Speaker Tracking Using Eigendecomposition and an Index Tree of Reference Models

Mohammad Hossein Moattar and Mohammad Mehdi Homayounpour

This paper focuses on online speaker tracking for telephone conversations and broadcast news. Since the online applicability imposes some limitations on the tracking strategy, such as data insufficiency, a reliable approach should be applied to compensate for this shortage. In this framework, a set of reference speaker models are used as side information to facilitate online tracking. To improve the indexing accuracy, adaptation approaches in eigenvoice decomposition space are proposed in this paper. We believe that the eigenvoice adaptation techniques would help to embed the speaker space in the models and hence enrich the generality of the selected speaker models. Also, an index structure of the reference models is proposed to speed up the search in the model space. The proposed framework is evaluated on 2002 Rich Transcription Broadcast News and Conversational Telephone Speech corpus as well as a synthetic dataset. The indexing errors of the proposed framework on telephone conversations, broadcast news, and synthetic dataset are 8.77%, 9.36%, and 12.4%, respectively. Using the index tree structure approach, the run time of the proposed framework is improved by 22%.

Keywords: Audio document indexing and retrieval, speaker indexing, speaker tracking, eigendecomposition, index structure, generic speaker models.

Manuscript received Nov. 15, 2010; revised Mar. 15, 2011; accepted Apr. 22, 2011.

Mohammad Hossein Moattar (phone: +98 2164542722, email: moattar@aut.ac.ir) and Mohammad Mehdi Homayounpour (email: homayoun@aut.ac.ir) are with the Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran.

http://dx.doi.org/10.4218/etrij.11.0110.0686

I. Introduction

Speaker indexing is the process of labeling different segments of a given speech stream with the labels defining the identity of the speakers. In speaker indexing, there is no primary model of the speakers and the system should operate in an open-set manner. This means that the system should first determine if there exists a trained model of the current speaker in the model set or the utterance corresponds to a new speaker. When there is prior knowledge via sample speech from the speakers, the task becomes a speaker detection task [1].

Speaker indexing consists of two phases. In the first step, the audio document is segmented according to the speaker changes. This segmentation is performed in a manner that each speech segment contains the utterance of a single speaker (speaker segmentation). In the second step, the speech file is traversed, and all the speech segments uttered by the same speaker are labeled with an identical label (speaker tracking). Speaker indexing is also called diarization in some resources [2]. The speaker segmentation phase often provides an initial segmentation for indexing systems, which will be clustered and re-segmented later. Some researchers found no significant performance degradation when using an initial uniform segmentation [2]. Therefore, this paper only concerns the tracking problem due to its importance.

Speaker indexing can be divided into online and offline categories. Most of the previous indexing approaches perform the task in an offline manner. In these techniques, the speech segments are merged in consecutive iterations of a hierarchical clustering [3]-[5]. In online indexing, the speech stream is gradually fed into the system and there is no information on the future data. Hence, traditional clustering methods cannot be applied in this context. More explanation on offline approaches

 © 2011 ETRI

can be found in [5].

In online speaker indexing, we are limited to making decisions with only current and previously seen speech data. Furthermore, since the models of speakers are not available a priori for indexing, we need to create and update them on the fly. Under these circumstances, data is not sufficient to build a speaker model initially. Without good initial models for speaker indexing, we cannot effectively build/update speaker models. To address this problem, generic models are proposed [6]. These models can help initialize the speaker models and also provide a measure for computing the similarity between incoming segments. The generic model set is predetermined by training. The speakers in the training data should be independent of the testing data, so that the generic models can be used for initializing any speaker indexing process.

Generic speaker models are originally introduced by [6] for unsupervised online indexing. In this approach, a set of various speaker models, called sample speaker models, are selected from a pool of general speaker models using the Markov Chain Monte Carlo sampling method [7]. This work has used the NIST Speaker Recognition Benchmark speech corpus and the HUB-4 Broadcast News Evaluation test material as the evaluation datasets. It has achieved 92.5% accuracy on two-speaker telephone conversations, 89.6% on four-speaker conversations, and 87.2% on broadcast news.

In [8], gender background models (which include one model for male speakers and one model for female speakers) are used as the generic models. With each new speaker, his/her gender is identified, and a Gaussian mixture model (GMM) [9] is learned using the speech segment data. This approach uses a fixed global threshold for model selection. However, in [10], different thresholds are used for male and female speakers, and likelihood ratio normalization is applied before making decisions. This has reduced the threshold dependency on the number of speakers. In [11], a speaker tracking system is presented which outputs decisions at one second intervals by scoring the segments on generic speaker models. The length of segments is empirically set to provide good time resolution and small latency for tracking.

In [12], a framework based on multimodal information and Dynamic Bayesian Networks is proposed with the goal of creating an online speaker diarization system. Initial experiments using the framework are encouraging, but the experimental results were very controlled and consisted of a tiny database. The approach presented in [13] fuses video information with audio captured using microphone array to perform online speaker tracking.

In [14], a hybrid low-latency speaker diarization system is proposed which consists of offline diarization and online diarization subsystems. In this system, the offline subsystem is used to produce the corresponding labels for the first T_1 time intervals. The core of the offline subsystem is the ICSI speaker diarization system [15]. The output of this subsystem is used to improve the output of the online indexing. The online approach uses a universal background model (UBM) to initial the model of each new speaker using maximum a posteriori (MAP) adaptation. The best overall online diarization error rate of this framework on Dev09 NIST Rich Transcription evaluation set is 37.75%.

The proposed framework was inspired by the idea of generic speaker models originally proposed in [6]. One of the main concerns of generic model-based indexing approaches is the time complexity of model selection with the increment of the detected speakers. In this paper, an index structure of the generic models is proposed to decrease the search space when seeking the most similar model to the current segment. Another concern is training and selection of the generic models, so that the resulted models are sufficiently general and can cover the speaker's space thoroughly. To improve the generic model capacity, adaptation approaches in eigenspace are proposed. The best number of required generic speaker models is also another concern in this group of approaches.

To continue this paper, we have to clarify some definitions. Generic speaker models are the set of general speaker-dependent models. On the other hand, reference speaker models are the models selected from a pool of generic speaker model and are used in the tracking procedures.

The rest of this paper is organized as follows. Section II briefly describes the proposed framework. An explanation on eigenvoice adaptation is offered in section III. The approach for reference model selection is discussed in section IV. Section V explains the construction of the index tree of reference models. The tracking procedure is defined in section VI. Sections VII and VIII describe the evaluation datasets and the performance measure, respectively. A detailed discussion on the experiments and achievements is presented in section IX. Finally, section X includes the conclusions and the direction for future works.

II. Proposed Framework

The proposed tracking framework is illustrated in Fig 1. The differences of the proposed framework compared to the other approaches are in the use of index tree of reference speakers and eigenvoice adaptation for generic model formation.

Before the tracking starts, generic speaker models are trained (building generic models) (Fig. 1(a)). To build the generic models, we adapt a UBM [16] using an adaptation algorithm. The adaptation methods included are MAP adaptation [16], maximum likelihood eigendecomposition (MLED) adaptation, and MAP eigendecomposition (MAPED) adaptation

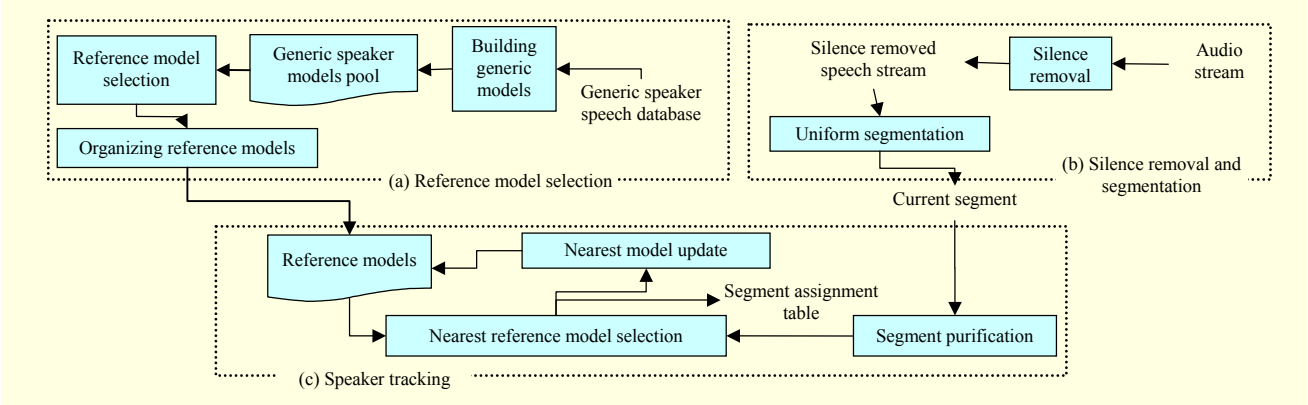

Fig. 1. Proposed speaker tracking framework: (a) generic speaker models are trained on a set of arbitrary speech dataset, (b) audio stream is silence removed and uniformly divided into speech segments, (c) input to tracking algorithm is a sequence of uniformly segmented utterances. The output of the tracking step is a segment assignment table.

approaches [17], [18]. The resulting generic models are then filtered, and a few of these models are selected as reference models (reference model selection). To decrease the number of similarity computation in the tracking stage, we propose the use of an index tree on the selected reference models (organize reference models).

The main indexing procedure starts with silence removal. The resulting stream is uniformly divided into segments (Fig. 1(b)). The resulted segments are fed into the tracking procedure (Fig. 1(c)). The tracking stage is summarized in algorithm 1.

Algorithm 1. Speaker tracking.

1. The speech segment is purified to exclude the frames which are likely to be silence, noise, or the less informative frames (segment purification) [19].
2. The likelihood of the current segment to each reference model is computed, and the most similar model is selected (nearest reference model selection).
3. The most similar reference model (which may be an already identified speaker) is adapted with the current segment and replaced in the reference model set. If the input segment is identified as belonging to a new speaker (the most similar model is an original reference model), the model is adapted with the current segment and the resulting model is added to the reference set (nearest model update).
4. Corresponding label is assigned to the segment.

III. Eigenvoice Adaptation

The generic speaker models are built using eigenvoice adaptation approaches as introduced in [18]. The advantage of adaptation methods in eigenspace is their better robustness to data insufficiency [18].

The eigenvoice adaptation approach has two steps which are eigenvoice construction and coefficient estimation. The second phase is usually known as the adaptation phase. In the first phase, a set of generic speaker models from G speakers is trained. For each model, the mean vectors of the Gaussian components are concatenated to form a supervector. Suppose that the dimension of the feature space is d, and the number of component in the GMM is denoted by M. Therefore, the dimension of the resulting supervector, υ, is $D=d\times M$. Then, the $D\times D$ covariance matrix is calculated from G supervectors. Then, eigendecomposition, that is, principle component analysis (PCA) [20], is applied on this covariance matrix. The K most dominant eigenvectors are selected to form a K-dimensional eigenspace.

In the adaptation phase, we would like to adapt a speaker independent GMM to a specific speaker using the provided adaptation data X. For this purpose, we have to estimate the location of the new speaker in the K-dimensional eigenspace, that is, find the set of K weight coefficients $\{\omega(j), j=1,\ldots, K\}$ corresponding to K eigenvectors $\{e(j), j=1,\ldots, K\}$. The supervector of the adapted speaker model is constructed as

$$\upsilon = \omega(1)e(1)+\ldots+\omega(K)e(K) = \sum_{i=1}^{K}\omega(i)e(i). \quad (1)$$

The goal of speaker adaptation methods in eigenvoice space is to find the corresponding weights so that the resulting speaker model is fitted to the adaptation data. The maximum likelihood method can be used to estimate the above weight coefficients by solving the following equation:

$$\hat{\omega} = \arg\max_{\omega} P(X|\omega). \quad (2)$$

We can use the expectation maximization [21] algorithm to solve (2) (MLED). The MLED method is prone to biased estimates of combination weights when the amount of the adaptation data is small. To solve this problem, MAP criterion is proposed for estimating ω. Therefore, this method is known

as MAP eigendecomposition (MAPED). It is known that MAPED achieves better performance than MLED with little adaptation data [18].

In general, the eigenvoice space characterizes the principal directions of speaker variations, and the adapted models are found through linear combination of eigenvoices. In the proposed framework, we apply eigenvoice adaptation approaches for adapting primary generic speaker models. We suppose that using these approaches, the spatial locations of the generic speakers will also be embedded in the corresponding adapted models. This will help to select the most spatially different speakers and cover the speaker space more efficiently.

IV. Reference Model Selection

Not all the models in the generic model set are appropriate for use in the proposed framework. Therefore, a model selection strategy should be applied to select the most dissimilar and spatially distributed speaker models. It should be noted that the proposed framework is based on the Gaussian mixture modeling of the speakers. The proposed reference model selection approach starts by computing the following Kullback-Leibler (KL_m) distance between every two speaker models:

$$KL_m(Model_1, Model_2) = \sum_{i=1}^{M_1} w_{i1} \min_{j=1}^{M_2} KL(Norm_{i,1}, Norm_{j,2}), \quad (3)$$

where $Norm_{ik}$ and w_{ik} are the i-th Gaussian component and its corresponding weight in $Model_k$, respectively, and M_k denotes the number of components in $Model_k$. The above distance metric is not symmetric, and $KL_m(Model_1, Model_2)$ is different from $KL_m(Model_2, Model_1)$. Hence, the minimum of the two values above is applied as the final distance between the two models.

In the above equation, $KL(.,.)$ is the standard KL divergence between two Gaussian distributions and is computed as

$$KL(Norm_i(\mu_i, \Sigma_i), Norm_j(\mu_j, \Sigma_j)) = \frac{1}{2}\left(\log\frac{|\Sigma_j|}{|\Sigma_i|} + Tr(\Sigma_j^{-1}\Sigma_i) + (\mu_i - \mu_j)^T \Sigma_j (\mu_i - \mu_j) - d \right), \quad (4)$$

where d is the dimension of the Gaussian distributions, and μ_i and Σ_i are the mean vector and the covariance matrix of the i-th distribution, respectively. After computing the distance between every two models, the pair with the maximum pairwise distance is selected. Then, the selected models are added to the set of reference models. In the next iteration, the model with the maximum distance from the reference models is selected and added to the reference set. The distance of a model from the reference set is denoted by the minimum distance between that model and each member of the reference set. This algorithm is repeated until the required number of reference models is achieved. After selecting a number of reference models, these models are organized in a tree structure.

V. Organizing Reference Models

In the proposed framework, we address an approach for organizing the set of N reference speaker models in the form of a tree to obtain lower computational cost (that is, $< O(N)$) and speed up the tracking process. Organizing an index of the reference speaker models relates to the classical issue of indexing multidimensional data. Previous research has put forward a considerable amount of contribution based on a variety of structures [22], [23]. The difference is in the nature of the entities to be indexed, namely, probability distributions.

In the proposed approach, we aim to form a hierarchy of reference speakers by grouping N models bottom-up. In this paper, we consider a binary tree as the index structure. In this structure, the two GMM speaker models, $Model_1$ and $Model_2$, are represented by a single parent, $Model_{\text{Merged}}$, which is also presented by a GMM. Let us denote the current segment in the tracking process by U_c. The cost reduction at tracking time, when the tree is explored, is obtained by computing a single value $P(U_c|Model_{\text{Merged}})$ instead of both $P(U_c|Model_1)$ and $P(U_c|Model_2)$. Consequently, $Model_{\text{Merged}}$ should be designed so that $P(U_c|Model_{\text{Merged}})$ is as close to both $P(U_c|Model_1)$ and $P(U_c|Model_2)$ as possible. The cost of representing $Model_1$ and $Model_2$ by $Model_{\text{Merged}}$, can be expressed as

$$Cost \approx \sum_{k=1,2} E\left[\ln p(U_c|Model_k)\right] - E\left[\ln p(U_c|Model_{\text{Merged}})\right]. \quad (5)$$

Also, let us define $Model_{\text{Splitted}}$ as the sum of the child models, in which the Gaussian components of both $Model_1$ and $Model_2$ are gathered under the same mixture. Hence, supposing that the number of components in $Model_1$ and $Model_2$ are M_1 and M_2, respectively, the number of components in $Model_{\text{Splitted}}$ is M_1+M_2. To make sure that the sum the weights of the components is equal to 1, the weight coefficients of the components of model $Model_{\text{Splitted}}$ is divided by 2. Assuming that both child models are equally probable, the cost of representing $Model_1$ and $Model_2$ by $Model_{\text{Merged}}$ can be expressed by

$$Cost \approx E\left[\ln p(U_C|Model_{\text{Splitted}})\right] - E\left[\ln p(U_C|Model_{\text{Merged}})\right]. \quad (6)$$

Equation (6) can be minimized if the following equation is minimized:

$$Cost \approx \int_{\rho} Model_{\text{Merged}} \ln\left(Model_{\text{Splitted}} / Model_{\text{Merged}}\right), \quad (7)$$

where ρ is the search space of all possible Gaussian components. Equation (7) is actually the definition of KL divergence [24] between the two models. Therefore, the optimal model which minimizes the cost stated in (6) is

$$\hat{Model}_{\text{Merged}} = \arg\min_{\rho} KL_m(Model_{\text{Splitted}}, Model_{\text{Merged}}). \quad (8)$$

Equation (8) means that the optimal parent model to represent its child models is the one that includes the components of $Model_{\text{Splitted}}$. $Model_{\text{Splitted}}$ has too many components to be computationally efficient. Therefore, we should find the optimal mapping that maps M_1+M_2 components of $Model_{\text{Splitted}}$ to M_{Merged} components of $Model_{\text{Merged}}$ ($M_{\text{Merged}}<M_1+M_2$) so that the KL divergence expressed in (8) is minimized.

To achieve this goal, an iterative algorithm is proposed which is similar to the classical *k*-means algorithm with a slight difference. In the proposed clustering algorithm, which is summarized in algorithm 2, the Gaussian components, rather than simple observation vectors, are assigned to a number of clusters. In this scheme, the elements to be clustered are the components of $Model_{\text{Splitted}}$, and the final cluster representatives are the components of $Model_{\text{Merged}}$. Apparently, the clustering objective is to minimize the KL_m divergence of (8).

Algorithm 2. Merging algorithm for estimating $Model_{\text{Merged}}$ from $Model_1$ and $Model_2$.

- Assign M_1+M_2 Gaussian components to M_{Merged} clusters randomly.
- Repeat steps 1 and 2 until convergence.

1. The parameters of the new mixtures are updated using the following equations:

$$\hat{w}_{j,\text{Merged}} = \sum_{i\in Set(j)} w_{i,\text{Splitted}} \quad (9)$$

$$\hat{\mu}_{j,\text{Merged}} = \sum_{i\in Set(j)} w_{i,\text{Splitted}}\mu_{i,\text{Splitted}} \Big/ \hat{w}_{j,\text{Merged}} \quad (10)$$

$$\delta\mu = (\mu_{i,\text{Splitted}} - \hat{\mu}_{j,\text{Merged}})$$

$$\hat{\Sigma}_{j,\text{Merged}} = \sum_{i\in Set(j)} w_{i,\text{Splitted}}(\Sigma_{i,\text{Splitted}} + \delta\mu.\delta\mu^T) \Big/ \hat{w}_{j,\text{Merged}} \quad (11)$$

where w_i, μ_i, and Σ_i are the weight, mean vector, and the variance vector of the *i*-th Gaussian component, respectively. Also $\hat{|}$ denotes the new estimation of the '*Merged*' model parameters and *Set*(*j*) is the set of components which are assigned to the *j*-th cluster in the current iteration.

2. Reassign the '*Splitted*' components to the '*Merged*' components so that the KL divergence of this assignment is minimized:

$$Set(j)_{\text{New}} = \arg\min_{i} KL(Norm_{i,\text{Splitted}}, Norm_{j,\text{Merged}}). \quad (12)$$

- Update the parameters of the resulting Gaussian components.

Having algorithm 2 for reorganizing the components of the child nodes in the single mixture of components as the parent model, the algorithm of organizing the reference models in an index tree structure is as algorithm 3.

Algorithm 3. Organizing the reference models in an index tree.
Perform the following steps until only two models remain:

- Compute the KL_m (3) distance between every two models.
- Select the two models with the least pair wise distance.
- Use algorithm 2 to merge the selected models.
- The resulting model is the parent of the previous models.

If the number of Gaussian components in each of the child models is *M*, for better presentation of the two speakers, the number of components in $Model_{\text{Merged}}$ would be greater than *M*. The number of Gaussian components in $Model_{\text{Merged}}$ should also be clearly smaller than 2*M* to ensure the computational cost reduction. For simplicity and further cost reduction, we suppose the number of components of the intermediate node models, $Model_{\text{Merged}}$, to be equal to the number of components of the original reference models, that is, *M*.

Since new speakers are incrementally added to the original set of reference models, the model indexing approach should be amenable to incremental processing, that is, it can accommodate new speaker models at the leaves. Since the process of model insertion is performed in the tracking time and one of the main requirements of tracking is low computational cost, the update process is only performed locally on the immediate parent. Figure 2 explains the process of inserting a new adapted model into the tree. In this figure, the reference models are depicted with *R* and the intermediate models in the tree are depicted with *I*. The new speaker model and the speech segment are depicted with *S* and *U*, respectively.

Suppose that the first segment, U_1, is found to be closest to the reference model, R_1. The new speaker model, S_1, is formed by adapting R_1 using the data available in the current segment. Model S_1 should be inserted into the index structure. Since both R_1 and S_1 are to be used in the tracking process, these models should be placed in the leaf nodes. Hence, the tree structure is

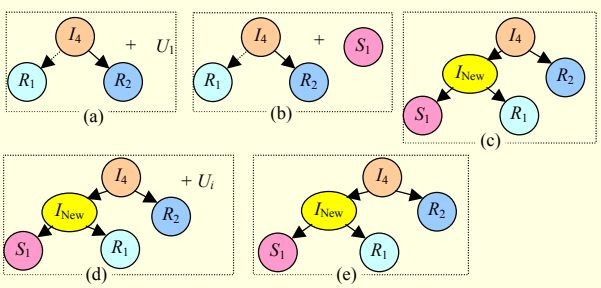

Fig. 2. Process of updating tree structure with entrance of new utterance.

changed to include a new node, I_{New}, which is assumed to be the parent of R_1 and S_1 as shown in Figs. 2(a) to 2(c). The parameters of I_{New} are computed using the merging scheme described in algorithm 2. Now suppose that with the continuation of tracking, a new segment, U_i, is also found which belongs to S_1. As illustrated in Figs. 2(d) to 2(e), S_1 is updated with the new utterance and no further update or reformation is supposed. Obviously, if we update the parameters of I_{New}, the index tree would be more accurate, but this updating will be an obstacle for the low computational cost.

In this paper, another solution for constructing the intermediate nodes of the tree undergoes experimentation. This solution concerns the adaptation of the intermediate nodes using the data in their child nodes. In this solution, the merging algorithm (algorithm 2) is substituted with the adaptation of UBM using the speech data of child nodes.

VI. Speaker Tracking

The proposed tracking approach starts by silence removal and uniform speech segmentation. The applied method for silence removal is described in [25]. The main task is to classify the input utterances to their corresponding speakers. This process is briefly described in section II. The following is a more detailed discussion on the tracking steps.

Step 1. Purification. The speech segment is purified to exclude the frames likely to be silence, noise, or the frames which are less informative [19]. In this approach, a GMM is constructed for the current segment. Then, the likelihood of each frame to the GMM is computed, and a number of speech frames with the highest likelihood are removed from the segment. The idea behind this approach is that the silence or noisy frames of a speech segment have the least variance and are clustered under the same component of the GMM. Hence, when the likelihood is computed, the silence or noisy frames fall into the least variance components and receive the highest likelihood score.

Step 2. Assignment. The index tree of reference models is traversed to reach the nearest reference model to the current segment. To measure the similarity between the current segment and reference models, the likelihood ratio test is applied. When reaching one of the leaf nodes, that is, a reference model in the tree, if a previously identified speaker model exists in that leaf, its model is updated with the current segment and the segment is labeled with a label corresponding to that speaker. If the reached model is not an identified speaker, the tree structure is updated according to Figs. 2(a) to 2(c) and a new speaker with an identical label is added to the set of currently found speakers.

The exhaustive search strategy which scans the whole set of reference models linearly is also experimented upon and compared with the index tree traversing approach in this paper. Let us discuss the complexity cost using either the exhaustive (linear) search or the tree search. Suppose that we have N reference models in the set, and up to the i-th time slice, L speakers are detected in the speech signal. For the tree search, there may be three different scenarios. The ideal scenario is that the constructed tree is balanced. In this case, the number of comparisons to find the most similar model to the input utterance is $\log_2(N)$ for U_1 and $\log_2(N+L)$ for U_i. This case is highly improbable. The worst condition is when the tree is biased in one direction. In this case, the tree search is actually a linear search and the average number of comparisons is equal to the current number of models. Therefore, the average number of comparisons for U_1 and U_i is N and $N+L$, respectively. This scenario is also improbable. The intermediate scenario is that the index tree is neither fully balanced nor biased. We can assume that the complexity of search for this case is the average of the two previous ones.

Table 1. Average number of comparisons for exhaustive (linear) search and tree-based search.

N	L	Linear search	Balanced tree	Biased tree	Average
8	0	8	3 (ef = 2.66)	8 (ef = 1)	5.5 (ef = 1.45)
	2	10	3.32 (ef = 3.01)	10 (ef = 1)	6.66 (ef = 1.5)
16	0	16	4 (ef = 4)	16 (ef = 1)	10 (ef = 1.6)
	8	24	4.58 (ef = 5.24)	24 (ef = 1)	14.29 (ef =1.67)

Table 1 illustrates the discussion above in a numerical form including the linear search condition. In this table, two problems are exemplified. In the first problem, the number of reference models is equal to 8 and the number of detected speakers up to the i-th time slice is 2. In the second case, the number of reference models and detected speakers are 16 and 8, respectively. In Table 1, ef is the effectiveness quotient of the tree-based search compared to the linear search. As Table 1 denotes and the previous explanations suggest, the effectiveness of the index tree structure magnifies with the increase in the number of speakers in the speech signal. However, at the worst case, the effectiveness of the tree-based search will be equal to the exhaustive search.

VII. Evaluation Datasets

The evaluation database consists of the 2002 Rich Transcription Broadcast News (BN) and Conversational Telephone Speech (CTS) [26], which are a part of the NIST evaluation databases for rich transcription. The corpus consists

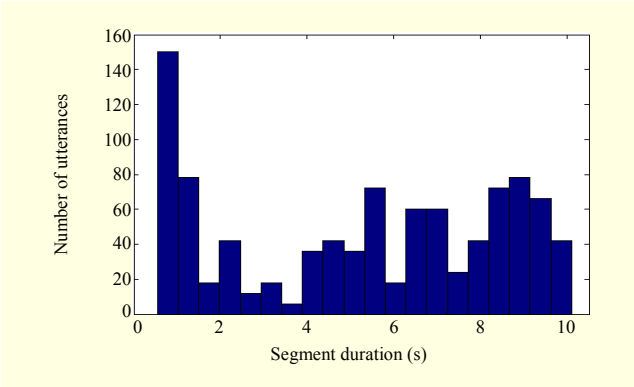

Fig. 3. Histogram of durations of speaker segments of synthetic dataset.

of the speech files for evaluating rich transcription systems in the domains of broadcast news and conversational telephone speech. The CTS data is composed of 60 approximately five-minute excerpts from 60 different conversations: 20 excerpts from Switchboard-1 data, 20 excerpts from Switchboard-2 data, and 20 excerpts from Switchboard Cellular-2 data. The BN data is composed of 6 approximately 10-minute excerpts from 6 different broadcasts. The broadcasts were selected from programs from MNB, PRI, NBC, CNN, VOA, and ABC, all collected in 1998. The minimum and maximum numbers of speakers in the BN data are 5 and 19, respectively. This dataset includes overlap regions which are excluded from the signals.

We have also generated a synthetic conversational dataset using the OGI multilingual database [27] which contains speech utterances from 11 different languages. To generate this dataset, we used about 800 utterances of 80 speakers from 4 languages, including English, Farsi, German, and French, and concatenated the utterances of different speakers in a random order. The synthetic utterances are generated with different numbers of speakers, 4, 8, and 40. Figure 3 shows the histogram of the durations of utterances in the synthetic conversational dataset.

Also, in the proposed framework, it is necessary to train generic models. For this purpose, Farsdat telephony and microphone speech datasets are applied [28]. Both of these datasets are in Farsi and are usually used for speech and speaker recognition over a telephone and microphone, respectively. Using these two databases, the independency of the generic models and the evaluation databases is ensured.

VIII. Evaluation Metric

The indexing evaluation metric is inspired from the RT-09 NIST Rich Transcription evaluations [29]. To evaluate the accuracy of the indexing, an optimum mapping from the real speakers in the conversation to the output speakers should be found. The criterion for this mapping is the percentage of the speech parts which are jointly assigned to the real speaker and the output speaker. In other words, a real speaker and an output speaker with the most speech parts in common are supposed to be the same. In this mapping, each real speaker should be mapped to at most one output speaker and each output speaker should also be mapped to only one real speaker.

Since there may exist segmentation errors in reference files, 250 ms of speech signal from each side of the change point is excluded from the evaluations. The total speaker indexing error is computed as the percentage of speech parts which are not assigned to the true speaker:

$$\text{Indexing error}=\frac{\sum_{\text{allsegs}} \{dur(s)(\max(N_{\text{R}}(s),N_{\text{O}}(s))-N_{\text{C}}(s))\}}{\sum_{\text{allsegs}} \{dur(s).N_{\text{R}}(s)\}}. \quad (13)$$

In the above equation, for each speech segment s, the following values should be computed:

$dur(s)$: Speech segment duration,

$N_{\text{R}}(s)$: The number of real speakers in the speech segment,

$N_{\text{O}}(s)$: The number of output speakers in the speech segment,

$N_{\text{C}}(s)$: The number of real speakers which have participated in the speech segment and the corresponding speaker is recognized in the segment.

IX. Experiments

The first operation in the proposed framework is the construction of the generic models. Since our evaluations are performed in both telephone conversation and broadcast news domains, to achieve the most conformity with the environment, it is better to use the corresponding datasets to make the generic speaker models. For our purpose, 50 different speakers from the Farsdat telephone speech dataset and 200 speakers from the Farsdat microphony dataset are selected. The speakers are randomly selected, but the number of male and female speakers is equal. After silence removal from the speech utterances [25], a UBM consisting of 64 components is trained using the speech data from all speakers. To construct the model of each speaker, the UBM is adapted with the speech data from that speaker. For adapting each speaker model, 30 s of speech data of the speaker is used.

As explained before, in the proposed framework, we relinquish actual speaker segmentation and instead perform uniform segmentation. Long speaker segments can capture the speaker-related information better, but to detect shorter speech episodes, we should use as short an analysis segment as possible. Therefore, we used one second uniform speech

segmentation. This duration seems to be rational because speech utterances shorter that one second are rare. The same duration settings are applied in [11] with good achievements.

In literature, there is a preference towards mel-frequency cepstral coefficient (MFCC) [30] vectors as speech features for speaker indexing [5]. Therefore, the features used in our experiments are 12-dimensional MFCCs with 24-channel mel-filter bank. To extract these features, a 30 ms Hamming window with 10 ms shift is used.

Two different eigenspaces are built for telephony and microphone speech. To build these eigenspaces, the corresponding set of generic speaker models is used. As mentioned previously, each supervector is made by concatenating the mean vectors of the corresponding GMM. Since GMMs have 64 mixtures and the dimension of the feature vectors is 12, the dimension of supervectors, D, is 768 (64×12). After PCA eigendecomposition, the first 33 eigenvoices are used as the base for the eigenspace.

The experimental results are discussed separately. First, the experiments are concerned with the optimum number of reference models. Then, the effect of generic model adaptation approaches on the indexing error rate is evaluated. Finally, the index tree structure for indexing is evaluated. Except for the first and final experiments, which are evaluated on the broadcast news and the synthetic dataset too, the other approaches are only evaluated on the CTS part of NIST 2002 Rich Transcription evaluation database, and the results are supposed to be extensible.

1. Number of Reference Models

It is not rational to set the optimal number of reference models using the test data because it is supposed that there is no information about the actual nature of the input conversation. However, it will be useful to monitor the indexing accuracy on different test sets for various number of reference models.

Figure 4 illustrates the indexing error of the proposed approach on three different datasets versus the number of reference speaker models. In Fig. 4, Synthetic k means that the number of speakers in the synthetic conversations is k. We considered different number of reference models, 4, 8, 16, 32, and 64. Note that in this experiment and the next one which concern the reference model selection and generic model construction, the exhaustive search approach is applied for the nearest model selection.

Figure 4 shows that the average indexing error of the proposed approach increases by the increment in the number of reference models. As the number of reference models increases, more similar models are chosen. In this situation, one test speaker would be recognized as two or more reference

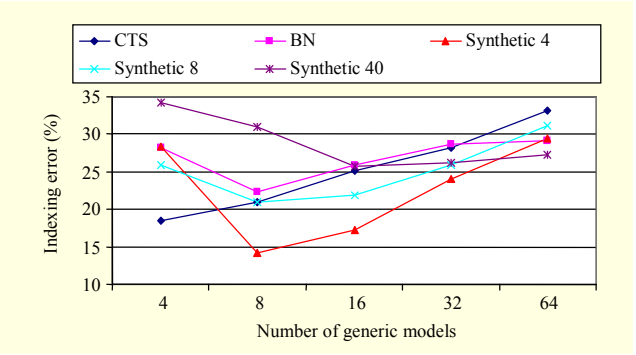

Fig. 4. Indexing error of proposed framework on different datasets (CTS, BN, and synthetic) for various number of reference models.

speakers. Only when the number of speakers in the conversation is high, as in the 40 speaker conversations, increasing the number of reference speaker models may improve the results.

Figure 4 shows that 4 reference speaker models are the best choice for telephone conversations. For broadcast news data, the optimum number of reference models is between 8 and 16. This is because the maximum number of speakers in this dataset is 19 while the minimum is 5. On the synthetic dataset, the best choice for conversations with 4 or 8 speakers is 8 reference models, whereas the optimum number for conversations with 40 speakers is higher.

We have to consider that using a fixed-sized set of reference speakers for various applications is a double-edged sword. Having few reference speakers may lead to misclassification. On the other hand, a very large reference set may cause the utterances of a single speaker to be divided into two or more classes. Hence, having a moderate number of speakers is best. From the above experiments, we can conclude that 8 to 16 reference speakers is optimum. In the following experiments, 8 reference models are used, although 16 reference speakers could also be evaluated.

2. Training Generic Models

In these experiments, we evaluated the influence of the generic speaker model adaptation approaches on the indexing error. For this purpose, the effect of eigenvoice adaptation methods is compared to MAP adaptation methods. The results of these experiments are summarized in Table 2.

Table 2 shows that the MAPED and MLED approaches outperform the MAP adaptation. As mentioned before, eigenvoice space represents the directions of speaker variations. Hence, when speaker models are constructed using the linear combination of eigenvoices (that is, adaptation), they rest in the orientation of the speaker variations. Therefore, the adapted

speaker models capture the structure of the speaker space and cover the speaker space better. Clearly, when the speaker space is covered better and the generic models are more diverse, the tracking precision will improve.

3. Index Structure

In the final experiments, we evaluated the effect of the index tree of the reference models on the accuracy of the indexing. The results of applying this structure in the indexing procedure are summarized in Table 3. These results are extracted on the CTS test set.

These experiments show that the first tree construction approach which uses algorithm 2 increases the indexing error rate compared to the exhaustive search (indicated in Table 2). It is due to the fact that the model resulting from algorithm 2 through the merging scheme is weak and cannot represent its child well enough. On the other hand, the adaptation approaches for constructing the intermediate nodes of the tree decreases the indexing error. This improvement is the highest when both reference models and the nodes of the tree are trained using the MAPED adaptation. The reason behind this better outcome is that the adaptation approach constructs a better parent model; hence, the cost function for representing two models with their parent model is minimized. Since the index tree formation is an offline procedure, the runtime of the tracking operation will be indifferent using either of these methods for constructing the index tree.

Table 2. Indexing error rate on CTS data using different adaptation methods to build generic models.

Search method	Reference model adaptation approach	Indexing error (%)
Exhaustive search	MAP	18.54
	MLED	13.58
	MAPED	10.47

Table 3. Indexing error of tree search strategies on CTS data. Experiments are performed for different reference model adaptation approaches and intermediate node generation methods.

Search method	Reference model adaptation	Intermediate node update	Indexing error (%)
Tree search	MAP	Algorithm 2	18.55
	MLED		11.77
	MAPED	Algorithm 2	11.34
		MAP	12.41
		MLED	9.48
		MAPED	8.77

4. Final Evaluation

The indexing error rate for all three evaluation datasets using the proposed framework is illustrated in Table 4. The indexing error on the synthetic dataset is averaged on all three subsets. As illustrated in Table 4, the indexing error is almost the same for different application domains which shows the robustness of the indexing procedure. Only the error rate on the synthetic dataset is relatively higher which is due to the diversity of speakers and languages in these conversations.

Table 5 tries to illustrate the effectiveness of the proposed index tree structure from the process time point of view. These process times are achieved on an AMD Athlon dual-core processor with 2.7 GHz clock rate and 3 GB of RAM for each second of the input speech. These results only include the segment assignment and tree update operations. The process time of fixed tasks, such as silence removal, feature extraction, and frame purification, are excluded from these experiments. The overhead cost of time and space for building the tree is not

Table 4. Indexing error rate on all three datasets using proposed framework for both exhaustive search and index tree search.

Evaluation database	Indexing error (%)	
	Exhaustive search (without index tree)	Index tree search (proposed approach)
CTS	10.47	8.77
BN	11.41	9.36
Synthetic dataset	14.58	12.4

Table 5. Average process time of assigning each second of speech to its corresponding speaker (ms).

	CTS	Syn. 4	Syn. 8	BN	Syn. 40	Average
Linear search	380	409.7	421.6	455.1	523.1	437.9
Tree structure	330	330.6	335.5	345.2	351.3	338.5
Improved (%)	13.15	19.30	20.42	24.14	32.84	22.69

Table 6. Average real-time factor (xRT) of different modules of the proposed framework.

	Silence removal	Feature extraction	Uniform segmentation
xRT	0.014	0.020	≈0

included in Table 5 and is not necessary to be included because this operation is performed offline.

These results show that the proposed tree structure does not significantly improve the process time of the tracking task when the number of speakers in the speech data is low (that is, telephone conversation or 4-speaker conversations). The main improvement arises when the number of speakers in the speech documents increases. This enhancement is clearly seen for broadcast news and 40-speaker synthetic conversations. Table 5 shows that even for the telephone conversations with two speakers, the process time have improved by 13%.

To show the applicability of the proposed approach in online applications, we have to prove that the run time of the proposed framework satisfies real-time requirements and its real-time factor (xRT) is lower than 1. Table 6 shows the average real-time factor of each module of the proposed framework.

Since the average xRT of the proposed speaker tracking approach, using index tree structure, on different datasets is 0.338 (Table 5), the average xRT of the whole framework will be 0.372. This shows that the processing time of the proposed framework meets the real-time requirements.

X. Conclusion

In this paper, a speaker indexing approach was proposed. The main obstacle in the tracking task is that the segments to be classified are gradually fed into the system and we do not have much information about the speakers to come. The generic models are used as a reference for model comparison and selecting the most similar speaker model to the input segment. To decrease the time complexity of the model search procedure, an index tree on the reference models is proposed. In addition to these approaches, frame purification and eigenspace speaker model adaptation are also proposed in this paper.

The proposed framework is evaluated on the NIST 2002 Rich Transcription evaluation database as well as a synthetic dataset. The indexing errors of the proposed framework on telephone conversations, broadcast news, and synthetic dataset were 8.77%, 9.36%, and 12.4%, respectively. The main improvement of the proposed approach was in its response time. Using the proposed index structure idea, the speaker tracking time is improved between 13% for 2-sided telephone conversations to 32% for 40-speaker synthetic conversations.

Although the achieved indexing results are relatively good, there are still some directions for future work and improvement. For example, a back-track approach may help decrease the wrong classification of the speech segments. Approaches that apply a decision tree to record the history of classifications can be proposed in this direction [31]. Also, likelihood normalization, as proposed in [9], [10], may improve the indexing results. We also propose experimenting with other eigenadaptation approaches such as eigenspace-based maximum likelihood linear regression (Eigen-MLLR) adaptation [32] or kernel PCA-based eigenadaptation [33].

References

[1] A. Martin and M. Przybocki, "Speaker Recognition in a Multi-speaker Environment," *Proc. Eur. Conf. Speech Commun. Technol.*, vol. 2, 2001, pp. 787-790.

[2] S. Meignier et al., "Step-by-Step and Integrated Approaches in Broadcast News Speaker Diarization," *Comput. Speech Language*, vol. 20, no. 2-3, 2006, pp. 303-330.

[3] T.H. Nguyen, H. Li, and E.S. Chng, "Cluster Criterion Functions in Spectral Subspace and Their Application in Speaker Clustering," *Proc. ICASSP*, 2009, pp. 4085-4088.

[4] K. Iso, "Speaker Clustering Using Vector Quantization and Spectral Clustering," *Proc. ICASSP*, 2010, pp. 4986-4989.

[5] M. Kotti, V. Moschou, and C. Kotropoulos, "Speaker Segmentation and Clustering," *Signal Process.*, vol. 88, no. 5, 2008, pp. 1091-1124.

[6] S. Kwon and S. Narayanan, "Unsupervised Speaker Indexing Using Generic Models," *IEEE Trans. Speech Audio Process.*, vol. 13, 2004, pp.1004-1013.

[7] M. Davy et al., "Supervised Classification Using MCMC Methods," *Proc. ICASSP*, 2000, pp. 33-36.

[8] K. Markov and S. Nakamura, "Never-Ending Learning with Dynamic Hidden Markov Network," *Proc. Interspeech*, 2007, pp. 1437-1440.

[9] D.A. Reynolds, "Speaker Identification and Verification Using Gaussian Mixture Speaker Models," *Speech Commun.*, vol. 17, no. 1-2, 1995, pp. 91-108.

[10] K. Markov and S. Nakamura, "Improved Novelty Detection for Online GMM Based Speaker Diarization," *Proc. Interspeech*, 2008, pp. 363-366.

[11] M. Zamalloa et al., "Low Latency Online Speaker Tracking on the AMI Corpus of Meeting Conversations," *Proc. ICASSP*, 2010, pp. 4962-4965.

[12] A.K. Noulas and B.J.A. Krose, "Online Multimodal Speaker Diarization," *Int. Conf. Multi-modal Inferences*, 2007, pp. 350-357.

[13] J. Schmalenstroeer et al., "Fusing Audio and Video Information for Online Speaker Diarization," *Proc. ASRU*, 2007, pp. 1163-1166.

[14] C. Vaquero, O. Vinyals, and G. Friedland, "A Hybrid Approach to Online Speaker Diarization," *Proc. Interspeech*, 2010, pp. 2638-2631.

[15] C. Wooters and M. Huijbregts, "The ICSI RT07s Speaker Diarization System," *Proc. RT Meeting Recognition Evaluation Workshop*, 2007.

[16] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn, "Speaker Verification Using Adapted Gaussian Mixture Models," *Digit. Signal Process.*, vol. 10, no. 1-3, 2000, pp. 19-41.

[17] R. Kuhn et al., "Rapid Speaker Adaptation in Eigenvoice Space," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 4, 2000, pp. 695-707.

[18] C.H. Huang, J.T. Chien, and H.M. Wang, "A New Eigenvoice Approach to Speaker Adaptation," *Proc. Int. Symp. Chinese Spoken Language Process.*, 2004, pp. 109-112.

[19] X. Anguera et al., "Frame Purification for Cluster Comparison in Speaker Diarization," *Proc. 2nd Int. Workshop Multimodal User Authentication*, 2006, pp. 135-139.

[20] I.T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, 1986.

[21] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc.*, series B, vol. 39, no. 1, 1977, pp. 1-38.

[22] S. Berrani, L. Amsaleg, and P. Gros, "Robust Content-Based Image Searches for Copyright Protection," *Proc. ACM Workshop Multimedia Databases*, 2003, pp. 70-77.

[23] P. Zezula et al., "Similarity Search: The Metric Space Approach," *Adv. Database Syst.*, vol. 32, 2006, pp. 23-38.

[24] S. Kullback and R.A. Leibler, "On Information and Sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, 1951, pp. 79-86.

[25] M.H. Moattar and M.M. Homayounpour, "A Weighted Feature Voting Approach for Robust and Real-Time Voice Activity Detection," *ETRI J.*, vol. 33, no. 1, 2011, pp. 99-109.

[26] J. Garofolo et al., "NIST Rich Transcription 2002 Evaluation: A Preview," *Proc. Language Resources Evaluation Conf.*, May 2002.

[27] Y.K. Muthusamy et al. "The OGI Multi-language Telephone Speech Corpus," *Proc. ICSLP*, vol. 2, 1992, pp. 895-898.

[28] M. Bijankhan, *Great Farsdat Database*, Technical report, Iran Research Center on Intelligent Signal Processing, 2002.

[29] The 2009 (RT-09) Rich Transcription Evaluation Plan, http://www.itl.nist.gov/iad/mig//tests/rt/2009/docs/rt09-meeting-eval-plan-v2.pdf, last accessed on Dec. 6, 2010.

[30] L.R. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*, Englewood Cliffs, NJ: Prentice-Hall, 1993.

[31] W. Wang et al., "A Decision-Tree-Based Online Speaker Clustering," *Lect. Notes Comput. Sci.*, vol. 4477, 2007, pp. 555-562.

[32] K. Chen et al., "Fast Speaker Adaptation Using Eigenspace-Based Maximum Likelihood Linear Regression," *Proc. ICSLP*, 2000, pp. 742-745.

[33] B. Mak, J.T. Kwok, and S. Ho, "A Study of Various Composite Kernels for Kernel Eigenvoice Speaker Adaptation," *Proc. ICASSP*, vol. 1, 2004, pp. 325-328.

Mohammad Hossein Moattar received the BSc in computer engineering from Azad University of Mashhad, Mashhad, Iran, in 2003. He received the MSc and PhD in computer engineering from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 2005 and 2011, respectively. His research interests include speech and signal processing and audio indexing and retrieval.

Mohammad Mehdi Homayounpour received the BSc in electronics engineering from Amirkabir University of Technology in 1986 and the MSc in telecommunications from Khajeh Nasireddin Toosi University, Tehran, Iran, in 1989. He received his PhD in electrical engineering from University of Paris 11 (Orsay), Paris, France, in 1995. Since 1995, he has been an associate professor of the Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Iran. His current research interests are natural language processing, speech and signal processing, and audio indexing. He is a member of the Computer, Information and Telecommunication, and Cryptology Societies of Iran.