Efficient Speaker Identification and Retrieval

Hagai Aronowitz[1], *David Burshtein*[2]

[1]Department of Computer Science, Bar-Ilan University, Israel
[2]School of Electrical Engineering, Tel-Aviv University, Israel
aronowc@cs.biu.ac.il, burstyn@eng.tau.ac.il

Abstract

In this paper we present techniques for efficient speaker recognition of a large population of speakers and for efficient speaker retrieval in large audio archives. We deal with aspects of both time and storage. We use Gaussian mixture modeling (GMM) for representing both train and test sessions and show how to perform speaker recognition and retrieval efficiently with only a small degradation in accuracy compared to classic GMM based recognition. We present techniques for achieving a dramatic acceleration of both tasks. Finally, we present a GMM compression algorithm that decreases considerably the storage needed for speaker retrieval.

1. Introduction

The GMM algorithm [1-3] has been the state-of-the automatic speaker recognition algorithm for many years. The GMM algorithm calculates the log-likelihood of a test utterance given a target speaker by fitting a parametric model (a GMM) to the target training data and computing the average log-likelihood of the test-utterance feature vectors assuming frame independence. In [4, 5] a new speaker recognition technique was presented. The idea is to train GMMs not only for target speakers but also for the test sessions. The likelihood of a test utterance is approximated using only the GMM of the target speaker and the GMM of the test utterance. This technique of representing a test session by a GMM was named test utterance parameterization (TUP) and the technique of approximating the GMM algorithm using TUP was named GMM-simulation.

In [4, 5] it was shown that the GMM-simulation technique is useful for speaker retrieval in large audio archives. It was shown that the GMM-simulation algorithm speeds up dramatically the speaker retrieval task compared to classical GMM recognition and preserves the accuracy of the classical GMM recognition. In [6, 11] it was shown that the TUP framework is useful not only for reducing complexity but also for improving accuracy. The TUP framework was used for modeling intra-speaker inter-session variability resulting in a significant error reduction.

In this paper we aim to (a) improve the time and storage efficiency of the GMM-simulation algorithm, (b) address the task of performing efficient speaker recognition of a large population of speakers, and (c) show how to deal with mismatch in the length of train and test data.

The organization of this paper is as follows: we review the GMM-simulation speaker recognition algorithm in section 2. In section 3 we describe the experimental setup and the data sets. In section 4 we show how to deal with mismatch in the length of train and test data in the GMM-simulation framework. In section 5 we present techniques for efficient recognition of a large population of speakers in a speaker identification framework and for efficient speaker retrieval. In section 6 we present an algorithm for compression of GMMs used by our speaker retrieval system. Finally, we present our conclusions in section 7.

2. GMM-simulation

The basic idea of GMM-simulation algorithm is to approximate the GMM algorithm by applying the following procedure:

1. Estimate GMM Q for target speaker.
2. Estimate GMM P for test session.
3. Compute score $S=S(P, Q)$.
4. Normalize score (T-norm, Z-norm, H-norm etc.).

In [4, 5] it was shown that for a scoring function listed in equation (1), the score calculated by the GMM-simulation algorithm approximates the score calculated by the classic GMM algorithm.

$$S(P,Q)=\sum_{i=1}^{G} w_i^P \max_j \left\{ \log w_j^Q - \sum_{d=1}^{\dim} \frac{\left(\mu_{i,d}^P-\mu_{j,d}^Q\right)^2}{2\sigma_d^{\,2}} \right\} + C \quad (1)$$

In equation (1) we use the following notation:

w_i^P, w_j^Q : Weight of the Gaussian *i*/*j* of distribution *P*/*Q*.

$\mu_{i,d}^P, \mu_{j,d}^Q$: Element d of the mean vector of Gaussian *i*/*j* of distribution *P*/*Q*.

σ_d : Element (d,d) of the covariance matrix (all Gaussians share the same diagonal covariance matrix).

G: Number of Gaussians of distribution P.

dim: Dimension of the acoustic vector space.

C: Constant.

In order to accelerate the GMM-simulation algorithm, the maximization done in equation (1) is replaced by maximization over a subset of the Gaussians which is selected in advance by choosing for every Gaussian in the UBM its top-N closest Gaussians. A typical value for N is in the range of 1-20.

$$S(P,Q)\cong\sum_{i=1}^{G} w_i^P \max_{j\in \mathrm{topN}(i)} \left\{ \log w_j^Q - \sum_{d=1}^{\dim} \frac{\left(\mu_{i,d}^P-\mu_{j,d}^Q\right)^2}{2\sigma_d^{\,2}} \right\} + C \quad (2)$$

3. Experimental setup

3.1 The SPIDRE and the NIST-2004 datasets

We use the SPIDRE corpus [8] for training universal background models (UBMs) [1]. The SPIDRE corpus is a subset of the Switchboard-I corpus and consists of 5 minute conversations from land-line phones with mixed handsets.

We use the core dataset of the NIST-2004 speaker evaluation [9] for testing our techniques. The data set consists of 616 single sided conversations for training of 616 target models and 1174 single sided test conversations. All conversations are about 5 minutes long and originate from various channels and handset types. In order to increase the number of trials, every target model was tested against every test session.

In order to test our techniques on short test sessions, we have 3 testing conditions. In the first test condition we use full conversations as test sessions. In the second test condition we use the first 30 seconds speech segments (after silence removal) which are extracted from the full conversations. In the third test condition we use the first 3 seconds speech segments (after silence removal) which are extracted in the same manner.

3.2 The baseline GMM system

The baseline GMM system in this paper was inspired by the GMM-UBM system described in [1-3]. A detailed description of the baseline system can be found in [4,5]. The baseline system is based on ETSI-MFCC [7] + derivatives and an energy based voice activity detector. For all the experiments we trained 2048-component GMMs with a shared global diagonal covariance matrix. In the verification stage, the log likelihood of each test session given a target speaker is computed using the top-*N* speedup technique [3] (*N*=5) and divided by the length of the test session.

3.3 The GMM simulation system

We use the same setup of the GMM baseline for training GMMs for both train sessions and test session. For some experiments we used a smaller number of components for parameterization of test sessions. We used top-*N* pruning with *N*=10 for all experiments unless reported otherwise.

3.4 Normalization techniques

We normalize the raw scores by applying a technique we name TZ-norm. We first apply T-norm [10] and then normalize the resulting scores using non-parametric Z-norm. Non-parametric Z-norm is similar to Z-norm [2] but does not assume that the distribution of log-likelihood scores is Gaussian. According to [11], TZ-norm is significantly more accurate than T-norm.

3.5 Performance measures

We report two performance measures. The first one is equal error rate (EER) and the second one is min-DCF [9] which is the minimal value of the detection cost function (DCF) defined as:

$$DCF = 0.1*Pr(Misdetection) + 0.99\ Pr(False\ acceptance)$$

4. Recognizing short test sessions in the GMM-simulation framework

In [5] it was shown empirically that the GMM-simulation approximation is as accurate as the classic GMM algorithm in the case of testing on full conversations. However, the case where a test session is significantly shorter than the training session was not examined. An intuitive approach for dealing with such a case would be to parameterize short test sessions with smaller Gaussian mixtures. We use 128-component GMMs adapted from a 128-component UBM for representing 30sec and 3sec test sessions. We present the results of this approach in tables (1, 2).

We conclude from tables (1, 2) that decreasing the number of Gaussians in a mixture is not a suitable approach for modeling short test segments (16-24% relative degradation).

	EER (%)	min-DCF
Baseline GMM	12.7	0.047
GMM simulation 128 Gaussians	15.7	0.056
Relative degradation	24%	19%
GMM simulation 2048 Gaussians	12.9	0.048
Relative degradation	1.6%	2.1%

Table 1: GMM-simulation compared to baseline on 30-seconds test segments.

	EER (%)	min-DCF
Baseline GMM	15.3	0.058
GMM simulation 128 Gaussians	17.7	0.068
Relative degradation	16%	17%
GMM simulation 2048 Gaussians	15.5	0.059
Relative degradation	1.3%	1.4%

Table 2: GMM-simulation compared to baseline on 3-seconds test segments.

Our next approach is to parameterize short test segments by 2048-component GMMs, similarly to the method training sessions are parameterized. The results are reported in tables (1, 2). We can see that by increasing the number of Gaussians to 2048 we get a very small relative degradation (1.3-2.1%) compared to the baseline classic GMM algorithm. It may be unintuitive that a 2048-component GMM is the most suitable GMM representation for parameterization of a 3-seconds (300 frames) test session. We explain this result by the fact that GMM parameterization of a test session is effectively equivalent to clustering of the test vectors, and therefore we get better accuracy when we increase the number of clusters.

5. Improving time complexity of the GMM-simulation algorithm

We aim to speed-up two frequent tasks. The first task is identification of a large population of speakers given a single test session, and the second task is speaker retrieval in a large audio archive.

5.1 Two phase recognition

In order to accelerate the recognition algorithm there is sometimes a need to sacrifice accuracy. Usually the degradation in accuracy can be mended by a second phase of verification which is performed by running an accurate but slow algorithm on a small subset of the test sessions which pass the first phase with a relatively high score. For example, if we are interested in a false acceptance rate of 1%, we can set the false acceptance rate of the first phase to a somewhat higher false acceptance rate and filter the excessive false acceptance by running the second phase.

5.2 Top-N pruning

We use N=1 for the top-N parameter used in equation (2). Therefore the GMM-simulation score is:

$$S(P,Q)=\sum_{i=1}^{G} w_i^P \left(\log w^Q_{top1(i)} - \sum_{d=1}^{\dim} \left(\hat{\mu}^P_{i,d} - \hat{\mu}^Q_{top1(i),d} \right)^2 \right) + C \quad (3)$$

In equation (3) $\hat{\mu}^P_{i,d} / \hat{\mu}^Q_{top1(i),d}$ are defined as $\mu^P_{i,d} / \mu^Q_{top1(i),d}$ divided by σ_d. The results are summarized in table (3).

	EER (%)	min-DCF
GMM simulation N=10	13.2	0.048
GMM simulation N=1	13.2	0.049

Table 3: GMM-simulation using top-N pruning tested on full test conversations.

5.3 Gaussian pruning

In order to approximate equation (3) it is possible to sum over only a subset of the Gaussians. We propose to prune Gaussians with small weights:

$$S(P,Q)=\sum_{i=1:\, w_i>\frac{K}{G}}^{G} w_i^P \left(\log w^Q_{top1(i)} - \sum_{d=1}^{\dim} \left(\hat{\mu}^P_{i,d} - \hat{\mu}^Q_{top1(i),d} \right)^2 \right) + C \quad (4)$$

The results are listed in table (4).

K	Speed-up factor	EER (%)	min-DCF
0	1	13.2	0.048
2	7	13.2	0.049
3	14	13.7	0.051
4	25	14.1	0.052
5	40	15.0	0.055

Table 4: GMM-simulation using Gaussian pruning tested on full test conversations.

5.4 Efficient identification of a large population of speakers

We present architecture for a speaker identification system of a large population of speakers. Given a test session and a speaker population of size n, the goal is to detect all speakers whose normalized scores are better than a threshold tuned to a fixed false-acceptance rate. We denote t as the average number of test frames after silence removal (12000). We denote g as the number of components in a GMM (2048). We denote d as the dimension of the feature space (26).

We use top-N pruning (N=1) and Gaussian pruning (K=5 for system #1 and K=4 for system #2) for the first phase and then rescore test sessions that exceed a false acceptance threshold (0.05 for system #1 and 0.1 for system #2). The algorithm we use for rescoring is the GMM-simulation algorithm with top-N pruning (N=1). In table (5) we compare the average time complexity of our proposed systems to the baseline GMM and the baseline GMM-simulation algorithm. The DET curves of the various systems are presented in Figure (1).

system	Complexity	Ops. ($\times 10^6$)	speedup (n→∞)
Classic GMM	$^a gdt + {}^b 5ndt$	640+1.5n	1
GMM simulation	$^c gdt + {}^d ngd$	640+0.05n	30
Accelerated system #1	$^c gdt +$ (^{e}1/40+^{f}0.05)ngd	640+0.004n	375
Accelerated system #2	$^c gdt +$ (^{e}1/25+^{f}0.1)ngd	640+0.0075n	200

Table 5: Identification of a large population of speakers: Time complexity analysis of our proposed systems.

a UBM decoding.
b Top-5 scoring.
c Test session parameterization.
d GMM-simulation using top-1 pruning
e Phase one: GMM-simulation using top-1 and Gaussian pruning techniques.
f Phase two: rescoring using GMM-simulation (top-1).

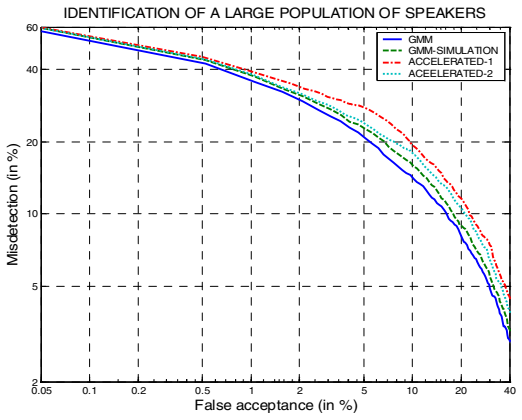

Figure 1: Comparison of the performance of our proposed systems for identification of a large population of speakers.

The time complexity of the classic GMM algorithm is linear in the size of a test session. The proposed accelerated algorithm

is not linear in the size of the test session. Therefore, the speedup will be smaller for short test sessions. However, for short test sessions we can use more aggressive Gaussian pruning because a larger proportion of the Gaussians has very small weights.

5.5 Efficient speaker retrieval

We define the speaker retrieval task as scanning a large audio for a single target speaker, allowing reasonable preprocessing (indexing) of the archive which is not taken into account when assessing complexity. Our proposed architecture for speaker retrieval is similar to the architecture described in the previous subsection with the exception of parameterization of the test session in the pre-processing phase. In table (6) we compare the average time complexity of our proposed algorithm to the baseline GMM and to the GMM-simulation algorithm.

system	Complexity	Ops.($x10^6$)	speedup
Classic GMM	gdt	640	1
GMM simulation	gd	0.05	12,000
Accelerated system #1	$(1/40+0.05)gd$	0.004	160,000
Accelerated system #2	$(1/25+0.1)gd$	0.0075	86,000

Table 6: Speaker retrieval: Time complexity analysis of our proposed systems.

6. GMM compression

Our proposed algorithm for speaker retrieval requires storing a GMM for every audio file in the archive. In order to reduce the size of the stored GMMs, we need to be able to compress GMMs.

In [12] an algorithm for compression of GMMs was presented. The main idea is to exploit the fact that all GMMs are adapted from the same UBM. The first stage is therefore to compute a difference between a GMM and the UBM and then to quantize those parameters (of the difference GMM) which are significantly not equal to zero. The quantization is done independently for every mean coefficient, variance coefficient and every weight.

In this paper we propose a different way to compress GMMs. We optimize our compression algorithm to the speaker retrieval task and the GMM-simulation algorithm. We need only to compress the weights and the mean vectors. The weights are compressed by similar techniques as in [12]. The means however are compressed by using vector quantization. We compute GMMs for sessions in a development set and then pull together all the difference vectors between every mean vector and the corresponding UBM mean vector. We cluster these vectors into 60,000 clusters. In order to compress a GMM we just replace every mean vector by its cluster index. This compression algorithm results in a 1:50 compression rate but with some degradation in accuracy. In order to eliminate the degradation in accuracy we locate badly quantized Gaussians (10% in average) by calculating for every mean vector the product of its weight and its quantization error. The badly quantized Gaussians are compressed by quantization of every coefficient independently to 4 bits. The compression factor of our algorithm is 30 (7KB for a single GMM).

7. Conclusions

In this paper we have presented an algorithm for efficient and accurate speaker recognition. The algorithm is based on the GMM-simulation algorithm and is useful for both identification of a large population of speakers and for speaker retrieval. For example, assuming a false acceptance rate of 1%, we get a speedup factor of 3.3 for identification of 1,000 speakers and a speedup factor 23 for 10,000 speakers. For the speaker retrieval task we get a speedup factor of 160,000. We verified that our techniques are also suitable when testing on short test sessions. Finally, we presented an algorithm for GMM compression which is used to compress the index built by our speaker retrieval algorithm.

8. Acknowledgements

This research was supported by Muscle, a European network of excellence funded by the EC 6th framework IST programme.

9. References

[1] Reynolds D. A., Quatieri T. F. and Dunn R. B., "Speaker verification using adapted Gaussian mixture models", *Digital Signal Processing*, Vol. 10, No.1-3, pp. 19-41, 2000.

[2] Reynolds, D. A., "Comparison of background normalization methods for text-independent speaker verification", in Proc. *Eurospeech*, pp.963-966, 1997.

[3] McLaughlin J., Reynolds D. A., and Gleason T., "A study of computation speed-ups of the GMM-UBM speaker recognition system", in Proc. *Eurospeech*, pp.1215-1218, 1999.

[4] Aronowitz H., Burshtein D. and Amir A., "Speaker indexing in audio archives using Gaussian mixture scoring simulation", in *MLMI:* Proceedings of the Workshop on Machine Learning for Multimodal Interaction, Springer-Verlag LNCS, 2004.

[5] Aronowitz H., Burshtein D. and Amir A,, "Speaker indexing in audio archives using test utterance Gaussian mixture modeling", in Proc. of *ICSLP*, 2004.

[6] Aronowitz H., Burshtein D. and Amir A., "Speaker indexing in audio archives using test utterance Gaussian mixture modeling", in Proc. of *ICASSP* 2005.

[7] "Speech processing, transmission and quality aspects (stq); distributed speech recognition; front-end feature extraction algorithm; compression algorithms," ETSI Standard: ETSI-ES-201-108-v1.1.2, 2000, http://www.etsi.org/stq.

[8] Linguistic Data Consortium, SPIDRE documentation file, http://www.ldc.upenn.edu/Catalog/readme_files/.

[9] "The NIST Year 2004 Speaker Recognition Evaluation Plan", http://www.nist.gov/speech/tests/spk/2004/.

[10] Auckenthaler R., Carey M., and Lloyd-Thomas H., "Score normalization for text-independent speaker verification systems", *Digital Signal Processing*, vol. 10, pp. 42-54, 2000.

[11] Aronowitz H., Irony D., Burshtein D. "Modeling Intra-Speaker Variability for Speaker Recognition", in Proc. of *Interspeech*, 2005.

[12] Reynolds D. A., "Model compression for GMM based speaker recognition systems", in Proc. of *Eurospeech*, pp. 2005-2008, 2003.