting because it has the smallest parameter count, and this shows the importance of the weights in this model. The parameter that makes the least difference is $\mathbf{\Sigma}_i$, which makes sense because the UBM's initial variances $\bar{\mathbf{\Sigma}}_i$ are a reasonable setting. We repeated the baseline GMM experiment and the best SGMM experiment (originally 49.3%) with a trigram language model. These numbers reduced WERs to 52.5% and 47.4%. The relative WER reduction is about the same as before, at 9.7%.

Table 2 shows results with various forms of speaker adaptation, again with a bigram language model. The baseline used Constrained MLLR adaptation with and without Speaker Adaptive Training (SAT). We show results with and without SAT and with and without the speaker vectors of Equation 7. Silence was excluded from the estimation of all speaker adaptation parameters in SGMM experiments (except for the SGMM+SAT experiment, where silence was used in training time). Results are shown with a bigram language model; the trigram results are: GMM 49.7%, +SAT 46.0%, SGMM 45.9%, +SAT 45.5%, SGMM+spk-vecs+SAT 44.7%, SGMM+spk-vecs 44.5%. Comparing the best baseline and SGMM results (46.0% vs 44.5%) we have a 3.3% relative improvement.

8. CONCLUSIONS

We have described a new type of statistical model, the Subspace Gaussian Mixture Model (SGMM), and demonstrated that it can give substantially better results than a conventionally structured model, particularly without adaptation. We have shown the importance of various features of the model, such as modeling the weights; using full-covariance Gaussians; and using sub-states. In companion papers, we will describe the methods we used to optimize the Constrained MLLR transforms with this model [3] and show how we are able to leverage out-of-domain data to further improve error rates [2].

9. REFERENCES

[1] N. Goel et al., "Approaches to automatic lexicon learning with limited training examples," 2010, Submitted to: ICASSP.

[2] Lukas Burget, Petr Schwartz, et al., "Multilingual Acoustic Modeling for Speech Recognition based on Subspace Gaussian Mixture Models," 2010, Submitted to: ICASSP.

[3] Arnab Ghoshal, D. Povey, et al., "A Novel Estimation of Feature-space MLLR for Full Covariance Models," 2010, Submitted to: ICASSP.

[4] D. Povey, "A Tutorial Introduction to Subspace Gaussian Mixture Models for Speech Recognition," Tech. Rep. MSR-TR-2009-111, Microsoft Research, 2009.

[5] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid Speaker Adaptation in Eigenvoice Space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, Nov. 2000.

[6] M. J. F. Gales, "Multiple-cluster adaptive training schemes," in *ICASSP*, 2001.

[7] P. Kenny, P. Ouellet, N. Dehak, and V. Gupta, "A study of Interspeaker Variability in Speaker Verification," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 980–987, 2008.

[8] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A Compact Model for Speaker-Adaptive Training," in *ICSLP*, 1996.

[9] S. Axelrod, V. Goel, R. A. Gopinath, P. A. Olsen, and K. Visweswariah, "Subspace constrained Gaussian mixture models for speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 6, pp. 1144–1160, 2005.

[10] M.J.F Gales, "Maximum Likelihood Linear Transformations for HMM-based Speech Recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1997.

[11] D. Povey, S. M. Chu, and J. Pelecanos, "Approaches to Speech Recognition based on Speaker Recognition Techniques," Book chapter in forthcoming book on GALE project, 2009.

[12] S. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book (for version 3.4)*, Cambridge University Engineering Department, 2009.

[13] Mehryar Mohri, Fernando Pereira, and Michael Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech and Language*, vol. 20, no. 1, pp. 69–88, 2002.

[14] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: a general and efficient weighted finite-state transducer library," in *CIAA*, 2007.

[15] A. Stolcke, "SRILM - An Extensible Language Modeling Toolkit," in *ICSLP*, 2002.

[16] A. Canavan, D. Graff, and G. Zipperlen, *CALLHOME American English Speech*, Linguistic Data Consortium, 1997.

[17] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87, pp. 1738–1752, 1990.

[18] J. J Godfrey et al., "Switchboard: Telephone speech corpus for research and development," in *ICASSP*, 1992.