ISCA Archive
http://www.isca-speech.org/archive

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

A STUDY OF COMPUTATION SPEED-UPS OF THE GMM-UBM SPEAKER RECOGNITION SYSTEM *

Jack McLaughlin, Douglas A. Reynolds and Terry Gleason

Information Systems Technology Group, Lincoln Laboratory, MIT,
Lexington, MA 02420-9185
jackm | dar | tpg @sst.ll.mit.edu

ABSTRACT

The Gaussian Mixture Model Universal Background Model (GMM-UBM) speaker recognition system has demonstrated very high performance in several NIST evaluations. Such evaluations, however, are concerned only with classification accuracy. In many applications, system effectiveness must be evaluated in light of both accuracy and execution speed. We present here a number of techniques for decreasing computation. Using data from the Switchboard telephone speech corpus, we show that significant speed-ups can be obtained while sacrificing surprisingly little accuracy. We expect that these techniques, involving lowering model order as well as processing fewer speech frames, will apply equally well to other recognition systems.

1. INTRODUCTION

The Gaussian Mixture Model Universal Background Model (GMM-UBM) speaker recognition system [1] has demonstrated very high performance in several NIST evaluations. Total system effectiveness, however, is a function of both accuracy and computation. While the GMM-UBM system requires relatively modest computation requirements, there are many straightforward techniques which can be applied to further decrease the computation while maintaining the desired high accuracy. In this paper we present results of applying several algorithmic speed-ups to the GMM-UBM system showing the tradeoff between accuracy and computation. Since the computation factors for the GMM-UBM system are the same as for other GMM based recognizers, it is expected that results found in this study will generally apply to other systems.

Two factors dominate the computations in the GMM-UBM system: the number of Gaussians in the UBM and the number of feature vectors to be scored in a test utterance. Thus, our search for computation improvements focused on decreasing the acoustic resolution of the UBM (number of mixtures) and the temporal resolution of the input speech (number of vectors). Decreasing the acoustic resolution of the UBM can be done simply by training smaller UBMs. Decreasing the temporal resolution of the feature vectors is effectively done using various forms of decimation. In this paper we examined three types of decimation: fixed-rate decimation (score 1 out of every N vectors), variable-rate decimation (score only vectors which pass a difference threshold test), and adaptive-rate decimation (adapt decimation factor to produce a fixed number of vectors per test utterance). *

The remainder of the paper is organized as follows. In section 2 we provide a brief description of the GMM-UBM system used for the experiments. Section 3 describes model order reduction and the three methods of reducing the number of feature vectors to score. The effects on accuracy of each of these techniques using the 1998 NIST summer development corpus are shown in section 4. Finally, section 5 presents some conclusions and discussion.

2. SPEAKER RECOGNITION SYSTEM

The baseline system in this paper is the GMM-UBM speaker recognition system operating on mel-cepstral based feature vectors [1]. The feature vectors used in this system are 38 dimensional vectors consisting of appended 19 dimensional cepstra and 19 dimensional delta cepstra derived from bandlimited (300-3300 Hz) mel-filterbank spectra. Feature vectors are computed using a 20 ms window of speech and are produced every 10 ms (100 vectors/sec). An energy based silence detector is used to discard low energy portions of the signal (approximately 20-25% for telephone speech) and cepstral mean subtraction is performed for channel compensation.

*This work was sponsored by the Department of Defense under Air Force Contract F19628-95-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Air Force.

* The authors kindly acknowledge input from Robert Allison on decimation experiments.

The system is basically a likelihood ratio detector consisting of a speaker-independent universal background model (UBM) and a claimant model derived from the UBM via Bayesian adaptation. In this paper the UBM is trained using approximately two hours of speech evenly divided between male and female speakers. Claimant models are trained using approximately two minutes of speech evenly extracted from two separate conversations from the same telephone number. Our baseline system uses a 2048 mixture UBM. The claimant model will be of the same order as the UBM since it is derived from the UBM.

Because the claimant model is derived from the UBM, however, a fast scoring technique can be used during recognition similar to short-list scoring in speech recognition systems. For each input feature vector, all the UBM mixtures are scored to determine the top 5 highest scoring mixtures. Since UBM and claimant mixtures share a correspondence, the claimant model likelihood is computed using only the 5 claimant mixtures corresponding to the top 5 from the UBM. The final utterance score is then the difference between the claimant and UBM log-likelihood values (the log-likelihood ratio). Performance on a set of files is computed using a speaker-independent threshold and reported in full via Detection Error Tradeoff (DET) curves or simply as the equal-error rate (EER).

3. TECHNIQUES FOR REDUCING COMPUTATION

3.1 Reduction in Number of Mixtures

The easiest computational factor to decrease in the GMM-UBM system is the acoustic resolution (number of mixtures) of the UBM. While this reduction also reduces computation during training, our focus is on the computational savings during recognition. Even with using the fast-scoring technique described above, the size of the UBM dominates the computations since all mixtures must be evaluated to determine the top 5 scoring components. Other potential speed-ups in scoring the UBM mixtures, such as fast search techniques to determine the top-5 mixtures, selectively merging Gaussians to provide for more region-specific resolution control or hierarchical indexing into a high-resolution UBM using a lower-resolution map, were not pursued in this work.

3.2 Reduction in Number of Frames

For decreasing the temporal resolution of the input speech (number of vectors) we examined decimation techniques. While vector reduction can decrease computation during training, we applied decimation only during recognition. We did not want to convolve any effects the decimation may have had on the model parameter estimation with the recognition computational reductions we were examining in this paper. In addition, we restricted application of decimation to the feature vector stream after completion of all front-end processing (feature extraction, speech detection and channel compensation). We also excluded decimation by using longer analysis windows. These were done again to avoid any unwanted interactions between the front-end processing steps and decimation. This, of course, means there is no computational reduction in the front-end processing steps; but these are relatively fast operations compared to the GMM-UBM scoring.

The three types of decimation examined were: fixed-rate decimation, variable-rate decimation, and adaptive-rate decimation In fixed-rate decimation, one out of every N vectors is scored giving a fixed reduction of 1/N for each input utterance. This results in a predictable number of feature vectors from each utterance given the original utterance duration.

Variable frame rate (VFR) decimation attempts to select the subset of feature vectors to process by using some quantitative measure of how similar one feature vector is compared to surrounding vectors. In a sequence of vectors deemed to be similar, only the first of these is scored and that score is replicated for the remaining vectors under the assumption they would have produced similar scores. We use a weighted Euclidean distance as a similarity measure.

After scoring the first feature vector of an utterance, we compute the distance between that first vector and succeeding vectors until a distance exceeds some preset threshold. This succeeding vector is then scored and becomes the reference for future distance computations. By increasing the threshold, we decrease the number of feature vectors actually scored against the models. Although the threshold is fixed, the effective vector rate varies depending upon the data.

The nature of the VFR processing is that it tends to select transition regions in the speech signal, which raises the issue of whether the models should be retrained using VFR so there is not a data mismatch between what is modeled and what is used in the test data. Initial experiments found that retraining while using VFR actually showed a decrease in accuracy. However, it is possible this was due to the elimination of too many training vectors to adequately train the models.

In adaptive-rate decimation, the fixed-rate decimation factor is adapted on each input utterance to produce a fixed number of vectors for scoring. For example, for a target output of 500 vectors, the decimation rate for an input utterance with 2000 vectors would be adapted to 4. Whereas the decimation rate would be 2 for an input utterance with 1000 vectors. This has the advantage of producing a fixed number of vectors for scoring regardless of the input utterance duration. This type of decimation would be most valuable when there is an absolute hard upper-limit to computational resources.

4. RESULTS

4.1 Corpus

Experiments were conducted on the NIST 1998 summer development evaluation corpus following the

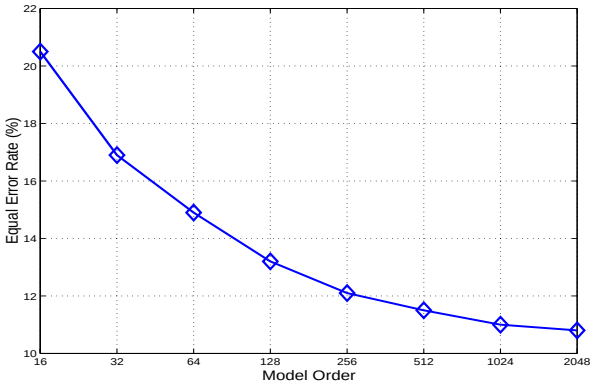

Figure 1: EER versus UBM model order.

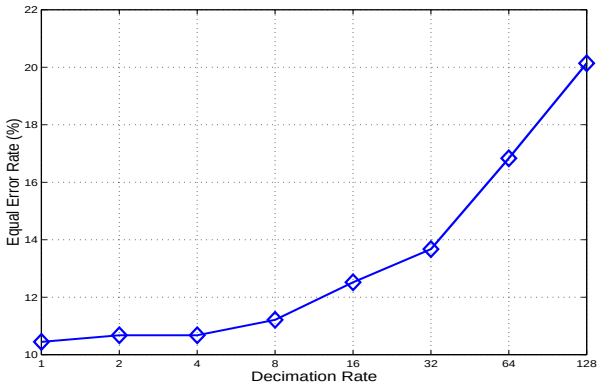

Figure 2: EER versus fixed decimation factors.

NIST evaluation plan [2]. This corpus is derived from the larger Switchboard-II phase 2 conversational telephone corpus collected by the Linguistic Data Consortium (LDC) [3]. The NIST corpus consists of 3509 male test files and 3653 female test files of variable durations between 0.5 and 60.7 seconds (normally distributed with a mean of 32 seconds ad a standard deviation of 11 seconds). Test files were scored against 10 or 20 speaker models drawn from a pool of 250 male models and 250 female models. Only same sex tests were performed (e.g., male test files were only scored against male models). Speaker models were trained with 2 minutes of speech derived from two different conversations collected from the same phone number. For computing results, all test scores were used for a total of approximately 5000 target trials and 100000 nontarget trials.

4.2 Model Order Reduction

In Figure 1 we show the EER versus UBM mixture order for eight different orders (16, 32, 64, 128, 256, 512, 1024, 2048). Relative to our baseline of 2048 mixtures, the plot shows that while model orders of 16 or 32 lead to considerable loss of accuracy, the use of 1024 or even 512 mixtures leads to less than a 1% loss in EER with roughly half (or less) of the computational burden.

4.3 Reduction in Number of Frames

Figure 2 plots the ERR for different fixed rate decimation factors. Relating to the number of vectors processed, decimation by 5 retains only 20% of the input vectors, while decimation by 10 retains 10% and decimation by 20 retains only 5%. Surprisingly, despite these high degrees of decimation, accuracy does not fall off substantially. Even with a decimation factor of 10 (90% discarded), the EER only drops by less than 1% absolute. So for little performance loss we can decrease computation by approximately a factor of 10.

Figure 3 plots the EER versus threshold for VFR processing. In parenthesis next to each threshold we show the average percentage of vectors retained. One difficulty with applying VFR is that the mapping of thresholds to percent vectors retained must be learned empirically and is front-end processing dependent so it must be adjusted with any front-end changes. As with fixed rate decimation, we are able to discard a large number of vectors with surprisingly little loss in performance.

Figure 4 plots the EER versus the target number of output vectors for the adaptive-rate decimation. In parenthesis next to the target number of output vectors is the actual average number of vectors processed. The actual number of vectors processed is lower than the target number of vectors because there is a portion of test utterances which have fewer vectors than the target number of vectors. The EER behavior of adaptive rate decimation is very similar to the fixed-rate decimation as expected.

Finally, in Figure 5 we compare all three decimation techniques by plotting EER versus percent of vectors retained on average. From this plot it is clear that fixed and adaptive decimation provide better performance than VFR. As stated before, this is due to the difficulty in defining an intelligent vector selector requiring a good distance measure and threshold.

5. DISCUSSION

Decreasing the model order from 2048 in our baseline system can readily improve speed while sacrificing little in terms of accuracy. This is not very surprising since our baseline system is geared to accuracy primarily, not speed. A factor of 2 speed up with only a

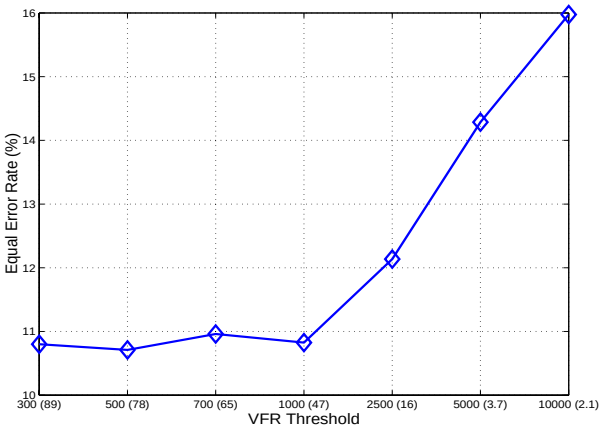

Figure 3: EER versus VFR threshold.

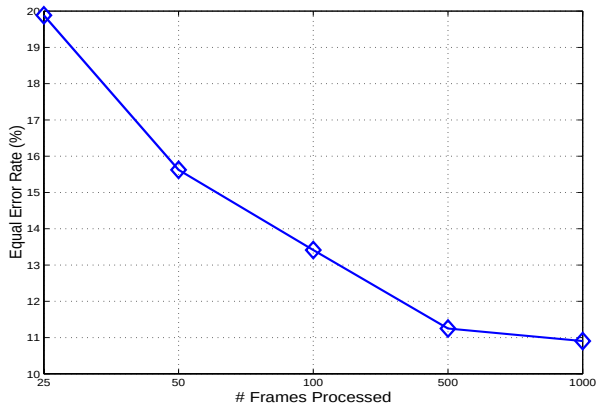

Figure 4: EER versus target number of frames for adaptive decimation.

0.2% reduction in EER is easily obtained by using a 1024 mixture.

What is surprising is the degree to which feature vectors can be decimated without loss in accuracy. Decimation by 20 yields an EER less than 2% worse than that for no decimation. On average this is using only 1.4 seconds of speech (140 vectors). However, it is clear from other experiments that performance using utterances of only 3 seconds yields very poor performance. The key factor seems to be the acoustic variety of the vectors scored, not the absolute number of vectors. Temporally consecutive vectors taken from an utterance perform worse than the same number of vectors widely spaced because there is less acoustic variety in the consecutive vectors. This implies that intelligent vector selection like VFR may have potential for improved performance with frame reduction over simple fixed decimation. This will require better selection than simple decisions based on Euclidean distances.

To examine the tradeoff of acoustic and temporal resolution reduction on accuracy and computational load, we can plot EER versus the "computational budget" (defined as the product of the number of UBM mixtures and the average number of vectors scored). In Figure 6 we show two curves of EER versus budget. The upper curve is derived from reducing the model order while leaving

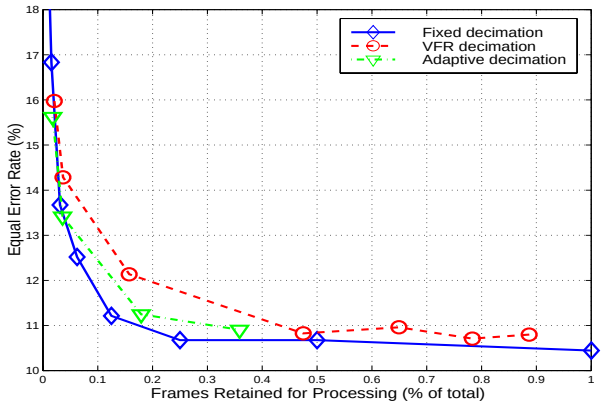

Figure 5: EER versus the number of frames retained for the three types of decimation studied.

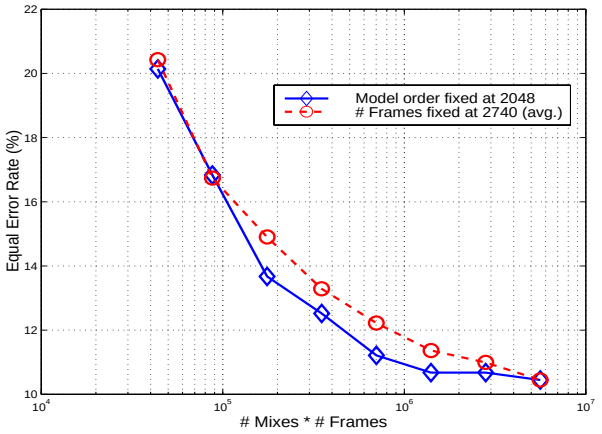

Figure 6: EER plotted against the product of the number of model mixtures and the number of frames processed.

the number of vectors unaltered (average of 2740 vectors). The lower curve is derived from reducing the number of vectors via fixed decimation and leaving the model order fixed at 2048. It is clear from this plot that for a fixed budget, is it better (from an accuracy standpoint) to use more detailed, higher order models and process fewer vectors.

6. SUMMARY

This paper has examined two main factors that dominate the computations in a GMM-UBM speaker recognition system: number of mixtures and number of vectors to score. We have shown that reduction of the UBM mixture order by a factor of 4 and decimation factors as high as 20 have little impact on speaker recognition accuracy while dramatically reducing computational load. Further speed-ups are possible with minor accuracy loss by combining both acoustic and temporal resolution reduction. In choosing between a decrease in model size and a decrease in the number of feature vectors to process, the best choice is to use a larger mixture with a reduction in vectors.

REFERENCES

[1] Reynolds, D. “Comparison of Background Normalization Methods for Text-Independent Speaker Verification,” Eurospeech 97, pp. 963-967, 1997.

[2] National Institute of Standards and Technologies, 1998 Summer Development Speaker Recognition Evaluation Plan http://www.nist.gov/speech/msrec98.html

[3] Linguistic Data Consortium, http://www.ldc.upenn.edu.