Speaker Model Clustering for Efficient Speaker Identification in Large Population Applications

Vijendra Raj Apsingekar and
Phillip L. De Leon, *Senior Member, IEEE*

***Abstract*—In large population speaker identification (SI) systems, likelihood computations between an unknown speaker's feature vectors and the registered speaker models can be very time-consuming and impose a bottleneck. For applications requiring fast SI, this is a recognized problem and improvements in efficiency would be beneficial. In this paper, we propose a method whereby GMM-based speaker models are clustered using a simple *k*-means algorithm. Then, during the test stage, only a small proportion of speaker models in selected clusters are used in the likelihood computations resulting in a significant speed-up with little to no loss in accuracy. In general, as the number of selected clusters is reduced, the identification accuracy decreases; however, this loss can be controlled through proper tradeoff. The proposed method may also be combined with other test stage speed-up techniques resulting in even greater speed-up gains without additional sacrifices in accuracy.**

***Index Terms*—Clustering methods, speaker recognition.**

I. Introduction

The objective of speaker *identification* (SI) is to determine which voice sample from a set of known voice samples best matches the characteristics of an unknown input voice sample [1]. SI is a two-stage procedure consisting of training and testing. In the training stage, M speaker-dependent feature vectors $\mathbf{x}_m^{\text{train}}$ are extracted from a training speech signal and a speaker model $\boldsymbol{\lambda}_s$ is built for each speaker's feature vectors. Normally, SI systems use the Mel-frequency cepstral coefficients (MFCCs) as the $L \times 1$ feature vector and a Gaussian mixture model (GMM) of the feature vectors for the speaker model. The GMM is parameterized by the set $\{w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$ where w_i are the weights, $\boldsymbol{\mu}_i$ are the mean vectors, and $\boldsymbol{\Sigma}_i$ are the covariance matrices of the W Gaussian component densities of the GMM. In the SI testing stage, M' feature vectors $\mathbf{x}_m^{\text{test}}$ are extracted from a test signal (speaker unknown), scored against all S speaker models using a log-likelihood calculation, and the most likely speaker identity $\hat{s}$ decided according to

$$\hat{s} = \arg \max_{1 \le s \le S} \sum_{m=1}^{M'} \log p\left(\mathbf{x}_m^{\text{test}} | \lambda_s\right). \tag{1}$$

In assessing an SI system, we measure identification accuracy as the number of correct identification tests divided by the total number of tests. For many years now, GMM-based systems have been shown to be very successful in accurately identifying speakers from a large population [1], [2].

In speaker *verification* (SV), the objective is to verify an identity claim. Although the SV training stage is identical to that for SI, the test stage differs. In the SV test stage, for the given test feature vectors a likelihood ratio is formed from the claimant model and that of a background model. If the likelihood ratio is greater than a threshold value, the claim is accepted otherwise it is rejected. In SV, maximum *a posteriori* (MAP) adapted speaker models from a universal background model (UBM) with likelihood normalization are normally used [3]. There are also more advanced techniques used in SV such as support vector machine (SVM) generalized linear discriminant sequence (GLDS) [4] and SVM-supervectors [5].

In this correspondence, we consider the problem of slow speaker *identification* for large population systems. In such SI systems (and SV systems as well), the log-likelihood computations required in (1) have been recognized as the bottleneck in terms of time complexity [2], [6]. Although accuracy is always the first consideration, efficient identification is also an important factor in many real-world systems and other applications such as speaker indexing and forensic intelligence [7], [8].

Among the earliest proposed methods to address the slow SI/SV problem were pre-quantization (PQ) and pruning. In PQ, the test feature vectors are first compressed through subsampling (or another method) before likelihood computations [9]; fewer feature vectors directly translate into faster SV/SI. It has been found that reducing the test feature vectors by a factor as high as 20 does not affect SV performance [9]. Application of PQ in order to speed-up SI was investigated in [2] and was found to result in a real-time speed-up factor of as high as 5× with no loss in identification accuracy using the TIMIT corpus. In pruning, a small portion of the test feature vectors is compared against all speaker models and those speaker models with the worst scores are pruned out of the search space [10]. In subsequent iterations, other portions of the test feature vectors are used and speaker models are scored and pruned until only a single speaker model remains resulting in an identification. Using the TIMIT corpus, a speed-up factor of 2× has been reported with pruning [2]. Variants of PQ and pruning as well as combinations of the methods applied to efficient SI/SV were extensively evaluated using TIMIT and NIST1999 corpora in [2]. In [5], a GMM supervector kernel for SVM-based SV was proposed in which the test speech is adapted to a UBM and the mean vectors of the adapted UBM are used as supervectors. A kernel is designed in which an inner product between the target model and supervector is computed to obtain a score. Though the scoring is fast, test stage adaptation may require significant time but details are not provided.

In [11], a hierarchical speaker identification (HSI) was proposed that uses *speaker clustering* which, for HSI purposes, refers to the task of grouping together feature vectors from different speakers and modeling the superset, i.e., a speaker cluster GMM. (In most other papers such as [12], the term "speaker clustering" refers to the task of grouping together unknown speech utterances based on a single speaker's voice characteristics which is entirely different than what is done in [11].) In HSI, a non-Euclidean distance measure between an individual speaker's GMM and the cluster GMMs is used to assign speakers to a cluster. Feature vectors for intra-cluster speakers are recombined, cluster GMMs are rebuilt, distance measures are recalculated, and speakers are reassigned to "closer" clusters. The procedure iterates using the ISODATA algorithm until speakers have been assigned to an appropriate cluster. During the test stage, the cluster/speaker model hierarchy is utilized: first log-likelihoods are computed against the given cluster GMMs in order to select the appropriate cluster for searching. Then log-likelihoods are computed against those speaker models in the cluster in order to identify the speaker.

Using a 40-speaker corpus, HSI requires only 30% of the calculation time (compared to conventional SI) while incurring an accuracy loss of less than 1% (details of the corpus and procedure for timing are not described). Unfortunately, HSI has a number of drawbacks including an extremely large amount of computation (which the authors

Manuscript received February 13, 2008; revised November 18, 2008. Current version published April 03, 2009. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Simon King.

The authors are with the Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM 88003 USA (e-mail: vijendra@nmsu.edu; pdeleon@nmsu.edu).

Digital Object Identifier 10.1109/TASL.2008.2010882

acknowledge) required for clustering. Because of this required computation, the HSI method does not scale well with large population size. Although HSI was shown to speed up SI with little accuracy loss, the small number of speakers used in simulation does not provide any indication of how accuracy would degrade with much larger populations [13].

A similar idea for reducing a search space using clusters or classes has long been used in the area of content-based image retrieval (CBIR) [14]. In this application, only those images within a few predetermined classes that are similar to the query image are searched rather than searching the entire image database. Although hierarchical and structural arrangements of GMM-UBMs have been proposed in order to speed-up SV including those in [15], [16], it appears that [11] was one of the first to use clusters for speeding up SI. Finally, speaker clusters (as defined in [12]) have been used for fast speaker adaptation in speech recognition applications [17], speaker indexing [18], and in the open-set speaker identification (OSI) problem [19].

In a recent publication, a different approach toward efficient SV/SI has been investigated. In [6], the authors *approximate* the required log-likelihood calculations in (1) with an approximate cross entropy (ACE) between a GMM of the test utterance and the speaker models; speed-up gains are realized through reduced computation in ACE. The authors acknowledge potential problems with constructing a GMM of the test signal and offer methods to reduce this bottleneck. Also, if the test signal is short the GMM may not be accurate. Evaluation of MAP-ACE to the baseline SV system indicates no significant accuracy differences; however, no information regarding actual speed-up (as compared to the baseline SV system) is given [6]. SV using MAP-ACE with Gaussian pruning, results in a speed-up factor of $3\times$, $8\times$ with a 0.1%, 0.66% degradation in equal error rate (EER) when compared to MAP-ACE with no pruning. For SI systems (VQ-tree-based, GMM-UBM) using ACE with top-N pruning, the authors report a theoretical speed-up of $43\times$ for 100 speakers; however, accuracy results and actual speed-ups are not provided [6].

In this paper, the focus is *strictly* on efficient speaker *identification* and we propose the use of training stage clustering methods in order to reduce test stage log-likelihood calculations. Our work differs from [11] in two regards. First, rather than iteratively grouping feature vectors from different speakers and modeling the whole cluster with a GMM, we form clusters directly from the individual speaker models which we term "speaker *model* clustering." This difference is important as it allows utilization of the simple k-means algorithm and leads to a scalable method for clustering which we demonstrate using large population corpora. Second, we investigate searching more than one cluster so that any loss in identification accuracy due to searching too few clusters can be controlled; this allows a smooth tradeoff between speed and accuracy. Our work also differs from [6] in that we make no approximations to (1) relying instead on a reduction in the number of speaker models that (1) has to be calculated against for the speed-up. In addition, whereas the majority of the results presented in [6] are for SV, our focus is on efficient SI. Finally, since the proposed speaker model clustering is applied at the training stage (after speaker modeling), it can be combined with test stage speed-up methods such as PQ, pruning, and ACE, resulting in even greater speed increases.

This paper is organized as follows. In Section II, we describe application of the k-means algorithm for clustering GMM speaker models and criteria for which clusters to search. In Section III, we describe the experimental evaluation and provide results using several large population corpora with different channels (TIMIT, NTIMIT, and NIST 2002) [2], [6]. In Section IV we conclude the article.

II. Speaker Model Clustering

In an SI system for a large and acoustically diverse population, only a few speaker models actually give large log-likelihood values for (1). In fact, the basis for speaker pruning is to quickly eliminate speaker models for which it is clear the log-likelihood score is going to be low thus reducing unnecessary computation in (1) [10]. In this correspondence, we propose that speaker models be clustered during the training stage (after speaker modeling); during the test stage only those clusters likely to contain a high-scoring speaker model will be considered. Ideally, the speaker models are clustered according to a distance measure based on log-likelihood due to the decision rule in (1). However, a direct method of determining clusters, taking into account all speaker models and training feature vectors (which would provide the log-likelihood measure), leads to a difficult nonlinear optimization problem.

In order to develop clustering methods which are based on the k-means algorithm and can scale with population size, we propose three configurations based on a cluster center or centroid definition and a distance measure from λ_s to the center or centroid; a fourth configuration uses a distance measure based on an alternate speaker model. In addition, each configuration includes a criterion for cluster selection.

A. Euclidean Distance-Based Clustering

The first configuration is based on a Euclidean distance measure and designed for simplicity. We begin by representing the GMM-based speaker model simply as a point in L-dimensional space determined by the weighted mean vector (WMV) [20]

$$\bar{\boldsymbol{\mu}} = \sum_{i=1}^{W} w_i \boldsymbol{\mu}_i. \tag{2}$$

The WMV can be thought of geometrically as the centroid of the speaker model and gives an approximation for *position* in the speaker model space. The WMV can also be thought of as a *vectorization* of the speaker model. From (2), one can define the centroid of a cluster of GMM speaker models as

$$\mathbf{r} = \frac{1}{K} \sum_{k=1}^{K} \bar{\boldsymbol{\mu}}_k \tag{3}$$

where $\bar{\boldsymbol{\mu}}_k$ is the WMV for λ_k, and K is the number of speaker models in the cluster. Fig. 1 gives an illustration of the speaker model space. We use a Euclidean distance measure from speaker model s to the cluster centroid $\mathbf{r}_n$ defined by [20]

$$d_1(\lambda_s, \mathbf{r}_n) = \left[(\bar{\boldsymbol{\mu}}_s - \mathbf{r}_n)^T (\bar{\boldsymbol{\mu}}_s - \mathbf{r}_n) \right]^{1/2}. \tag{4}$$

The algorithm for speaker model clustering using the centroid definition in (3) and distance measure in (4) is given in Algorithm 1.

Algorithm 1 Euclidean distance-based speaker model clustering

1: Initialize cluster centroids $\mathbf{r}_n = \bar{\boldsymbol{\mu}}_n, 1 \leq n \leq N$ using randomly chosen speaker models.
2: Compute distance using (4) from λ_s to $\mathbf{r}_n, 1 \leq s \leq S$.
3: Assign each λ_s to the cluster with the minimum distance.
4: Compute new cluster centroids $\mathbf{r}_n$ using (3).
5: Goto step 2 and terminate when cluster membership does not change.

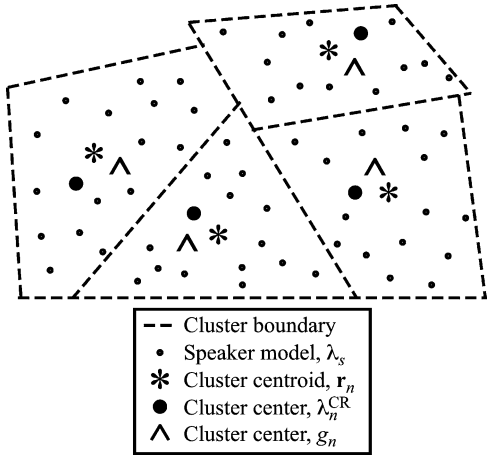

Fig. 1. Space of speaker models, clusters and three cluster centroid definitions.

In order to select the cluster that will be searched in the test stage, the average of the test feature vectors from the unknown speaker is computed as

$$\bar{\mathbf{x}}^{\mathrm{test}} = \frac{1}{M'} \sum_{m=1}^{M'} \mathbf{x}_m^{\mathrm{test}} \tag{5}$$

and cluster C_n whose centroid is nearest (Euclidean distance) to this average is selected as

$$C_n = \arg \min_{1 \le n \le N} (\bar{\mathbf{x}}^{\mathrm{test}} - \mathbf{r}_n)^T (\bar{\mathbf{x}}^{\mathrm{test}} - \mathbf{r}_n). \tag{6}$$

B. Kullback–Leibler, GMM-Based Clustering

Equations (2)–(4) provide a simple approach toward k-means-based speaker model clustering; however, the SI decision in (1) *is* based on log-likelihood and not on a Euclidean distance measure to the GMM. If the centroid is based on a distributional parameterization then a more appropriate distance measure such as Kullback–Leibler (KL) divergence may be used. Therefore, for the second configuration, we define the cluster center as the GMM speaker model λ_u^{CR} which is nearest to $\mathbf{r}_n$

$$\lambda_n^{\mathrm{CR}} = \arg \min_{1 \le s \le S} d_1(\lambda_s, \mathbf{r}_n), \quad 1 \le n \le N. \tag{7}$$

This speaker model, called the "cluster representative" (CR) and illustrated in Fig. 1, serves to reduce the cluster to its most representative element [21]. Although we would like to use KL divergence from λ_s to λ_n^{CR} as the distance measure in k-means, there is currently no known closed-form expression between GMMs. However, one proposed method to approximate KL divergence between two speaker models uses actual acoustic data (feature vectors) [22]. Following the approach in [22], we propose a second distance measure used for speaker model clustering by approximating the KL divergence from λ_s to λ_n^{CR} with

$$d_2\left(\lambda_s, \lambda_n^{\mathrm{CR}}\right) \approx \frac{1}{M} \sum_{m=1}^{M} \log p\left(\mathbf{x}_{s,m}^{\mathrm{train}} | \lambda_s\right) - \frac{1}{M} \sum_{m=1}^{M} \log p\left(\mathbf{x}_{s,m}^{\mathrm{train}} | \lambda_n^{\mathrm{CR}}\right). \tag{8}$$

where M is the number of training feature vectors and $\mathbf{x}_{s,m}^{\mathrm{train}}$ are the training feature vectors for speaker s. The use of a CR overcomes the problem of the centroid $\mathbf{r}_n$ not having distributional parameters to compute KL divergence. The algorithm for speaker model clustering using the cluster center definition in (7) and distance measure in (8) is given in Algorithm 2. We refer to this as "KL GMM-based clustering."

Algorithm 2 KL GMM-based Speaker model clustering

1: Initialize cluster centers $\lambda_n^{\mathrm{CR}}, 1 \le n \le N$ using randomly chosen speaker models.
2: Compute distance using (8) from λ_s to $\lambda_n^{\mathrm{CR}}, 1 \le s \le S$.
3: Assign each λ_s to the cluster with the minimum distance.
4: Compute $\mathbf{r}_n$ and new cluster centers λ_n^{CR} using (7).
5: Goto step 2 and terminate when cluster membership does not change.

Alternatively, we can use the symmetric version of (8) to measure "distance" from λ_n^{CR} to λ_s [23]

$$d_{2\mathrm{sym}}\left(\lambda_n^{\mathrm{CR}}, \lambda_s\right) \approx \frac{1}{M} \sum_{m=1}^{M} \log p\left(\mathbf{x}_{n,m}^{\mathrm{CR}} | \lambda_n^{\mathrm{CR}}\right) - \frac{1}{M} \sum_{m=1}^{M} \log p\left(\mathbf{x}_{n,m}^{\mathrm{CR}} | \lambda_s\right) \tag{9}$$

where $\mathbf{x}_{n,m}^{\mathrm{CR}}$ are the training feature vectors for cluster representative n. The algorithm for clustering using the above is identical to Algorithm 2 except that in Step 2, distance is computed with (9). We refer to this as "KL (symmetric) GMM-based clustering."

In the test stage, we select the cluster whose log-likelihood, measured against λ_n^{CR}, is large [20]

$$C_n = \arg \max_{1 \le n \le N} \left[\sum_{m=1}^{M'} \log p\left(\mathbf{x}_{s,m}^{\mathrm{test}} | \lambda_n^{\mathrm{CR}}\right) \right]. \tag{10}$$

C. Kullback–Leibler, Gaussian-Based Clustering

In the third configuration, we define the cluster center with an L-dimensional Gaussian distribution $g : \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Theta})$ of the speakers' (within the cluster) training feature vectors where $\boldsymbol{\nu}$ is the mean vector and $\boldsymbol{\Theta}$ is the covariance matrix. We use KL divergence as the distance measure from λ_s to cluster center g_n in k-means, approximated as [22]

$$\begin{aligned} d_3(\lambda_s, g_n) &\approx \sum_{i=1}^{W} w_i \mathrm{KL}(f_i \| g_n) \\ &\approx \frac{1}{2} \sum_{i=1}^{W} w_i \left[\log \frac{|\boldsymbol{\Theta}|}{|\boldsymbol{\Sigma}_i|} + \mathrm{tr}\left(\boldsymbol{\Theta}^{-1} \boldsymbol{\Sigma}_i^{-1}\right) + (\boldsymbol{\mu}_i - \boldsymbol{\nu})^{\mathsf{T}} \boldsymbol{\Theta}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\nu}) - L \right] \end{aligned} \tag{11}$$

where $f_i : \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ is the ith component density of λ_s. The algorithm for speaker model clustering using the Gaussian cluster center definition and distance measure in (11) is given in Algorithm 3.

Algorithm 3 KL Gaussian-based Speaker model clustering

1: Randomly assign speakers to one of N clusters.
2: Using training feature vectors of intra-cluster speakers, compute $\boldsymbol{\nu}, \boldsymbol{\Theta}$ for cluster center g_n.
3: Compute distance using (11) from λ_s to $g_n, 1 \le s \le S$.
4: Assign each λ_s to the cluster with the minimum distance.
5: Goto step 2 and terminate when cluster membership does not change.

In the test stage, we select the cluster whose log-likelihood, measured against g_n, is large

$$C_n = \arg \max_{1 \le n \le N} \left[\sum_{m=1}^{M'} \log p \left(\mathbf{x}_{s,m}^{\text{test}} | g_n \right) \right]. \tag{12}$$

D. Log-Likelihood, Gaussian-Based Clustering

In the fourth configuration, the cluster center is defined as in Section II-C and $\mathbf{x}_{s,m}^{\text{train}}$ are modeled as $h_s : \mathcal{N}(\boldsymbol{\xi}, \boldsymbol{\Psi})$, where $\boldsymbol{\xi}$ is the mean vector and $\boldsymbol{\Psi}$ is the covariance matrix. The distance measure is based on the log-likelihood between $\mathbf{x}_{s,m}^{\text{train}}$ (modeled as h_s) and cluster center g_n

$$\begin{aligned} d_4 \left(\mathbf{x}_{s,m}^{\text{train}}, g_n \right) &= - \sum_{m=1}^{M} \log p \left(\mathbf{x}_{s,m}^{\text{train}} | g_n \right) \\ &= \frac{ML}{2} \log(2\pi) + \frac{M}{2} \log |\boldsymbol{\Theta}| \\ &\quad + \frac{1}{2} \sum_{m=1}^{M} \left[\left(\mathbf{x}_{s,m}^{\text{train}} - \boldsymbol{\nu} \right)^T \boldsymbol{\Theta}^{-1} \left(\mathbf{x}_{s,m}^{\text{train}} - \boldsymbol{\nu} \right) \right] \\ &\approx \frac{ML}{2} \log(2\pi) + \frac{M}{2} \log |\boldsymbol{\Theta}| + M \text{tr}(\boldsymbol{\Psi} \boldsymbol{\Theta}^{-1}) \\ &\quad + \text{tr} \left[(\boldsymbol{\xi} - \boldsymbol{\nu})(\boldsymbol{\xi} - \boldsymbol{\nu})^T \boldsymbol{\Theta}^{-1} \right]. \end{aligned} \tag{13}$$

We use a minus log-likelihood for proper clustering based on minimum distance or equivalently, maximum log-likelihood. The algorithm for clustering using the above configuration is similar to Algorithm 3 except that in step 3, distance is computed as in (13) and in step 4, λ_s is now h_s. In the test stage, we select the clusters to search according to (12).

E. Searching a Subset of Clusters

Rather than selecting a single cluster to search using criteria in (6), (10), or (12) we can also use a subset of clusters ranked according to these equations. Using a subset of clusters allows a smooth tradeoff between accuracy loss (due to searching too few clusters) and speed. All three cluster selection methods provide relatively fast and efficient ways to select clusters for searching which is an important consideration for test stage processing. Finally, we note that a GMM of the test feature vectors λ^{test} could be constructed as in [6] and clusters selected according to (9) using λ^{test} for λ_s. However, we found the time in computing the test GMM with the iterative EM algorithm as well as likelihood calculations required for cluster selection to exceed the time required when using the above cluster selection methods and not produce any better results. Furthermore, if the test signal is short, the GMM may not be sufficiently accurate enough to properly select clusters. Both of these issues were described in [6].

III. Experiments and Results

Our SI system is based on the system in [2] in order to facilitate comparisons. To demonstrate the applicability of the methods proposed in Section II to a wide variety of GMM-based SI systems, we have added to this system some additional elements such as delta MFCCs, cepstral mean subtraction (CMS), and RASTA processing depending on the corpus being used. Specifically, our baseline system uses an energy-based voice activity detector to remove silence; feature vectors composed of 29 MFCCs for TIMIT and 13 MFCCs + 13 delta MFCCs for NTIMIT and NIST 2002 extracted every 10 ms using a 25-ms window; CMS and RASTA processing on NIST 2002 [24]; and $W = 32$ component densities for the GMMs. For TIMIT/NTIMIT, we use approximately 24-s training signals and 6-s test signals and for NIST 2002 (one speaker detection cellular task) we use approximately 90-s training signals and 30-s test signals. With a complete

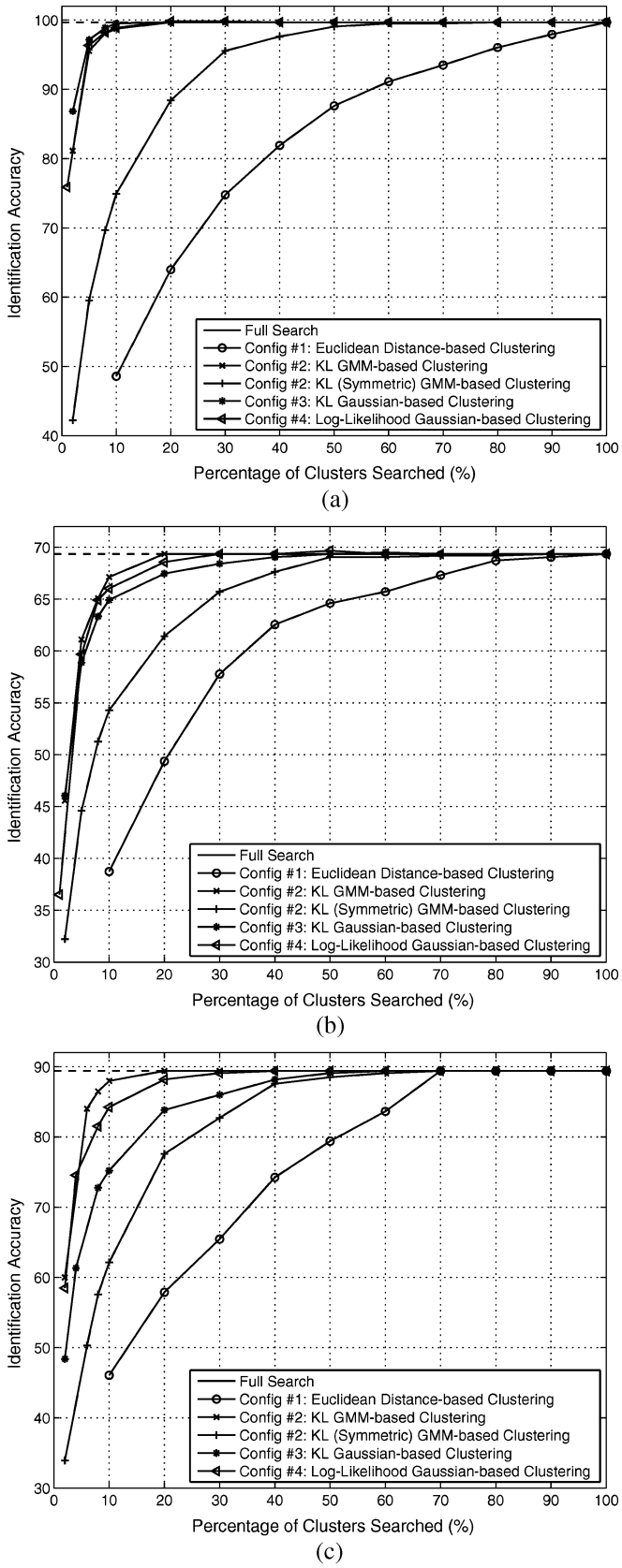

Fig. 2. Speaker identification accuracy versus percentage of clusters searched for (a) TIMIT, (b) NTIMIT, and (c) NIST 2002.

TABLE I
AVERAGE SPEED-UP FACTORS USING KL GMM-BASED CLUSTERING RELATIVE TO BASELINE SYSTEM. SI ACCURACIES FOR TIMIT, NTIMIT, AND NIST 2002 ARE NOTED FOR EACH COLUMN

Testing Method	10% of Clusters Searched (98.73%, 67.14%, 87.96%)	20% of Clusters Searched (99.68%, 69.37%, 89.39%)	100% of Clusters Searched (99.68%, 69.37%, 89.39%)
Clustering only	8.7×	4.4×	1.0×
Clustering + Pr ($R = 6$)	8.8×	6.5×	4.3×
Clustering + PQ ($P = 15$)	118.1×	62.5×	14.8×
Clustering + PQ ($P = 7$) + Pr ($R = 5$)	149.3×	74.0×	31.5×

calculation of (1), i.e., full search, our system has baseline identification accuracy of 99.68%, 69.37% for the 630-speaker TIMIT, NTIMIT corpus as shown by the dashed lines in Fig. 2(a) and (b), respectively. These values agree with values published in recent literature [2]. For the 330-speaker NIST 2002 corpus, the baseline accuracy is 89.39% as shown by the dashed line in Fig. 2(c).

A. Evaluation of Proposed Clustering Methods

We partitioned the speaker model space into N clusters using a range of values for N (guided by the silhouette index) and measured SI accuracy rates. We found $N = 100$ to give good performance with the TIMIT/NTIMIT corpora and $N = 50$ with the NIST 2002 corpus. We also found that the KL Gaussian-based clustering was very sensitive to initialization. In order to evaluate the proposed approach, we measure SI accuracy as a function of the percentage of clusters searched as shown in Fig. 2. This percentage is an approximation to the search space reduction in (1), since the number of speaker models in each cluster are not exactly the same but are more or less equally distributed. In evaluating the four configurations, we find that KL GMM-based clustering generally produces the highest SI accuracy results. For this configuration, we are able to search as few as 10% of the clusters and incur a 0.95%, 2.2%, and 1.4% loss in SI accuracy with the TIMIT, NTIMIT, and NIST 2002 corpora respectively; searching 20% of the clusters resulted in no accuracy loss.

B. Speed-Up Results

As described in Section I, the proposed method of speaker model clustering is applied during the training stage (after speaker modeling) and can be combined, as have other proposed speed-up methods, with test stage techniques such as PQ and pruning [2], [6]. Although many sophisticated pruning algorithms exist for both SV and SI, we use a simple static pruning algorithm which eliminates half of the speaker models at each pruning stage in order to illustrate the potential gain [2]. For this work, we benchmark using KL GMM-based clustering since the SI accuracies were the highest; KL Gaussian-based clustering was slightly faster but accuracies, as shown in the previous subsection, were lower. We searched 10% and 20% of the clusters and adjusted the PQ decimation factor P and number of pruning stages R so that the SI accuracies were the same over the testing methods. The speed-up factors (shown in Table I) were computed by carefully timing the test stage for a simulation involving the complete corpus and determining the average time for a single SI. These actual times were then normalized against the average time for a baseline SI (no clustering, PQ, or pruning). Speed-up gains using only PQ and/or pruning can be evaluated from the data in column 4 of Table I since searching 100% of the clusters amounts to using all speaker models. When using 10%, 20% of the clusters, the search space is reduced by a factor of 10×, 5× and the realized speed-up factor (average of three corpora) is 8.7×, 4.4×, respectively. The difference between the search space reduction and realized speed-up gain is due to additional computation involved in the cluster section and other overheads. Gains using only the clustering method are on par with gains using only PQ or pruning; adding PQ and/or pruning to the clustering method further speeds up SI consistent with previous research results [2].

IV. CONCLUSION AND FUTURE RESEARCH

In speaker identification, log-likelihood calculations in the test stage have been recognized as the bottleneck in terms of time complexity. In this paper, we have proposed a method whereby GMM-based speaker models are clustered using a simple k-means algorithm. Then, during the test stage, only a small proportion of speaker models in selected clusters are used in the likelihood computations resulting in a significant speed-up with little to no loss in accuracy. For the TIMIT, NTIMIT, and NIST 2002 corpora, we are able to search as few as 10% of the speaker model space and realize an actual speed-up of 8.7× with only a small loss in accuracy; searching 20% or more clusters results in accuracies equivalent to the full search. Using the proposed clustering method together with other speed-up methods results in actual speed-up factors as high as 74× with no loss in accuracy; speed-up factors as high as 149× are possible with a slight loss in accuracy.

REFERENCES

[1] D. Reynolds and R. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Signal Process.*, vol. 3, no. 1, pp. 72–83, Jan. 1995.
[2] T. Kinnunen, E. Karpov, and P. Franti, "Real-time speaker identification and verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 1, pp. 277–288, Jan. 2006.
[3] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Process.*, vol. 10, pp. 19–41, 2000.
[4] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and J. Navratil, "The MIT-LL/IBM 2006 speaker recognition system: High-performance reduced-complexity recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2007, vol. IV, p. 217–220.
[5] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, "SVM based speaker verification using GMM supervector kernel and NAP variability compensation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2006, vol. I, pp. 97–100.
[6] H. Aronowitz and D. Burshtein, "Efficient speaker recognition using approximated cross entropy (ACE)," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 7, pp. 2033–2043, Sep. 2007.
[7] J. Makhoul, F. Kubala, T. Leek, L. Daben, N. Long, R. Schwartz, and A. Srivastava, "Speech and language technologies for audio indexing and retrieval," *Proc. IEEE*, vol. 88, no. 8, pp. 1138–1353, Aug. 2000.
[8] H. Aronowitz, D. Burshtein, and A. Amir, "Speaker indexing in audio archives using test utterance Gaussian mixture modeling," in *Proc. IEEE Int. Conf. Spoken Lang. Process. (ICSLP)*, 2004, pp. 609–612.
[9] J. McLaughlin, D. A. Reynolds, and T. Gleeson, "A study of computation speed-ups of the GMM-UBM speaker recognition system," in *Proc. Eur. Conf. Speech Commun. Technol. (Eurospeech)*, 1999, pp. 1215–1218.
[10] B. L. Pellom and J. H. L. Hansen, "An efficient scoring algorithm for Gaussian mixture model based speaker identification," *IEEE Signal Process. Lett.*, vol. 5, no. 11, pp. 281–284, Nov. 1998.
[11] B. Sun, W. Liu, and Q. Zhong, "Hierarchical speaker identification using speaker clustering," in *Int. Conf. Natural Lang. Process. Knowledge Eng.*, 2003, pp. 299–304.

[12] W. H. Tsai, S. S. Cheng, and H. M. Wang, "Automatic speaker clustering using a voice characteristic reference space and maximum purity estimation," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 4, pp. 1461–1474, May 2007.

[13] D. Reynolds, "Large population speaker identification using clean and telephone speech," *IEEE Signal Process. Lett.*, vol. 2, no. 3, pp. 46–48, Mar. 1995.

[14] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and effective querying by image content," *J. Intell. Inf. Syst.*, vol. 3, no. 3/4, pp. 231–262, 1994.

[15] H. S. M. Beigi, S. H. Maes, J. S. Sorensen, and U. V. Chaudhari, "A hierarchical approach to large-scale speaker recognition," in *Proc. Eur. Conf. Speech Commun. Technol. (Eurospeech)*, 1999, pp. 2203–2206.

[16] B. Xiang and T. Berger, "Efficient text-independent speaker verification with structural Gaussian mixture models and neural network," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 5, pp. 447–456, Sep. 2003.

[17] T. Kosaka and S. Sagayama, "Tree-structured speaker clustering for fast speaker adaptation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 1994, pp. 245–248.

[18] A. Solomonoff, A. Mielke, M. Schmidt, and H. Gish, "Clustering speakers by their voices," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 1998, pp. 757–760.

[19] P. Angkititrakul and J. Hansen, "Discriminative in-set/out-of-set speaker recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 2, pp. 498–508, Feb. 2007.

[20] P. L. De Leon and V. Apsingekar, "Reducing speaker model search space in speaker identification," in *Proc. IEEE Biometrics Symp.*, 2007.

[21] S. Krstulovic, F. Bimbot, O. Boeffard, D. Charlet, D. Fohr, and O. Mella, "Optimizing the coverage of a speech database through a selection of representative speaker recordings," *Speech Commun.*, vol. 48, no. 10, pp. 1319–1348, Oct. 2006.

[22] J. Goldberger and H. Aronowitz, "A distance measure between GMMs based on the unscented transform and its application to speaker recognition," in *Proc. Interspeech*, 2005, pp. 1985–1988.

[23] M. Ben, R. Blouet, and F. Bimbot, "A Monte-Carlo method for score normalization in automatic speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2002.

[24] D. P. W. Ellis, PLP and RASTA (and MFCC, and Inversion) in Matlab. 2005 [Online]. Available: http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/